Cryptographically Enforced Distributed Data Access Control

Luan Ibraimi

Cryptographically Enforced Distributed Data Access Control

Luan Ibraimi

Composition of the Graduation Committee:

Prof. Dr. Ir.	A.J. Mouthaan	Universiteit Twente
Prof. Dr.	P.H. Hartel	Universiteit Twente
Prof. Dr.	W. Jonker	Universiteit Twente
Dr.	S. Nikova	Universiteit Twente and Katholieke Universiteit Leuven
Prof. Dr. Ir.	B. Preneel	Katholieke Universiteit Leuven
Dr. Ir.	B. Schoenmakers	Technische Universiteit Eindhoven
Dr. Ir.	R.N.J. Veldhuis	Universiteit Twente
Prof. Dr.	D. Pavlović	Royal Holloway, University of London and Universiteit Twente

This research is conducted within the Secure Patient-Centric Management of Health Data project supported by Philips Research and the University of Twente.

CTIT Ph.D. Thesis Series No. 11-208
Centre for Telematics and Information Technology
P.O. Box 217, 7500 AE
Enschede, The Netherlands.

SIKS Dissertation Series No. 2011-41
The research reported in this thesis has been carried out under the auspices of SIKS, the Dutch Research School for Information and Knowledge Systems.

ISBN: 978-90-365-3228-0
ISSN: 1381-3617 (CTIT Ph.D. thesis Series No. 11-208)
DOI: 10.3990/1.9789036532280
http://dx.doi.org/10.3990/1.9789036532280

Typeset with LaTeX. Printed by Wöhrmann Print Service.
Cover design: Dukagjin Borova, Professional Digital Recording & Design Studio MJELLMA .

CRYPTOGRAPHICALLY ENFORCED DISTRIBUTED DATA ACCESS CONTROL

DISSERTATION

to obtain
the degree of doctor at the University of Twente,
on the authority of the rector magnificus,
prof. dr. H. Brinksma,
on account of the decision of the graduation committee,
to be publicly defended
on Friday, 21st of October 2011 at 12.45

by

Luan Ibraimi

born on 10th of April 1984,
in Struga, Macedonia

The dissertation is approved by:

Prof. Dr.	P.H. Hartel	(promotor)
Prof. Dr.	W. Jonker	(promotor)
Dr.	S. Nikova	(assistant-promotor)

Abstract

Outsourcing data storage reduces the cost of ownership. However, once data is stored on a remote server, users lose control over their sensitive data.

There are two approaches to control the access to outsourced data. *The first approach* assumes that the outsourcee is fully trusted. This approach is also referred to as *server mediated access control* and works as follows: whenever a user wants to access the stored data, the user has to provide credentials to the server. If the credentials are valid and satisfy the access control policy, the user is allowed to access the stored data. However, fully trusting the server can be dangerous since if the server gets hacked, all users data would be readable by hackers. The *second approach* reduces the trust on the server and assumes that the server is *honest-but-curious*: the server is *honest* in the sense that it stores the data correctly and makes the data available to users, and the server is *curious* in the sense that it attempts to extract knowledge from the stored data. This approach is also referred as *cryptographically enforced access control* because it relies on encryption techniques to enforce an access control policy. The main idea of this approach is to map an access control policy into an encryption key, and then to encrypt the data under the encryption key such that only authorized users who possess a decryption key can access the data in clear. Even if the server gets hacked, user data are secure since the data are encrypted.

In this thesis we focus on the second approach and propose new encryption schemes for enforcing access control policies with significant advantages over existing ones. In particular, we push the limits of three cryptographic primitives: proxy re-encryption, attribute-based encryption and public-key encryption. Our contributions can be summarized as follows:

1. We propose a proxy re-encryption scheme which enables the delegator to provide a fine-grained access control policy. Proxy re-encryption is a cryptographic primitive developed to delegate the decryption right from one party (the delegator) to another (the delegatee). In our scheme, the delegator can categorize messages into different types and delegate the decryption right of each type to the delegatee through a proxy.

2. We propose two ciphertext-policy attribute-based encryption schemes which are more efficient and at least as expressive as the existing state-of-the-art schemes. In ciphertext-policy attribute-based encryption the data is encrypted under an access control policy defined over attributes. A user can decrypt the ciphertext only if the attribute set of her secret key satisfies the access control policy of the ciphertext.

3. We propose a ciphertext-policy attribute-based encryption scheme in which the secret keys of dishonest or compromised users are revoked.

4. We propose a ciphertext-policy attribute-based encryption scheme that allows users to update the access control policy of the ciphertext without decrypting it.

5. We propose a public-key encryption scheme that allows the secret key holder to delegate to the server the power to search her ciphertexts for possible malware without decrypting it.

Samenvatting

Het outsourcen van data opslag verlaagt de gemaakte kosten. Echter, zodra de data op een externe server wordt opgeslagen, verliezen de gebruikers de controle over hun gevoelige data.

Er zijn twee aanpakken om de toegang tot de externe data te beheersen. *De eerste aanpak* gaat uit van een volledig vertrouwde externe partij. Deze aanpak wordt ook wel *server mediated access control* genoemd en werkt als volgt: wanneer een gebruiker toegang tot de opgeslagen data will hebben, dient deze gebruiker zijn credentials aan de server te tonen. Als de credentials geldig zijn en ze aan de access control policy voldoen, dan krijgt de gebruiker toegang tot de data. Echter, het volledig vertrouwen van de server is gevaarlijk omdat als de server gehacked wordt, alle data van de gebruiker te lezen zijn door de hackers. *De tweede aanpak* vereist minder vertrouwen in de server en neemt aan dat de server *honest-but-curious* is: de server is *honest* in de zin dat deze de data correct op slaat en beschikbaar stelt aan de gebruikers, en de server is *curious* in de zin dat deze informatie probeert te verkrijgen uit de opgeslagen data. Deze aanpak wordt ook wel *cryptographically enforced access control* genoemd en maakt gebruik van encryptie technieken om een access control policy af te dwingen. Het belangrijkste idee achter deze aanpak is om een access control policy aan een encryptie sleutel toe te wijzen. Vervolgens wordt de data geencrypt met de encryptie sleutel, zodat alleen geautoriseerde gebruikers die de decryptie sleutel hebben de ontsleutelde data kunnen bemachtigen. Zelfs als de server gehacked wordt, dan is de gebruikers data veilig aangezien deze encrypted is.

In dit proefschrift richten we ons op de tweede aanpak en stellen we nieuwe encryptie schema's om access control policies af te dwingen voor, die significante voordelen hebben boven bestaande encryptie schema's. In het bijzonder zoeken we de grenzen op van drie cryptografische primitieven: proxy re-encryption, attribute-based encryption en publieke sleutel encryption. Onze bijdragen kunnen als volgt samen gevat worden:

1. We stellen een proxy re-encryption schema voor die de delegator in staat stelt om een gedetailleerde access control policy aan te geven. Proxy re-encryption is

een cryptografische primitief ontwikkeld om het decryptie recht van een partij (de delegator) aan een andere partij (de delegatee) te delegeren. In ons schema kan de delegator berichten in verschillende types categoriseren en het decryptie recht van elk type delegeren aan een delegatee via de proxy.

2. We stellen twee ciphertext-policy attribute-based encryption schema's voor die efficienter zijn en minstens zo expressief als de reeds bestaand state-of-the-art schema's. In ciphertext-policy attribute-based encryption is de data encrypted onder een access control policy gedefinieerd over de attributen. Een gebruiker kan de ciphertext alleen decrypten als de attributen verzameling van haar geheime sleutel aan de access control policy van de ciphertext voldoet.

3. We stellen een ciphertext-policy attribute-based encryption schema voor waarin de scret keys van oneerlijke of gecompromiteerde gebruikers ingetrokken kunnen worden.

4. We stellen een ciphertext-policy attribute-based encryption schema voor die de gebruikers in staat stelt de access control policy bij te werken zonder de ciphertext te hoeven decrypten.

5. We stellen een publieke sleutel encryption schema voor die de houder van de geheime sleutel in staat stelt om de mogelijkheid om malware in ciphertexts te zoeken zonder deze te decrypten aan de server te delegeren.

Acknowledgment

The thesis is finally finished! I am very happy that I am writing the acknowledgment, indicating that it marks the end of a very important chapter in my life.

Four years ago (around 2007), when I was doing my master studies in KTH - Sweden, the program coordinator suggested me the EEMCS faculty of the University of Twente as a potential place to do my PhD studies. Now after I have finished the thesis, I can say that I made the right decision when accepting to be part of the Twente University staff. The last four years have been the most rewarding years in my life. It was a true adventure; a job I greatly enjoyed, visiting places that before my PhD I could have seen in dreams only, and working in different places with a lot of people that I would like to thank.

First of all I would like to thank my promotores Pieter Hartel and Willem Jonker - thank you for believing in my professional capabilities and giving me the freedom to pursue my research. Your comments, discussions and feedback greatly influenced this thesis. Interesting meetings that we had almost every week helped me to better organize my ideas and improve my critical thinking. I am sure that the knowledge that I got from you will continue to influence every step of my professional live in the future. Willem, I am amazed with your out-of-the-box thinking and the ability to understand how things work immediately! The busy schedule did not prevent you to help me. Pieter, I am amazed with your energy and enthusiasm! You had always time to read my papers and provide valuable comments. I also would like to thank Pieter's wife Marijke for making me and my wife feeling home.

I want to thank members of the thesis committee for accepting to be part of the committee, and for taking their time and effort to read this thesis and providing me with many valuable comments.

Qiang, my daily supervisor in the first year of my PhD, it is your introduction to the topic of Identity-Based Encryption that led me to this topic and dissertation. The third chapter of this thesis is influenced a lot by your work and despite divergences of opinions that we had sometimes, I really enjoyed working with you. Thank you!

The second year of my PhD I spent entirely at Philips Research in Eindhoven.

During my internship at Philips, I enjoyed working with Milan Petković and Asim Muhammad, which resulted in a number of publications and patent applications. Thank you Milan and Asim, my thesis would not have been at this shape without your support. From Philips, I also want to thank all the members of the Information and System Security group for their hospitality.

Svetla, my daily supervisor in the second, third and fourth year of my PhD, thank you for helping me to improve the quality of my work by pushing me to aim higher and have papers published in well known conferences. Thanks also for helping me to get the internship in New York University (NYU). The internship that I had at NYU was a unique experience for me since I had the opportunity to talk and cooperate with the most brilliant people in the field of cryptography. And of course living in New York for three months was a great experience on its own. While we are at NYU, I want to thank Yevgeniy Dodis for hosting me. A very special thank is for Jöel Alwen, who despite being busy with his work, was able to work with me every day. I thank also Sze-Ming Chow for working with me the last week of my internship at NYU.

I thank my colleges from the Distributed and Embedded Security (DIES) group for the nice time we spent together: Sandro, Dusko, Frank, Damiano, Jonathan, Wolter, Emmanuele, Begül, Trajce, Michael, Dina, Andre, Arjan, Stefan, Saeed, Ivy, members of SecurityMatters, and former members: Ileana, Ayse, Richard, Marcin, Jeroen and Mohammed. Special thank goes to the Database (DB) group for inviting me to many dinners, and to the DB group secretary, Ida, for helping me with administrative issues. I thank also Nienke, Bertine, and Suse, who always helped me when Ida was not available. Thank goes also to my academic friends from the Hasso-Plattner-Institut: Sebastian Roschke, Feng Cheng and Christoph Meinel, and to Ruth Griepink for helping me to improve my written English.

I thank my Albanian friends at the Twente University: Arta, Aurel and Alma, for lunches that we had together, and the Albanian Scholarship Foundation for providing me a scholarship for book payment at the last stage of my PhD. I thank also my sister, my family-in-law, my relatives and friends in Macedonia for their motivation and continual interest in the progress of my studies.

Prindër të dashur faleminderit për dashurinë dhe përkrahjen tuaj të pakushtëzuar. Unë ju kam patur krahë në çdo hap dhe kam ndarë me ju çdo sakrific dhe sukses. Jam i sigurt se ambiciet e mia pa përkrhajen tuaj do kishin mbetur ëndrra të parealizuara.

Last and foremost I want to thank my sweet wife Evisa. Evisa, I shared with you every moment of my PhD life and I am forever grateful for your support and patients that you had during this time. I am blessed to have you next to me.

Enschede,
October 2011

Luan Ibraimi

Contents

Chapter 1

Introduction

This chapter provides an introduction and the motivation for our research. This chapter also describes the main research question, the contributions and the overall structure of the thesis.

1.1 Motivation

With the recent developments in cloud computing, a large number of users have been outsourcing their storage to third parties. Cloud storage providers, such as Amazon S3, provide users with the possibility to store and access their data anytime from anywhere. While outsourcing the storage is convenient and cost-effective, the outsourced data might be sensitive and an inappropriate disclosure may cause serious problems for users. Therefore, the proper enforcement of data access control is of central importance.

Access control (AC) mechanisms comprise a large set of technologies, which include mechanisms to authenticate and authorize individuals or systems to access data or resources. In the literature we find two approaches to enforce AC: *server mediated access control* and *cryptographically enforced access control* (see Table 4.1 for a comparison). To understand how these approaches work in practice, let us envisage the following scenario:

There is an online storage server maintained by a third party. The server is trusted to store the data correctly and to allow authorized users to access or update the data. Alice wants to store her Personal Health Records (PHR) on the server so that she can access them from everywhere using an Internet connection. In addition, Alice wants to share some of her health data with other users, including her general practitioner and some of her family members or friends. PHRs may contain different data categories which are sensitive such as details of Alice's disease, drug usage, sexual preferences, etc. Therefore, Alice is worried whether her PHRs will be treated as confidential by the party that runs the server.

In practice, examples of online storage servers which allow patients to store and share their PHRs are web-based PHR systems, such as Microsoft HealthVault.

1.1.1 Server Mediated Access Control

To protect her data, Alice has to specify an AC policy which defines the list of users and their permissions. The server uses the AC mechanism to enforce the specified policy. Typically the AC checks whether the user credentials satisfy the AC policy before they are allowed to access Alice's data. While this AC mechanism is an accepted way to protect the data as long as Alice fully trusts the server, this approach has several limitations when the server cannot be trusted:

- *the server has access to the plain data.* This might not be a problem if Alice uses the server to store public information, but it becomes a problem when Alice stores sensitive information such as her PHRs. In practice, there are a number of initiatives from different governments around the world, such as the directive on privacy and electronic communications in the U.S. known as the Health Insurance Portability and Accountability Act (HIPAA) [99], which specify rules and standards to achieve security and privacy of health data and EU Data Directive, which specify rules for protection of personal data within the EU. However, web-based PHR systems are not covered by these legislations, thus, companies running these systems have more freedom when it comes to sharing the stored data.

- *the data gets compromised once the server gets compromised.* If the server gets infected by a virus, the virus might be able to avoid or turn off the AC mechanism. An inappropriate disclosure of Alice's data can change her life, and there may be no way to repair such harm financially or technically. For instance, if Alice has some disease and a prospective employer learns this, then she might be discriminated when looking for a job. What makes things even worse is that in reality the data is stored in a distributed fashion across many storage servers (e.g. in cloud computing the data is stored and processed in different places). Hence, it is enough for only one server to get compromised, for Alice's data to leak.

- *the AC policy is not bound to the data.* The AC mechanism is only installed on the server, thus the AC policies are not enforced when the data travels from the server to the recipient or between servers in a distributed system. In particular, users do not have mechanisms to bind the AC policy to the data, but they can only consent to the applicable AC policy and then rely on the server to enforce it.

To overcome the above limitations, recent proposals in the literature (including this thesis) do not rely on the fully trusted server to enforce AC policies. Instead, they exploit the use of cryptography and they assume that the server is not fully trusted.

Table 1.1: *Data Access Control Enforcement.*

AC Enforcement	Confidentiality against a (compromised) Server	Policy Bound to the Data	Expressive Policies
Server Mediated AC	**NO**	**NO**	**YES**
Cryptographically Enforced AC	**YES**	**YES**	**NO**

1.1.2 Cryptographically Enforced Access Control

The cryptographically enforced AC approach relies on cryptographic primitives to enforce the AC policy under the assumption that the server itself is *honest-but-curious*; it is honest in the sense that it will store the data correctly and will follow the protocol, and it is curious in the sense that it wants to learn the content of the stored data.

When using cryptographically enforced access control Alice protects her data as follows. Alice maps an AC policy to a key and then *locks* the data with a key such that the data becomes self-protected (i.e. the AC policy is bound to the data). After that, Alice sends her locked data to the server. Since the data is locked, every user (i.e. including dishonest users) can get the locked data, however, only users who have the right key can *unlock* the locked data and access its content. This is important for situations when the data is stored in a distributed fashion (such as cloud storage) across many storage servers where, even if all servers get corrupted, the stored data will not get compromised. Note that under this approach the server does not obtain the key, otherwise it would have the same limitations as the *server mediated access control* approach.

Encryption is an indispensable cryptographic tool, which enables Alice to lock her data and which guarantees that only authorized users can unlock the data. The original purpose of encryption is to allow two parties, the sender and the receiver, to communicate privately over a medium, which might be under the control of an adversary. An example of such a medium is the Internet. In an encryption scheme, whenever a sender transmits a message (referred to as the plaintext) to the receiver, it runs the encryption algorithm which takes as input the plaintext and the encryption key and outputs a scrambled form of the plaintext, called the ciphertext. The receiver runs the decryption algorithm which takes as input the ciphertext and the decryption key and it outputs the original plaintext. In the context of enforcing AC policies, if Alice uses an encryption scheme to **map her AC policy into an encryption key**, then she is assured that only users who have the right decryption key can obtain the data.

There are two types of encryption schemes: symmetric-key and asymmetric-key. We now discuss how these encryption schemes can be used by Alice to enforce AC policies and analyze their limitations. Motivated by these limitations, we then introduce our main research question.

Access Control using Symmetric-Key Encryption

In symmetric-key encryption [84, 39], also known as private-key encryption, the encryption key and the decryption key are the same. This implies that the key must be kept secret. Alice can use symmetric-key encryption to enforce her AC policies in the following manner:

- Alice can generate a secret key and then use the key to encrypt her PHRs. Alice has "only" to distribute the secret key to authorized parties in order to allow them to access her data. The limitation of this approach is that the data sharing is *all-or-nothing* and Alice does not have the flexibility to choose a *fine-grained* AC policy. For instance, Alice does not have the option to restrict her doctor to access only some categories of her PHRs. Yet another drawback is that Alice has to distribute the secret key to all intended users and if only one user is compromised then all her PHRs are compromised.
- Alice can generate one key per category, and then distribute keys to authorized parties such that they are only allowed to access the specified category. Unfortunately, this approach, similar to the first approach, is too complex since it requires heavy key pre-distribution. For instance, if Alice wants to allow n parties with different access rights, then Alice has to create and securely distribute n keys.

Although symmetric-key encryption is efficient in computation, the key management problem makes it unsuitable for enforcing expressive AC policies when there are a large number of users involved, which is usually the case when managing PHRs.

Access Control using Asymmetric-Key Encryption

In asymmetric-key encryption [43, 85, 45], also known as public-key encryption (PKE), the encryption key is public and is mathematically related to the decryption key which is secret. In particular, one user publishes the public key and everyone can run the encryption algorithm and convert the plaintext into the ciphertext. However, only the user who knows the decryption key can convert the ciphertext into the plaintext.

Alice can use asymmetric-key encryption to enforce her AC policies in the following manner:

- Alice can generate a key pair and use the generated public key to encrypt her PHRs. To enable authorized parties to access her data, Alice first has to download from the server the category of the encrypted data the party is interested to access and then re-encrypt it under the public key of the intended party. The drawback of this approach is that Alice has to stay online and be involved in every request (e.g. from her doctor, family member) to decrypt and then re-encrypt her PHRs.

- Alice can directly encrypt her data using the public key of the authorized party. However, the problem of this approach is that the association between a user and a public key is one-to-one. This implies that when Alice wants to allow the same data to be accessed by n users, Alice has to encrypt the same data n times under n different public keys. This is not efficient both from the communication and processing point of view. Yet another drawback of this approach is that Alice has to know the identity of the recipient beforehand. However, there are many situations when access to the data should depend on user attributes and not on user identities.

In contrast to the symmetric-key setting, in asymmetric-key setting the encryption key is public, hence it can be sent from one user to another over a public medium without compromising the security. This implies that users do not need to share a key in a secret way prior to their communication.

Access Control using Advanced Asymmetric-Key Encryption

Although cryptographically enforced AC, compared to server mediated AC, provides better security when enforcing AC policies, it suffers from a major limitation. As we described above, traditional cryptographic schemes suffer from the key management problem when they have to enforce expressive AC policies. Therefore, a number of more advanced asymmetric-key encryption schemes have recently been proposed in the literature, including proxy re-encryption (PRE) and attribute-based encryption (ABE).

In PRE, one party (the delegator) assigns a key to a proxy to re-encrypt all messages encrypted with her public key such that the re-encrypted ciphertext can be decrypted using another party's (the delegatee) private key. In the context of enforcing AC policies, Ateniese et al. [13] show how Alice (the delegator) can use PRE to enforce her AC policy. Alice first encrypts all her data using her public key and then uploads the encrypted data to an honest-but-curious server. To allow authorized users to access her data, Alice computes re-encryption keys and sends them to the proxy. Whenever a user wants to access Alice's data, the proxy checks whether it has a re-encryption key. If so, the proxy re-encrypts (without decrypting) Alice's encrypted data so that authorized users can decrypt the data using their private keys. Note that, unlike in traditional PKE, Alice does not have to download and then re-encrypt the encrypted data; instead she has to compute re-encryption keys only. This is important for resource constrained devices that are capable to perform limited computation, such as to compute re-encryption keys only, but are not capable to perform more advanced computations, such as to download and re-encrypt the encrypted data.

The problem with all existing PRE schemes is that the proxy, once it gets one re-encryption key, is able to re-encrypt all Alice's ciphertexts so that other users (i.e. delegatees) can decrypt them using their private keys. Thus, Alice does not have the flexibility to define a fine-grained AC policy.

ABE, introduced by Waters and Sahai [104], extends traditional encryption such that a user is identified by a set of attributes instead of a name. In ABE, both a user secret key and the ciphertext are associated with a set of attributes. The secret key can decrypt the ciphertext only if both sets have at least t (threshold value) attributes in common. Goyal et al. [50] define two flavors of ABE: Ciphertext-Policy Attribute-Based Encryption (CP-ABE) and Key-Policy Attribute-Based Encryption (KP-ABE). In CP-ABE [20], a user encrypts the data according to a predicate (i.e. AC policy) defined over attributes, such that only the user who has a secret key associated with the attribute set which satisfies the predicate can decrypt the ciphertext. For example, Alice can encrypt her data according to an AC policy $\tau = (a_1 \wedge a_2) \vee a_3$. Another user, say Bob, can decrypt Alice's data only if his secret key is associated with one of the following attribute sets: (a_1, a_2), (a_3) or (a_1, a_2, a_3). In KP-ABE [104], the idea is reversed such that the ciphertext is associated with the attribute set and the secret key is associated with the predicate defined over attributes. For example, Alice can receive a secret key associated with the predicate $\tau = (a_1 \wedge a_2) \vee a_3$ and can decrypt every ciphertext that is associated with one of the following attribute sets: (a_1, a_2), (a_3) or (a_1, a_2, a_3). In general, CP-ABE is more practical than KP-ABE since it allows the encryptor to define the AC policy. Therefore in this thesis we focus on CP-ABE.

The main problem with all existing ABE schemes is that they are designed to work only for static environments. Problems arise when:

- secret keys eventually have to be revoked. Existing ABE schemes provide limited support for key revocation, a feature which is becoming increasingly important in modern systems. Key revocation may be necessary due to the following reasons: a) an attribute is not valid because it has expired, for instance, the attribute "project manager-January 2011" is valid until January 2011, b) a user is misusing her secret key, for instance, Alice might give a copy of her secret key to Bob who is not a legitimate user, or c) a user has lost her secret key.

- AC policies change frequently. Existing ABE schemes do not have efficient mechanisms to update AC policies.

1.2 Research Statement

As mentioned above, there are various asymmetric-key encryption schemes that can cryptographically enforce an AC policy, including public-key encryption (PKE), proxy re-encryption (PRE) and attribute-based encryption (ABE). Motivated by their limitations, in this thesis we pose the following main research question:

> **Research Question:** How to construct cryptographic schemes that can enforce distributed data access control efficiently in dynamic environments?

The above research question asks to improve existing techniques in the following aspects: i) for PRE to be more expressive without sacrificing the efficiency and ii) for ABE to be efficient, and also to be suitable for dynamic environments by supporting updating AC policies and revoking keys.

Note that we do not need to extend i) PRE with respect to updating AC policies and revoking keys and ii) PKE with respect to efficiency, revoking keys, expressivity and updating AC policies. For i), updating ciphertexts in PRE is already included in the definition of PRE, thus updating AC policies, without decrypting the ciphertext, is supported by *default* in PRE. In addition, revoking keys in PRE is easily achieved by using existing revocation techniques in PKE. For ii), there are many schemes in the literature which are efficient and address key revocation. Therefore in this thesis there is no need to provide another efficient PKE scheme and to address key revocation. In addition, updating the ciphertext in PKE is covered by PRE. Indeed, a PRE scheme is a PKE scheme which supports updating the ciphertext without decrypting it. Finally, the expressivity of PKE is covered by PRE and ABE; having an expressive PKE was the initial motivation when introducing PRE and ABE.

We divide the main research question into the following sub-questions:

Q1. *How to construct a* PRE *scheme which can support* ***fine-grained*** *AC policies, without sacrificing efficiency?*

A PRE scheme should guarantee that a user is capable of specifying fine-grained access control policies such that they can selectively share their data with other parties. What makes it challenging to construct such a scheme is the requirement that the delegator has to use only one key-pair and still being able to provide fine-grained re-encryption capability to his proxy.

Q2. *How to construct* ABE *schemes which are* ***efficient****, and support* ***revoking keys*** *and* ***updating*** *AC policies*?

Realizing efficient ABE schemes is important for resource constraint devices. In general, ABE schemes are more expensive than traditional PKE and PRE schemes since in ABE the ciphertext is associated with a predicate over attributes (i.e. the ciphertext is intended for many users) while in PKE and PRE the ciphertext is associated with an identity (i.e. the ciphertext is intended for one user). What is challenging when constructing ABE schemes, which also affects efficiency, is collusion. A collusion resistent scheme does not allow users to combine their secret keys and decrypt a ciphertext that colluding users separately cannot decrypt. Had it not been for the collusion resistance requirement, it would have been possible to construct ABE from PKE directly.

Key revocation is an important requirement in the domain of AC. Users whose keys are revoked are excluded from the right to access a resource even if they have the right attributes which satisfy the AC policy. In ABE, key revocation is hard due to the rich structure of the ciphertext and the secret key.

In practice there are situations in which the data owner wants to update the AC policy such that new users are allowed to access the data while some old users are not allowed access anymore. There should be efficient mechanisms to enable users to update the AC policy of the ciphertext without decrypting it. Downloading the data from the server and then re-encrypting them under a new AC policy is not efficient. Note that updating ciphertexts is not the same as key revocation since updating implies revoking old users only for that specific ciphertext.

Since we assume that the data is encrypted before it is stored on an *honest-but-curious* server, searching the encrypted data is considerably harder than searching the plaintext data. In addition, while encryption helps honest users to protect their sensitive data, the hardness of processing encrypted data without decrypting it, helps attackers to hide their viruses from being analyzed by Intrusion Detection Systems (IDS). Following our initial scenario, consider a situation when Alice's doctor encrypts a treatment plan for Alice with the public key of Alice, and stores the plan in the server, such that only Alice will be able to learn the contents of the data. However, the computer of the doctor is infected and unbeknown to the doctor it also embeds malware into Alice's plan. The server cannot scan the data for malicious content as the data is encrypted so the burden is on Alice to do the scan. However, this is not efficient. Once the decryption is performed by Alice, the infected data compromises Alice's computer. Since all Alice's secret keys will get compromised, the stored data in the server will get compromised as well. Thus, this attack renders the cryptographically enforced access control approach insecure.

To benefit fully from the advantages of cryptographically enforced access control, we have to look for solutions that allow Alice to delegate the searching power to the server in order to search Alice's ciphertexts (i.e. ciphertexts which are intended for Alice and created by other users) for malicious content. Therefore in this thesis we address the following sub-question:

Q3. *How to delegate the power to search in the encrypted data?*

One way to delegate the search in the encrypted data is to send the decryption key to the server. Once the server receives the decryption key, it decrypts the data and then searches on it. However, the drawback of this approach is that the server accesses sensitive plaintext data. To address this problem there is a need for an efficient solution allowing the server to search on the encrypted data without decrypting it.

1.3 Contributions

In this thesis we propose cryptographic schemes based on pairings on elliptic curves over finite fields. Our schemes enrich current cryptographically enforced access control approach, as illustrated in Figure 1.1. Our high-level goal is to design new practical yet provably secure cryptographic schemes. We highlight our main contributions as follows:

1. We propose a PRE scheme which enables the delegator to provide a fine-grained AC policy (Q1).

2. We propose two CP-ABE schemes which are more efficient and at least as expressive as the existing state-of-the-art CP-ABE schemes (Q2).

3. We propose a mediated CP-ABE scheme in which dishonest or compromised users are immediately revoked (Q2).

4. We present a CP-ABE scheme which allows users to update the AC policy of the ciphertext without decrypting it (Q2).

5. We propose a PKE scheme which allows the secret key holder to delegate to the server the power to search her ciphertexts for malware without decrypting it. We are the first to make a connection between searching on encrypted data techniques and detecting encrypted malware (Q3).

1.4 Outline of the Thesis

We organize the thesis into eight chapters. The outline of the thesis is as follows:

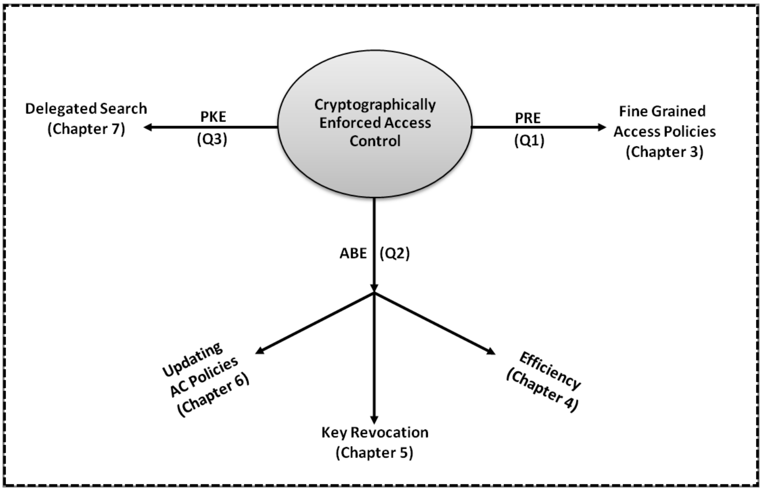
Cryptographically Enforced Access Control
Delegated Search (Chapter 7)
PKE
(Q3)
PRE
(Q1)
Fine Grained Access Policies (Chapter 3)
ABE
(Q2)
Updating AC Policies (Chapter 6)
Key Revocation (Chapter 5)
Efficiency (Chapter 4)

Figure 1.1: *Extending the Cryptographically Enforced Access Control approach.*

Preliminary Topics – Chapter 2

We present relevant background material and notations that are necessary to understand the remainder of the thesis. In particular, we give a brief introduction to relevant notions from mathematics and complexity theory. We also review security models that we use when we prove the security of our schemes. Finally we formalize identity-based encryption along with its security definitions.

Fine-Grained AC policy for PRE – ***Chapter 3***

In this chapter we present the first contribution and address the first sub-question. In particular, we propose a type-and-identity-based PRE scheme that enables the delegator to implement different AC policies for his ciphertexts against his delegatees. To attain our goal, in the proposed scheme, the delegator can categorize his messages into different types, and delegate the decryption right of each type to the delegatee through a proxy. One benefit of our scheme is that the delegator only needs one key pair to provide a fine-grained re-encryption capability to his proxy. In other words, the delegator needs only one key pair to provide a fine-grained AC policies for his ciphertexts against his delegatees. The other benefit is that there is no further assumption on the proxy compared to existing proxy re-encryption schemes. The contents of this chapter is adapted from two published papers: a workshop paper [5] and a journal paper [7].

Efficient* ABE *Schemes – Chapter 4

In this chapter we present the second contribution and address the efficiency part of the second sub-question. In particular, we propose two CP-ABE schemes which are more efficient than existing state-of-the-art schemes. The first scheme can express any policy represented by a Boolean formula involving conjunctions and disjunctions. In the second scheme, we extend the expressivity of the first scheme by including threshold operators. Both schemes are secure under standard complexity assumptions. We provide a comparison of our schemes with existing CP-ABE schemes and show that our schemes are more efficient, especially the computational work done by the decryptor is reduced. The contents of this chapter is adapted from a published conference paper [6].

Key Revocation in* ABE *– Chapter 5

In this chapter we present the third contribution and address the key revocation part of the second sub-question. In particular, we propose a mediated CP-ABE scheme which allows an authority to revoke secret keys. In the proposed scheme the secret key is divided into two shares, one share for the mediator and the other one for the user. To decrypt the encrypted data, the user must contact the mediator to receive a decryption token. The mediator keeps an attribute revocation list and refuses to issue the decryption token for revoked attributes. Without the token, the user cannot decrypt the ciphertext, therefore the attribute is implicitly revoked. As an application of the proposed scheme, we show a general architecture of a web-based PHR which helps patients to store and distribute their medical records securely. A precursor to this chapter appears in the workshop paper [4].

Updating AC Policies in* ABE *– Chapter 6

In this chapter we present the fourth contribution and address the updating AC policies part of the second sub-question. In particular, we present a new variant of the CP-ABE scheme which allows users to update the AC policy of the encrypted data without decrypting the ciphertext. The scheme uses an honest-but-curious entity, called a proxy, to re-encrypt the encrypted data according to a new AC policy such that only users who satisfy the new policy can decrypt the data. One of the distinctive features of the proposed scheme is that it is collusion resistant. The collusion resistance feature implies that even if the proxy and delegate collude they cannot generate a new secret key. This chapter is built on previous work presented in a patent application and a conference paper [1].

PKE ***with Delegated Search – Chapter 7***

In this chapter we present the fifth contribution and address the third subquestion. In particular, we propose a PKE scheme where the ciphertext is both searchable and decryptable (in existing searching on encrypted data schemes the ciphertext is searchable only). We construct a mechanism that enables the secret key holder to provide trapdoors to the server (i.e. delegate the power to the server) such that the server, given an encrypted data and a word, is able to search whether the encrypted data contains the word, without decrypting it. Having both searchable and decryptable ciphertexts is crucial since the server can search the entire contents of the message, in contrast to the existing searchable PKE schemes where the server can search only in the metadata part. We show how to apply the proposed scheme in different applications such as detecting encrypted malware and forwarding encrypted email. This chapter builds on a conference paper [3].

Conclusion – Chapter 8

In this chapter we provide conclusions and suggestions for future work.

1.5 Conclusion

The approach towards answering our research questions of Section 1.2 is by exploring proxy re-encryption (PRE), attribute-based encryption (ABE) and traditional public-key encryption (PKE). The main goal is to achieve better efficiency compared to existing relevant schemes and to extend existing cryptographic primitives with new properties which are useful in practice. We also elaborate on several applications for the proposed schemes in the domain of healthcare. In general, the thesis advances the field of enforcing AC policies by proposing new schemes along with their security definitions.

Chapter 2

Preliminary Topics

In this chapter we give the short background necessary to understand the remainder of the thesis. We start the chapter by giving a brief introduction to abstract algebra; in particular we explain algebraic structures such as groups, subgroups and fields. Next, we briefly review elliptic curves and bilinear maps. We also review computational complexity theory and related complexity assumptions under which our schemes are proven to be secure. Then, we explain security models. In particular, we discuss the standard model and two idealized models: the random oracle model and the generic group model. Finally, we explain identity-based encryption and its security definitions.

2.1 Abstract Algebra

A *group* $\mathbb{G}$ is a set of elements with an associated binary operation which satisfies the four group axioms: closure, associativity, the identity property, and the inverse property [70]. We write $(\mathbb{G}, *)$ to denote groups whose binary operation is a multiplication and $(\mathbb{G}, +)$ to denote groups whose binary operation is an addition. Sometimes we might abuse the notation and write only $\mathbb{G}$ for $(\mathbb{G}, *)$. A group $\mathbb{G}$ with a finite set of elements is called a finite group. The number of elements in a group $\mathbb{G}$ is the order of group $\mathbb{G}$. A cyclic group is the group which can be generated from a single element $g \in \mathbb{G}$ such that, when the binary operation is a multiplication, $\mathbb{G} = \langle g \rangle = \{g^i | \, i \in \mathbb{Z}\}$. This implies that for any $y \in \mathbb{G}$ there exists an integer i such that $g^i = y$. Given a non-empty subset H of the group $\mathbb{G}$ defined under a binary operation $(*)$, H is a *subgroup* of G if H is also a group under the operation $(*)$.

Let n be a positive integer. Let $\mathbb{Z}_n$ be the set of integers $\{0, 1, 2, ..., n-1\}$. If the operation in $\mathbb{Z}_n$ is addition modulo n, then the set $\mathbb{Z}_n$ is a group of order n. If the operation in $\mathbb{Z}_n$ is multiplication modulo n, then the set $\mathbb{Z}_n$ is not a group (not all elements in $\mathbb{Z}_n$ have a multiplicative inverse). Let $\mathbb{Z}_n^* = \{1, 2, ..., n-1\}$ where n is a prime number. If the operation in $\mathbb{Z}_n^*$ is multiplication modulo n, then the set $\mathbb{Z}_n^*$ is a group. If n is a *safe prime*, which means that $n = 2p + 1$ with p prime, then there is a cyclic subgroup $\mathbb{G}$ of the group $\mathbb{Z}_n^*$ of order p.

A *field* $\mathbb{F}$ is a set of elements with two binary operations, addition and multiplication, which satisfy the field axioms: associativity, commutativity, distributivity, the identity property, and the inverse property [70]. Examples of fields are the real numbers $\mathbb{R}$, the complex numbers $\mathbb{C}$ and the rational numbers $\mathbb{Q}$. A field is finite if it has a finite number of elements. The order of a field $\mathbb{F}$ is the number of elements in $\mathbb{F}$.

2.2 Elliptic Curves

Koblitz [61] and Miller [71] in their seminal work suggest the use of elliptic curves over a finite field in cryptography. The main advantage of Elliptic curve cryptography (ECC) compared to other public key cryptosystems is the short key size. For instance, the security level provided by a 160-bit key in ECC is the same as the security level provided by a 1024-bit key of the RSA cryptosystem [80, 44]. This advantage of ECC over other cryptosystems is due to the lack of efficient algorithms [12] to solve the discrete logarithm (DL) of the elliptic curve group over finite fields. On the other hand, the *index calculus* algorithm can efficiently solve the DL for multiplicative group over a finite field.

An elliptic curve E over the finite field $\mathbb{F}_q$ is the set of points (x, y) which fulfill:

$$y^2 = x^3 + ax + b \ \ mod \ \ q$$

along with the special point $\mathcal{O}$ known as the point of infinity, where $a, b \in \mathbb{F}_q$ and q is a prime power.

2.2.1 Bilinear Maps from Elliptic Curve

The application of bilinear maps to build cryptosystems is proposed by Verheul [101] and Joux [57]. Let $\mathbb{G}$ be an additive group of prime order p, and $\mathbb{G}_T$ be a multiplicative group of the same order as $\mathbb{G}$. Let P be a generator of the group $\mathbb{G}$. A pairing (or bilinear map) $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ has the following properties [27]:

1. Bilinearity: for all $u, v \in \mathbb{G}$ and any $a, b \in \mathbb{Z}_p^*$, we have:

$$e(u^a, v^b) = e(u, v)^{ab}.$$

2. Non-degeneracy: $e(P, P) \neq 1_{\mathbb{G}_T}$, where $1_{\mathbb{G}_T}$ is the identity element of the group $\mathbb{G}_T$.

3. The function e can be efficiently computed.

By modifying the Weil pairing [72] or the Tate pairing [46] on an elliptic curve E over the finite field $\mathbb{F}_q$ one can derive the map e. The reasons for modifying the

Weil and Tate pairing is because, if unmodified, then the pairing $e(P, P)$ returns the identity element $1_{\mathbb{G}_T}$. There are two well known techniques to modify Weil and Tate parings: distortion maps [102] and trace maps [28]. Distortion maps are applicable only to a specific class of curves called supersingular curves while the trace map is more general since it is applicable to all curves.

Most of the pairings used in this thesis will have both inputs from the same group $\mathbb{G}$, or $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$; this type of pairing is also known as Type-1 pairing. Type-2 pairings are asymmetric pairings where $e : \mathbb{G} \times \Gamma \rightarrow \mathbb{G}_T$ and $\mathbb{G} \neq \Gamma$, but there is an efficiently computable homomorphism $\psi : \mathbb{G} \rightarrow \Gamma$. Type-3 pairing are asymmetric pairings where there is no known efficiently computable homomorphism $\psi : \mathbb{G} \rightarrow \Gamma$.

2.3 Complexity Theory

An encryption scheme is perfectly secure if it is *impossible* for a *computationally unbounded* adversary to extract any information about the plaintext from the ciphertext. In terms of information theory, this means that the amount of entropy for the plaintext given the ciphertext is the same as the amount of entropy for the plaintext when the ciphertext is not given. Such schemes are also called *information-theoretically secure* since their security can be proven purely using information theory. Shannon [90] proved that the main requirement for a scheme to be perfectly secure is to have a key space which is at least as large as the message space. The key space is the set of all keys that can be computed by the key generation algorithm. The message space is the set of all messages that can be chosen during the encryption phase. This requirement also implies that during the encryption phase the length of the key should be the same as the length of the plaintext. The other requirement is that the key should be used only once. These requirements are too strong for most practical use.

In this thesis we follow a more practical approach when proving security, which assumes that adversaries are computationally bounded and run in polynomial time. In this setting, the word *impossible* is substituted with *infeasible*. This implies that given enough time and computation these schemes can be broken. Such schemes are known as *computationally secure* and their security is proven under certain complexity assumptions.

Complexity theory classifies computational problems according to the resources required to solve them. Usually the resources being considered are space and time. An important complexity theory notion is the *negligible function*. In modern cryptography *negligibility* is used to show that schemes are secure even if they can be broken with a negligible probability. All definitions in this section are adapted from a textbook [70].

Definition. *A function $\epsilon(\lambda)$ is said to be negligible in the parameter λ if for every integer $c \geq 0$ there exists an integer $\lambda_c > 0$ such that $\varepsilon(\lambda) < \lambda^{-c}$ for all $\lambda > \lambda_c$.*

In cryptography λ is called a *security parameter*. When we design our cryptographic schemes in the following chapters, the role of λ is very important since the size of λ influences many other parameters, including the level of security, the size of the secret keys, the size of the finite groups, the running time of an algorithm, etc.

Algorithm analysis estimates the running time needed by any algorithm to solve a given computational problem. The running time of an algorithm is a function associating the input length to the number of steps executed before the algorithm terminates. The "worse-case running time" is important in complexity theory since it represents the upper bound (i.e. the worst case) on the running time of the algorithm for any input. When analyzing algorithms it is usual to estimate their complexity using asymptotic measures; this is reflected by the use of the big-O notation.

Definition. *Let $f, g : \mathbb{R} \to \mathbb{R}$ be functions in the parameter λ. Then $f(\lambda) = O(g(\lambda))$ if there exist positive integers c and λ' such that $f(\lambda) \leq c \cdot g(\lambda)$ for all $\lambda > \lambda'$.*

Definition. *An algorithm is said to be a polynomial-time algorithm if its worse-case running time on input λ is of the form $O(\lambda^c)$, where c is a constant.*

Polynomial-time algorithms are considered to be efficient algorithms. In security proofs, we will often see that a polynomial-time algorithm, say the algorithm $\mathcal{B}$, runs as a subroutine another algorithm, say the algorithm $\mathcal{A}$. Since polynomial-time algorithms are closed under composition, the algorithm $\mathcal{A}$ is also a polynomial-time algorithm. It is also assumed that procedures which initialize $\mathcal{B}$ also run in polynomial time. Throughout the thesis we require all algorithms involved in a cryptographic scheme to run in polynomial-time. We also require the adversary to run in polynomial-time; indeed as mentioned above our proposed schemes are secure against polynomial-time adversaries only.

A *deterministic polynomial-time algorithm* is an algorithm with an execution path that is the same each time it gets executed on the same input. A *probabilistic polynomial-time algorithm* or a *randomized algorithm* is an algorithm that, in addition to its input, gets as input a uniformly-distributed random value. Due to the used randomness, the execution path of the probabilistic polynomial-time algorithm is different each time it is executed on the same input.

2.3.1 Complexity Assumptions

In the following we describe the complexity assumptions that we will need when we prove the security of our schemes. All these assumptions are *standard* in a sense that they have been widely accepted by the cryptographic community and are used by other authors in their security proofs.

Let $\mathcal{IG}$ be a polynomial-time algorithm that takes as input the security parameter λ and outputs the tuple $\langle \mathbb{G}, q, g \rangle$, where $\mathbb{G}$ is a cyclic group, q is the order of $\mathbb{G}$, and g is a generator of $\mathbb{G}$.

- The Discrete Logarithm (DL) assumption. The DL problem in $\langle \mathbb{G}, q, g \rangle$ is defined as follows: given (g, g^a), where a is randomly chosen from $\mathbb{Z}_q$, compute a. A polynomial-time adversary $\mathcal{A}$ has advantage ε in solving the DL problem in $\langle \mathbb{G}, q, g \rangle$ if:
$$\Pr\left[\mathcal{A}(g, g^a) = a\right] \geq \varepsilon \ ,$$

where the probability is over the random choice of $a \in \mathbb{Z}_q$ and the random bits of $\mathcal{A}$.

- The Computational Diffie-Hellman (CDH) assumption. The CDH problem in $\langle \mathbb{G}, q, g \rangle$ is defined as follows: given (g, g^a, g^b), where a, b are randomly chosen from $\mathbb{Z}_q$, compute g^{ab}. A polynomial-time adversary $\mathcal{A}$ has advantage ε in solving the CDH problem in $\langle \mathbb{G}, q, g \rangle$ if:

$$\Pr\left[\mathcal{A}(g, g^a, g^b) = g^{ab}\right] \geq \varepsilon \ ,$$

  where the probability is over the random choice of $a, b \in \mathbb{Z}_q$ and the random bits of $\mathcal{A}$.

- The Decisional Diffie-Hellman (DDH) assumption. The DDH problem in $\langle \mathbb{G}, q, g \rangle$ is defined as follows: given (g, g^a, g^b, Z), where a, b are randomly chosen from $\mathbb{Z}_q$ and Z is randomly chosen from $\mathbb{G}$, determine if $Z = g^{ab}$. A polynomial-time adversary $\mathcal{A}$ has advantage ε in solving the DDH problem in $\langle \mathbb{G}, q, g \rangle$ if:

$$\left|\Pr\left[\mathcal{A}(g, g^a, g^b, g^{ab}) = 0\right] - \Pr\left[\mathcal{A}(g, g^a, g^b, Z) = 0\right]\right| \geq \varepsilon \ ,$$

  where the probability is over the random choice of $a, b \in \mathbb{Z}_q$ and $Z \in \mathbb{G}$, and the random bits of $\mathcal{A}$.

  The DL, CDH, and DDH assumptions state that no polynomial-time algorithm can solve the DL problem, CDH problem and DDH problem, respectively, for $\langle \mathbb{G}, q, g \rangle$ generated by $\mathcal{IG}$ on input λ with a non-negligible advantage.

The above assumptions are related to each other. For instance, an algorithm that solves the DL problem can be used to solve both the CDH problem and DDH problem. An algorithm which solves CDH problem can be used to solve DDH problem. However, still it is not proven whether an algorithm that solves the DDH problem can be used to solve the CDH problem.

2.3.2 Bilinear Complexity Assumptions

Let $\mathcal{IG}$ be a polynomial-time-algorithm that takes as input the security parameter λ and outputs the tuple $\langle \mathbb{G}, \mathbb{G}_T, q, g, e \rangle$, where $\mathbb{G}, \mathbb{G}_T$ are cyclic groups, q is the order of $\mathbb{G}$, g is a generator of $\mathbb{G}$, and $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$.

- The bilinear Diffie-Hellman (BDH) assumption. The BDH problem in $\langle \mathbb{G}, \mathbb{G}_T, q, g \rangle$ is defined as follows: given (g, g^a, g^b, g^c), where a, b, c are randomly chosen from $\mathbb{Z}_q$, compute $e(g, g)^{abc}$. A polynomial-time adversary $\mathcal{A}$ has advantage ε in solving the BDH problem in $\langle \mathbb{G}, \mathbb{G}_T, q, g \rangle$ if:

$$\left|\Pr\left[\mathcal{A}(g, g^a, g^b, g^c) = e(g, g)^{abc}\right]\right| \geq \varepsilon \ ,$$

  where the probability is over the random choice of $a, b, c \in \mathbb{Z}_q$ and the random bits of $\mathcal{A}$.

- The decisional bilinear Diffie-Hellman (DBDH) assumption. The DBDH problem in $\langle \mathbb{G}, \mathbb{G}_T, q, g \rangle$ is defined as follows: given (g, g^a, g^b, g^c, Z), where a, b, c are randomly chosen from $\mathbb{Z}_q$ and Z is randomly chosen from the target group $\mathbb{G}_T$, determine if $Z = e(g, g)^{abc}$. A polynomial-time adversary $\mathcal{A}$ has advantage ε in solving the DBDH problem in $\langle \mathbb{G}, \mathbb{G}_T, q, g \rangle$ if:

  $$\left| \Pr\left[\mathcal{A}(g, g^a, g^b, g^c, e(g, g)^{abc}) = 0 \right] - \Pr\left[\mathcal{A}(g, g^a, g^b, g^c, Z) = 0 \right] \right| \geq \varepsilon \ ,$$

  where the probability is over the random choice of $a, b, c \in \mathbb{Z}_q$ and $Z \in \mathbb{G}_T$, and the random bits of $\mathcal{A}$.

  The BDH and DBDH assumptions state that no polynomial-time algorithm can solve the BDH problem and DBDH problem, respectively, for $\langle \mathbb{G}, \mathbb{G}_T, q, g, e \rangle$ generated by $\mathcal{IG}$ on input λ with a non-negligible advantage.

Note that the DDH assumption does not hold in $\langle \mathbb{G}, q, g \rangle$ when $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ (Type-1 pairing) [58]. An attacker $\mathcal{A}$ can use the properties provided by e to easily solve the DDH problem as follows: The attacker gets the tuple (g, g^a, g^b, Z) and computes $e(g^a, g^b)$. Next the attacker checks whether $e(g^a, g^b)$ is equal to $e(Z, g)$. If they are equal, $\mathcal{A}$ knows that $Z = g^{ab}$, otherwise it knows that $Z \in \mathbb{G}_T$ is a random element. The CDH assumption holds even when $\mathbb{G}$ is a bilinear group. In the literature, groups in which the CDH assumption holds and DDH assumption does not hold are called *gap groups*. Similarly, it can be shown that the DDH assumption does not hold in $\mathbb{G}$ and Γ when $e : \mathbb{G} \times \Gamma \to \mathbb{G}_T$ (Type 2 pairing) and when there is a known efficiently computable isomorphism $\psi : \mathbb{G} \to \Gamma$. In a similar way, we can show that an algorithm that solves either CDH problem or DL problem can solve both the BDH problem and DBDH problem.

However, the DDH assumption does hold in $\mathbb{G}$ and Γ when $e : \mathbb{G} \times \Gamma \to \mathbb{G}_T$ (Type 3 pairing) and when there is no known efficiently computable isomorphism $\psi : \mathbb{G} \to \Gamma$. This is covered by the following assumption:

- The Symmetric External Diffie-Hellman (SXDH) assumption. Let $\mathcal{IG}$ be a polynomial-time-algorithm that takes as input the security parameter λ and outputs the tuple $\langle \mathbb{G}, \Gamma, \mathbb{G}_T, q, g, \gamma, e \rangle$, where $\mathbb{G}, \Gamma, \mathbb{G}_T$ are cyclic groups, q is the order of $\mathbb{G}$, g is a generator of $\mathbb{G}$, γ is a generator of Γ, and $e : \mathbb{G} \times \Gamma \to \mathbb{G}_T$.

  The SXDH problem in $\langle \mathbb{G}, \Gamma, \mathbb{G}_T, q, g, \gamma, e \rangle$ is defined as follows: given (γ, g, g^a, g^b, Z) *or* $(g, \gamma, \gamma^a, \gamma^b, Z')$, where a, b are randomly chosen from $\mathbb{Z}_q$, Z is randomly chosen from $\mathbb{G}$ and Z' is randomly chosen from Γ, determine if $Z = g^{ab}$ or $Z' = \gamma^{ab}$. A polynomial-time adversary $\mathcal{A}$ has advantage ε in solving the SXDH problem in $\langle \mathbb{G}, \Gamma, \mathbb{G}_T, q, g, \gamma, e \rangle$ if:

  $$\left| \Pr\left[\mathcal{A}(\gamma, g, g^a, g^b, g^{ab}) = 0 \right] - \Pr\left[\mathcal{A}(\gamma, g, g^a, g^b, Z) = 0 \right] \right| \geq \varepsilon$$

  or

  $$\left| \Pr\left[\mathcal{A}(g, \gamma, \gamma^a, \gamma^b, \gamma^{ab}) = 0 \right] - \Pr\left[\mathcal{A}(g, \gamma, \gamma^a, \gamma^b, Z') = 0 \right] \right| \geq \varepsilon \ ,$$

  where the probability is over the random choice of $a, b \in \mathbb{Z}_q$, $Z \in \mathbb{G}$ and $Z' \in \Gamma$, and the random bits of $\mathcal{A}$.

The SXDH assumption states that no polynomial-time algorithm can solve the SXDH problem for $\langle \mathbb{G}, \Gamma, \mathbb{G}_T, q, g, \gamma, e \rangle$ generated by $\mathcal{IG}$ on input λ with a non-negligible advantage.

In this thesis (in Chapter 7), we also use a slightly stronger variant of the CDH assumption which we call the *modified* CDH (mCDH).

- The modified Computational Diffie-Hellman (mCDH) assumption. Let $\mathcal{IG}$ be a polynomial-time-algorithm that takes as input the security parameter λ and outputs the tuple $\langle \mathbb{G}, \Gamma, \mathbb{G}_T, q, g, \gamma, e \rangle$, where $\mathbb{G}, \Gamma, \mathbb{G}_T$ are cyclic groups, q is the order of $\mathbb{G}$, g is a generator of $\mathbb{G}$, γ is a generator of Γ, and $e : \mathbb{G} \times \Gamma \to \mathbb{G}_T$.

  The mCDH problem is defined as follows: given $(g, g^a, g^b, \gamma, \gamma^b)$, where a, b are randomly chosen from $\mathbb{Z}_q$, compute g^{ab}. A polynomial-time adversary $\mathcal{A}$ has advantage ε in solving the mCDH problem in $\langle \mathbb{G}, \Gamma, \mathbb{G}_T, q, g, \gamma, e \rangle$ if:

  $$\Pr\left[\mathcal{A}(g, g^a, g^b, \gamma, \gamma^b) = g^{ab}\right] \geq \varepsilon \ ,$$

  where the probability is over the random choice of $a, b \in \mathbb{Z}_q$ and the random bits of $\mathcal{A}$.

  The mCDH assumption states that no polynomial-time algorithm can solve the mCDH problem for $\langle \mathbb{G}, \Gamma, \mathbb{G}_T, q, g, \gamma, e \rangle$ generated by $\mathcal{IG}$ on input λ with a non-negligible advantage.

The mCDH assumption is implied by the following assumption [35].

- The Bilinear Diffie-Hellman in Type 3 (BDH-3) assumption. Let $\mathcal{IG}$ be a polynomial-time-algorithm that takes as input the security parameter λ and outputs the same tuple as in the mCDH assumption.

  The Bilinear Diffie-Hellman in Type 3 (BDH-3) problem in $\langle \mathbb{G}, \Gamma, \mathbb{G}_T,\ q, g, \gamma, e \rangle$ is defined as follows: given (g, g^a, g^b, g^c) and $(\gamma, \gamma^b, \gamma^c)$, where a, b, c are randomly chosen from $\mathbb{Z}_q$, compute $e(g, \gamma)^{abc}$. A polynomial-time adversary $\mathcal{A}$ has advantage ε in solving the BDH-3 problem in $\langle \mathbb{G}, \Gamma, \mathbb{G}_T,\ q, g, \gamma, e \rangle$ if:

  $$\Pr\left[\mathcal{A}(g, g^a, g^b, g^c, \gamma, \gamma^b, \gamma^c) = e(g, \gamma)^{abc}\right] \geq \varepsilon \ ,$$

  where the probability is over the random choice of $a, b, c \in \mathbb{Z}_q$ and the random bits of $\mathcal{A}$.

  The BDH-3 assumption states that no polynomial-time algorithm can solve the BDH-3 problem for $\langle \mathbb{G}, \Gamma, \mathbb{G}_T, q, g, \gamma, e \rangle$ generated by $\mathcal{IG}$ on input λ with a non-negligible advantage.

Lemma 1. *BDH-3 assumption implies the* mCDH *assumption.*

Proof. We show how an algorithm $\mathcal{B}$ can break the BDH-3 assumption by running as a sub-routine the algorithm $\mathcal{A}$ that can break the mCDH assumption. To solve the BDH-3 problem, the algorithm $\mathcal{B}$ operates as follows. On input of the BDH-3

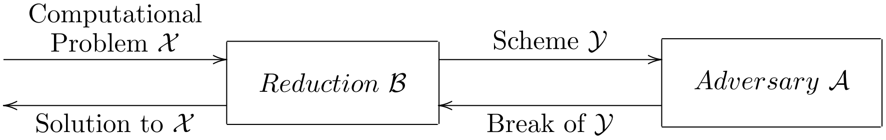
Computational Problem $\mathcal{X}$
Solution to $\mathcal{X}$
Reduction $\mathcal{B}$
Scheme $\mathcal{Y}$
Break of $\mathcal{Y}$
Adversary $\mathcal{A}$

Figure 2.1: *Proof by Reduction.*

instance (g, g^a, g^b, g^c) and $(\gamma, \gamma^b, \gamma^c)$, $\mathcal{B}$ passes an instance of the mCDH (g, g^a, g^b) and (γ, γ^b) to $\mathcal{A}$. $\mathcal{A}$ solves the mCDH problem by computing g^{ab} and sends it to $\mathcal{B}$. Finally, $\mathcal{B}$ solves the BDH-3 problem by computing $e(g^{ab}, \gamma^c)$ which is equal to $e(g, \gamma)^{abc}$. □

2.4 Standard Model

A cryptographic scheme is secure in the *standard model* if its security is proven *only* under complexity assumptions. As we explained above, an assumption states that a specific computational problems, e.g. DLP, cannot be solved by a polynomial-time algorithm. The strategy for proving the security is by reduction.

Proofs by reduction state that as long as the computational problem is hard to solve then a given cryptographic scheme is secure. A proof by reduction proceeds as follows: first we assume that a computational problem $\mathcal{X}$ is hard to solve. Then, we fix a polynomial-time algorithm $\mathcal{A}$ against the scheme $\mathcal{Y}$. We also fix a polynomial-time algorithm $\mathcal{B}$ trying to solve $\mathcal{X}$. If $\mathcal{A}$ breaks the scheme $\mathcal{Y}$ with a non-negligible probability, then $\mathcal{B}$ solves the hard computational problem $\mathcal{X}$ with a non-negligible probability (see Figure 2.1). However, since we assumed that the computation problem $\mathcal{X}$ is hard, we get a *contradiction.* At this point, we prove the security of the scheme.

2.5 Idealized Security Models

Proving the security of the scheme in the standard model is usually difficult. Sometimes for a given construction it is hard to construct a reduction algorithm which reduces the problem of breaking the scheme to the problem of breaking the standard complexity assumption. In addition, practice has shown that most of the schemes with a security proof in the standard model are not practical. Indeed, a lot of cryptographic schemes with the security proof in the standard model are designed to have an efficient reduction algorithm, which renders the scheme inefficient [59, Chapter 11]. As an alternative to the standard model, a number of practical schemes in cryptography are proven secure in idealized models, such as the *random oracle model* [18] and the *generic group model* [93].

2.5.1 Random Oracle Model

The security proof in the random oracle model assumes that hash functions are indistinguishable from random functions. To understand how this model works, we first explain hash functions, then we explain the random oracle model.

Hash Functions

Hash functions play an important role in cryptography. One application of cryptographic hash functions is to ensure message integrity. For instance, to ensure the integrity of x the value $y_x = \mathsf{H}(x)$ is computed. If x is changed to x', then one hashes $y_{x'} = \mathsf{H}(x')$ and checks whether $y_x = y_{x'}$. If they are not equal, then x has been changed, otherwise not. Another application of hash functions is for password verification. Passwords in computers are not stored in cleartext, but to hide them from unintended users their hash value is stored. When a user wants to access a system, the password of the user is first hashed and the hash is compared to the stored hash value. If the hashes are the same then the system assumes that the user knows the password and is granted access to the computer, otherwise access is denied. For a formal treatment of hash functions refer to [86].

The Model

In the random oracle model, introduced by Bellare and Rogaway [18], the hash function is assumed to be indistinguishable from a random function. This model is widely used when proving security, but does not reflect the real world since there is no real function which can be implemented as a true random oracle. Nevertheless, a proof in the random oracle model is better than no security proof at all and it gives a confidence that the scheme is secure.

In this model an attacker has only oracle access to H and can evaluate it only by querying the oracle. The oracle is simulated as follows. A simulator answers the adversary's queries and maintains a list for all queried inputs x_i and their corresponding random values y_i. For each query x_i the simulator generates a fixed length random output y_i and returns it to the adversary. If the query x_i has been previously asked by the adversary, then the random oracle will output the same value y_i as it did before. This means that we can view the random oracle as a "box" that implements the hash function H and returns $\mathsf{H}(x)$ when evaluated on the input x. Note that internals of the box are not known to the adversary.

The strategy when proving the security in the random oracle model is by reducing the security of a system to the solvability of some computationally hard problems (cryptographic assumptions). For instance, in the same way as we have in the standard model, if there is an adversary that can break the cryptographic scheme with the random oracle, then there is an algorithm (i.e. the reduction) that solves the hard problem. Here the reduction contradicts the complexity assumption, hence the cryptographic scheme is secure.

As we mentioned above, in practice the random oracle might be instantiated with a cryptographic hash function. However, Canetti, Goldreich and Halevi [31] show

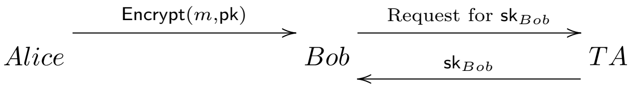

Figure 2.2: *Identity-Based Encryption.*

that there are encryption and signature schemes that are secure in the random oracle model but, but are insecure for any instantiation of the random oracle as a hash function.

2.5.2 Generic Group Model

In the generic group model, introduced by Shoup [93], group elements are encoded as unique random strings, in such a way that the adversary cannot test any property other than equality (i.e. the adversary is unable to compute group operations by himself). As in the random oracle model, this model does not reflect the real world since in practice an adversary has access to an efficient encoding of the group and is able to compute group operations by himself. While it is preferred to prove the security of a scheme by reducing the problem of breaking a scheme to a well studied computational problem, a proof in the generic model gives confidence in the security of the scheme.

In the generic model, the adversary has access to the oracles that compute group operations. An adversary sends two random encodings of the group elements and the group operation to the oracle. The oracle, on input of the two random encodings of the group elements, outputs a random encoding of a third element. If the scheme uses groups with pairing, then the adversary has access to an additional oracle for computing pairing operations. All queries made by the adversary are stored in a list, and if the adversary makes the same query twice, then the oracle returns the same answer on each query.

In a similar way as Canetti, Goldreich and Halevi [31] show the problem with the random oracle model, Dent [41] shows that there exist cryptographic schemes which are secure in the generic model but which are insecure in practice. Therefore care must be taken when implementing cryptographic schemes with a security proof in the generic group model.

2.6 Identity-Based Encryption

This scheme presented in Chapter 3 uses the Identity-Based Encryption (IBE) as a basic building block. That is why in this section we explain IBE and its security definition.

In traditional public-key encryption (PKE), the authenticity of cryptographic keys is important. The party who encrypts the data needs to be assured that the public key belongs to the right user who is also in possession of the corresponding private key. Therefore, there is a need for a Public Key Infrastructure (PKI). In PKI, the Certificate Authority (CA) generates a digital certificate, which contains a digital signature, to assure that the public key belongs to the right user. Whenever a user wants to use a public key, the user has to obtain the digital certificate and verify the signature. In practice PKI technology suffers from drawbacks such as certificate verification, revocation, distribution, storage, etc [52].

The concept of IBE was proposed by Shamir [89] in 1984, however, IBE has become practical only after 2001 when three IBE schemes were proposed by: Boneh and Franklin [27], Cocks [38] and Sakai, Ohgishi and Kasahara [87]. Unlike in PKE, an IBE does not require a digital certificate to certify the encryption key (public key) because the public key of any user can be an arbitrary string (e.g. email address, IP address). Key escrow is an inherent property in IBE systems - the Trusted Authority (TA), also referred to as the Key Generation Center (KGC), holds the master key from which it generates each user's private key. As mentioned in [36], the key escrow problem of IBE can be mitigated by applying some standard techniques (such as secret sharing) to the underlying scheme, hence, we skip any further discussion. IBE is a suitable technique to be used to exchange emails securely. For example, in Figure 2.2, when Alice wants to send an encrypted email to Bob, Alice can encrypt the email using a public key derived from Bob's identity and send the email via an insecure channel. Bob can authenticate himself to the TA to get his private key. Unlike in PKE where the private key and the public key have to be created simultaneously, in IBE the private key can be generated after the corresponding public key is generated.

Let $\mathcal{M}$ be the message space and $\mathcal{C}$ the ciphertext space. $\mathcal{M}$ is the set of all messages that can be chosen during the encryption phase and $\mathcal{C}$ is the set of all ciphertext that can be generated by the encryption algorithm. The IBE scheme consists of four algorithms: Setup, KeyGen, Encrypt and Decrypt, defined as follows:

- $\mathsf{Setup}(\lambda)$. This algorithm takes as input a security parameter λ and outputs public parameters $\mathcal{PP}$ and the master key msk. Public parameters $\mathcal{PP}$ contain descriptions about message space $\mathcal{M}$ and ciphertext space $\mathcal{C}$.

- $\mathsf{KeyGen}(\mathsf{msk}, id)$. This algorithm takes as input the master key msk and an identity $id \in \{0,1\}^*$. The algorithm returns the secret key sk_{id}.

- $\mathsf{Encrypt}(m, \mathsf{pk}_{id})$. This algorithm is a probabilistic algorithm which takes as input a plaintext $m \in \mathcal{M}$ and a public key pk_{id} generated from an identity $id \in \{0,1\}^*$. The algorithm outputs a scrambled form of the plaintext, called ciphertext $c \in \mathcal{C}$.

- $\mathsf{Decrypt}(c, \mathsf{sk}_{id})$. The decryption algorithm Decrypt is a deterministic algorithm which takes as input a ciphertext $c \in \mathcal{C}$ and the private key sk_{id} and it outputs either the plaintext m or an error symbol $\perp$.

Correctness. We say that IBE is *correct* if for all security parameters $\lambda \in \mathbb{N}$, for all master secret keys msk and public parameters $\mathcal{PP}$ produced by Setup, for all private keys sk_{id} produced by KeyGen, for all public keys pk_{id} generated from the identity *id*, for all ciphertexts c produced by $\mathsf{Encrypt}$, for every plaintext $m \in \mathcal{M}$, we should have:

$$\Pr\left[\begin{array}{c}(\mathsf{msk}, \mathcal{PP}) \leftarrow \mathsf{Setup}(\lambda), \mathsf{sk}_{id} \leftarrow \mathsf{KeyGen}(\mathsf{msk}, id) \\ c \leftarrow \mathsf{Encrypt}(m, id) : m \leftarrow \mathsf{Decrypt}(c, \mathsf{sk}_{id})\end{array}\right] = 1 \ .$$

2.6.1 Security Definitions

We describe a security property for an IBE scheme which is indistinguishability under adaptive chosen-ciphertext attacks (IND-ID-CCA). This property guarantees that ciphertexts are indistinguishable, and that it is infeasible for an adversary to learn any information about the plaintext from the ciphertext. The security against IND-ID-CCA is modelled by an IBE security-game. The security-game is carried out between a challenger and an adversary $\mathcal{A}$, where the challenger simulates the protocol execution and answers the queries from $\mathcal{A}$.

Definition. *An* IBE *scheme is said to be indistinguishable under adaptive chosen-ciphertext attacks* (IND-ID-CCA) *if no polynomial-time adversary has a non-negligible advantage against the challenger in the* IBE *security-game defined as follows:*

- *Setup: The challenger takes a security parameter λ and runs the* Setup *algorithm to generate public parameters $\mathcal{PP}$ and the master key* msk*.*

- *Phase 1: The adversary $\mathcal{A}$ takes $\mathcal{PP}$ as input and is allowed to issue two type of queries:*

  - KeyGen *query with any identifier id: The challenger returns the private key sk_{id} corresponding to id.*
  - Decrypt *query with any ciphertext c and any identifier id: The challenger runs* KeyGen *to generate the private key sk_{id} corresponding to id, and then returns the value of* $\mathsf{Decrypt}(c, \mathsf{sk}_{id})$*.*

  Once $\mathcal{A}$ decides that Phase 1 is over, it outputs two plaintexts m_0, m_1, such that $|m_0| = |m_1|$, and an identifier id^* on which it wishes to be challenged. The only constraint is that id^* has not been the input to any previous KeyGen *query.*

- *Challenge: The challenger picks a random bit $b \in \{0, 1\}$ and returns $c^* =$* $\mathsf{Encrypt}(m_b, id^*)$ *as the challenge to $\mathcal{A}$.*

- *Phase 2: $\mathcal{A}$ is allowed to continue issuing the same types of queries as in Phase 1. However, it is not allowed to ask a* KeyGen *query with the input id^* and a* Decrypt *query with the input (c^*, id^*).*

- *Guess: $\mathcal{A}$ outputs a guess $b' \in \{0, 1\}$, and wins the security-game if $b' = b$.*

The advantage of $\mathcal{A}$ in winning security-game is defined as:

$$\mathsf{ADV}^{\mathsf{IND-ID-CCA}}_{\mathcal{A},\mathsf{IBE}}(\lambda) \overset{def}{=} \left| \Pr[b' = b] - \frac{1}{2} \right| \ ,$$

where the probability is taken over the random values chosen by $\mathcal{A}$ and the challenger.

Definition. *An* IBE *scheme is said to be semantically secure against an adaptive chosen plaintext attack* ($\mathsf{IND\text{-}ID\text{-}CPA}$) *if any polynomial time* $\mathsf{IND\text{-}ID\text{-}CCA}$ *adversary's advantage is negligible when it makes no* $\mathsf{Decrypt}$ *query in the* IBE *security-game.*

Apart from semantic security, we can also define the *one-wayness* for IBE. This property guarantees that it is hard for an adversary to invert the ciphertext and to learn the plaintext if the adversary does not hold the receiver's private key. Formally, we have the following definition:

Definition. *An* IBE *scheme is said to be one-way if any polynomial time adversary's advantage is negligible in the following security-game:*

- *Setup: The challenger takes a security parameter* λ *and runs the* Setup *algorithm to generate public parameters* $\mathcal{PP}$ *and the master key* msk.

- KeyGen *query: The adversary* $\mathcal{A}$ *takes* $\mathcal{PP}$ *as input and is allowed to issue any number of* KeyGen *query with any identifier id: The challenger returns the private key* sk_{id} *corresponding to id. Once* $\mathcal{A}$ *decides that this phase is over, it outputs an identifier* id^* *on which it wishes to be challenged. The only constraint is that* id^* *has not been the input to any* KeyGen *query.*

- *Challenge: The challenger picks a random message* m *and returns the challenge* $c^* = \mathsf{Encrypt}(m, id^*)$ *to* $\mathcal{A}$.

- *Guess:* $\mathcal{A}$ *outputs a guess* m'*, and wins the game if* $m' = m$.

The advantage of $\mathcal{A}$ *in winning the security-game is defined as* $\Pr[m' = m]$*, where the probability is taken over the random values chosen by* $\mathcal{A}$ *and the challenger.*

2.6.2 Boneh-Franklin IBE

We briefly review the Boneh-Franklin scheme, which, compared to the original scheme [27], is slightly modified in the definition of the message domain. Nonetheless, we still call it the Boneh-Franklin scheme. Indeed, the reason for modifying the Boneh-Fraklin scheme is to make it compatible with the message domain of the proposed PRE. Note that the same modifications are also made in [51].

- $\mathsf{Setup}(\lambda)$: Run by the KGC, given a security parameter λ, the algorithm generates two cyclic groups $\mathbb{G}$ and $\mathbb{G}_T$ of prime order p, a generator g of $\mathbb{G}$, a bilinear map $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$, a master secret key $\mathsf{msk} = \alpha \in \mathbb{Z}_p^*$, and a hash function $\mathsf{H}_1 : \{0,1\}^* \rightarrow \mathbb{G}$ [1]. Public parameters are $\mathcal{PP} = (\mathbb{G}, \mathbb{G}_T, p, g, \mathsf{H}_1, e, \mathsf{pk})$, where $\mathsf{pk} = g^\alpha$ is the public key of the KGC.

  In the original Boneh-Franklin scheme, the plaintext space is $\{0,1\}^n$ where n is an integer and there is an additional hash function $\mathsf{H}_2 : \mathbb{G}_T \rightarrow \{0,1\}^n$.

[1] A formal definition of hash functions is given in Section 2.5.1

- $\mathsf{KeyGen}(\mathsf{msk}, id)$: Run by the KGC, given a master secret key msk and an identifier id, the algorithm outputs the private key $\mathsf{sk}_{id} = \mathsf{pk}_{id}^{\alpha}$, where $\mathsf{pk}_{id} = \mathsf{H}_1(id)$.
- $\mathsf{Encrypt}(m, id)$: Run by the message sender, given a message $m \in \mathbb{G}_T$ and an identifier $id \in \{0,1\}^*$ the algorithm outputs the ciphertext $c = (c_1, c_2)$ where $c_1 = g^r$, $c_2 = m \cdot e(\mathsf{pk}_{id}, pk)^r$, and $r \in \mathbb{Z}_p^*$.

  In the original Boneh-Franklin scheme, $c_2 = m \oplus \mathsf{H}_2(e(\mathsf{pk}_{id}, pk)^r)$.
- $\mathsf{Decrypt}(c, \mathsf{sk}_{id})$: Run by the receiver with identifier id, given a ciphertext $c = (c_1, c_2)$ and sk_{id}, the algorithm outputs the message $m = c_2 / e(\mathsf{sk}_{id}, c_1)$.

  In the original Boneh-Franklin scheme, $m = c_2 \oplus \mathsf{H}_2(e(\mathsf{sk}_{id}, c_1))$.

Implied by the security proof of the scheme IBP1 in [51], the Boneh-Franklin scheme is semantically secure against an IND-ID-CCA attack based on the DBDH assumption.

2.7 Conclusion

In this chapter we provide all the background information that is needed to understand the rest of the thesis. In particular, we briefly described groups, fields, elliptic curves and bilinear maps. We also reviewed complexity assumptions that we will use to prove the security of our schemes. Finally, we explained security models and identity-based encryption.

Chapter 3

Fine-Grained Access Policies for Proxy Re-Encryption

As described in Chapter 1, existing proxy re-encryption schemes provide limited support for the delegator to express a fine-grained access control policy. In this chapter we construct a mechanism that enables the delegator to provide a fine-grained access control policy by allowing the delegator to categorize his messages into different types. We refer to this mechanism as a Type-and-Identity-based Proxy Re-Encryption.

We start the chapter with an introduction of related work in proxy re-encryption. We continue by formally introducing the concept of Type-and-Identity-based Proxy Re-Encryption along with its security definition. We then present our construction and its security proof. The contents of this chapter is adapted from two refereed papers [5, 7].

3.1 Introduction

Proxy re-encryption is a cryptographic method developed to delegate the decryption rights from one party (the delegator) to another (the delegatee). In a proxy re-encryption (PRE) scheme, the delegator assigns a key to a proxy to re-encrypt all messages encrypted with his public key such that the re-encrypted ciphertexts can be decrypted with the delegatee's private key. The proxy is a semi-trusted entity i.e. it is trusted to perform only the ciphertext re-encryption, without knowing the private keys of the delegator and the delegatee, and without having access to the plaintext.

Since Mambo and Okamoto [68] first proposed the concept of PRE, a number of PRE schemes have been proposed [21, 55, 56, 51, 69]. In the context of public key encryption, PRE is used to forward encrypted messages [103] without revealing the plaintext. For example, in Figure 3.1, Alice (delegator) is the president of a company

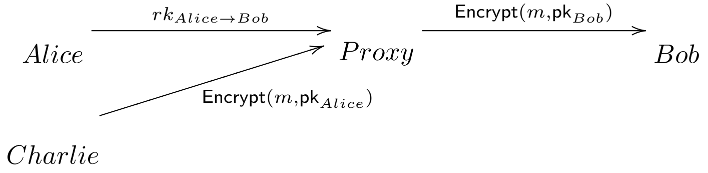

Figure 3.1: *Proxy Re-encryption.*

who wants to allow her secretary, Bob (delegatee), to read her emails when she is on vacation (note that Alice cannot give her private key to Bob). Using PRE, Alice can compute a re-encryption key to allow the proxy to transform a ciphertext for Alice, generated by Charlie, into a ciphertext for Bob, thus, Bob can decrypt the re-encrypted data using his private key. In practice the role of the proxy can be played by a commercial enterprise that has enough computation power to perform re-encryption services for a large number of users.

Proxy re-encryption has many other promising applications including law enforcement [55]. With the increasing privacy concerns over personal data, proxy re-encryption, in particular IBE proxy re-encryption schemes (due to their benefits [27]), will find more and more applications. For example, in the healthcare domain, many regulations, such as HIPPA [99], require that the patient is the owner of his personal health record and should control the disclosure policy for his Personal Health Record (PHR). The PHR contains all kinds of health-related information about the patient [97] and a proper enforcement of access control is one of the central themes in deploying a secure PHR system.

An observation on the existing PRE schemes is that the proxy is able to re-encrypt all ciphertexts from the delegator to the delegatee. As a result, it is difficult for the delegator to implement any further fine-grained cryptographically enforced access control (AC) policy for multiple delegation services. Suppose the delegator wants the delegatee to be able to recover different subsets of his messages. In this case, the delegator can only trust the proxy to enforce his policies by re-encrypting the legitimate ciphertexts. In practice, this trust assumption might be unrealistic (for example, the proxy can be corrupted). To solve this problem, an alternative solution could be that the delegator chooses a different key pair for each delegatee, which is also unrealistic.

High Level Contributions. In this chapter we propose a type-and-identity-based proxy re-encryption scheme (TID-PRE) which allows the delegator to define a fine-grained AC policy. The delegator can categorize messages into different types and delegate the decryption rights of each type to the delegatee through a proxy. Data categorization is needed since different data may have different levels of privacy requirements. One benefit of our scheme is that it does not suffer from key management problem since the delegator uses only one key pair to compute re-encryption keys. The delegator computes re-encryption keys and forwards them to the proxy which will re-encrypt the data without decrypting them such that the intended delegatee can decrypt the re-encrypted data using his private key. The other benefit is that there is no further trust assumption on the proxy compared to existing proxy re-encryption schemes. However, the proposed scheme only works for the ciphertexts generated by the delegator.

3.1.1 Related work

Mambo and Okamoto [68] were the first to propose the concept of delegation of decryption rights in the context of speeding up decryption operations. Blaze *et al.* [21] propose the concept of atomic proxy cryptography which is the current concept of proxy re-encryption. In a proxy re-encryption scheme, the proxy can transform ciphertexts encrypted with the delegator's public key into ciphertexts that can be decrypted with the delegatee's private key. Blaze *et al.* base their proxy re-encryption scheme on the ElGamal scheme [45]. One property of this scheme is that, with the same proxy key (or the same re-encryption key), the proxy can transform the ciphertexts not only from the delegator to the delegatee but also from the delegatee to the delegator. Schemes that have this property are known as "bi-directional" proxy re-encryption scheme. Bi-directionality might be a problem in some applications, but it might also be a desirable property in other applications. Jacobsson [56] addresses this "problem" using a quorum controlled asymmetric proxy re-encryption where the proxy is implemented with multiple servers and each of them performs partial re-encryption.

Dodis and Ivan [55] propose a generic construction method for proxy re-encryption schemes and also provide a number of example schemes. Their constructions are based on the concept of secret splitting, which means that the delegator splits his private key into two parts and sends them to the proxy and the delegatee separately. During the re-encryption process the proxy performs partial decryption of the encrypted message using the first part of the delegator's private key, and the delegatee can recover the message by performing partial decryption using the second part of the delegator's private key. One disadvantage of this method is that it is not collusion-resistant, i.e. the proxy and the delegatee together can recover the delegator's private key. Another disadvantage of this scheme is that the delegatee's public/private key pair can only be used for dealing with the delegator's messages. If this key pair is used by the delegatee for other encryption services, then the delegator can always decrypt the ciphertexts.

Ateniese *et al.* [13] propose several proxy re-encryption schemes based on the ElGamal scheme. In their schemes, the delegator does not have to interact and share his private key with the delegatee. The delegator stores two secret keys, a master secret key and a "weak" secret key. The ciphertext can be fully decrypted using either of the two distinct keys. Their scheme is collusion-resistant, since only the "weak" secret key is exposed if the delegatee and the proxy collude but the master key remains safe. The disadvantage of this scheme is that the delegator has to perform two levels of encryptions, the first level encryption encrypts messages that can be decrypted by the delegator, and the second level encryption encrypts messages that can be decrypted by the delegator and his delegatee. In addition, Ateniese *et al.* also discuss a number of properties for proxy re-encryption schemes in [13].

Recently, two Identity-Based Encryption (IBE) proxy re-encryption schemes were proposed by Matsuo [69] and Green and Ateniese [51], respectively. The Matsuo scheme assumes that the delegator and the delegatee belong to the same Key Generation Center (KGC) and use the Boneh-Boyen encryption scheme [22]. The Green-Ateniese scheme assumes that the delegator and the delegatee can belong to different KGCs, and the delegatee needs to obtain the public parameter of the delegator's KGC.

3.2 Type-and-Identity-based Proxy Re-encryption

In this section we formalize the concept of TID-PRE. TID-PRE involves four entities: the trusted authority (TA) or the Key Generation Center (KGC), the delegator, the proxy and the delegatee. Unlike in traditional PRE, in TID-PRE the delegator in the encryption phase categorizes his messages into different categories or types (see Figure 3.2). This enables him to delegate the decryption rights for each type to a specific delegatee. The delegator is also the only encryptor since ciphertexts are generated based on the delegator's secret key.

A TID-PRE scheme consists of the following six algorithms (Setup, KeyGen, Pkeygen, Encrypt, Preenc, Decrypt):

- $\mathsf{Setup}(\lambda)$: Run by the trusted authority (TA) or Key Generation Center (KGC) and takes as input a security parameter λ and outputs the master secret key $\mathsf{msk} \in \mathcal{K}$ and public parameters $\mathcal{PP}$. The latter contains the master public key $\mathsf{pk} \in \mathcal{K}$, which is an implicit input for all other algorithms, and descriptions about message space $\mathcal{M}$ and ciphertext space $\mathcal{C}$.

- $\mathsf{KeyGen}(\mathsf{msk}, id)$: Run by the TA, the algorithm takes as input the master secret key $\mathsf{msk} \in \mathcal{K}$ and an identity id, and it outputs a user private key sk_{id}.

- $\mathsf{Pkeygen}(id_i, id_j, t, \mathsf{sk}_{id_i})$: Run by the delegator, the algorithm takes the delegator's identifier id_i, the delegatee's identifier id_j, the type t, and the delegator's private key $\mathsf{sk}_{id_i} \in \mathcal{K}$ as input and outputs the proxy or re-encryption key $rk_{id_i \to id_j} \in \mathcal{K}$.

- $\mathsf{Encrypt}(m, t, id_i)$: Run by the delegator the algorithm takes as input a message $m \in \mathcal{M}$, a type t, and an identifier id, and it outputs the ciphertext $c_i \in \mathcal{C}$.

- $\mathsf{Preenc}(c_i, rk_{id_i \to id_j})$: Run by the proxy, the algorithm takes a ciphertext c_i and the re-encryption key $rk_{id_i \to id_j} \in \mathcal{K}$ as input, and outputs a new ciphertext $c_j \in \mathcal{C}$.

- $\mathsf{Decrypt}(c, \mathsf{sk}_{id})$: Run either by the delegator or the delegatee, the algorithm takes as input the ciphertext $c \in \mathcal{C}$ and the private-key $\mathsf{sk}_{id} \in \mathcal{K}$, and outputs either the message $m \in \mathcal{M}$ or $\perp$. The ciphertext $c \in \mathcal{C}$ can be either equal to $c_i \in \mathcal{C}$ (produced by $\mathsf{Encrypt}$) or equal to $c_j \in \mathcal{C}$ (produced by Preenc). When $c = c_i$, the secret key sk_{id} is equal to sk_{id_i} (delegator's private key) and when $c = c_j$, the secret key is equal to sk_{id_j} (delegatee's secret key).

Correctness. We say that $\mathsf{TID} - \mathsf{PRE}$ is *correct* if for all security parameters $\lambda \in \mathbb{N}$, for all master secret keys msk and public parameters $\mathcal{PP}$ produced by Setup, for all delegators' sk_{id_i} and delegatees' sk_{id_j} private keys produced by KeyGen, for all re-encryption keys $rk_{id_i \to id_j}$ produced by $\mathsf{Pkeygen}$, for all ciphertexts c_i produced by $\mathsf{Encrypt}$, for all re-encrypted ciphertexts c_j produced by Preenc, we should have:

$$\Pr\left[\begin{array}{c} (\mathsf{msk}, \mathcal{PP}) \leftarrow \mathsf{Setup}(\lambda),\ \mathsf{sk}_{id_i} \leftarrow \mathsf{KeyGen}(\mathsf{msk}, id_i), \\ \mathsf{sk}_{id_j} \leftarrow \mathsf{KeyGen}(\mathsf{msk}, id_j),\ rk_{id_i \to id_j} \leftarrow \mathsf{Pkeygen}(id_i, id_j, t, \mathsf{sk}_{id_i}), \\ c_i \leftarrow \mathsf{Encrypt}(m, t, id_i),\ c_j \leftarrow \mathsf{Preenc}(c_i, rk_{id_i \to id_j}) : \\ m \leftarrow \mathsf{Decrypt}(c_i, \mathsf{sk}_{id_i}) \vee m \leftarrow \mathsf{Decrypt}(c_j, \mathsf{sk}_{id_j}) \end{array}\right] = 1\ .$$

3.2.1 Security Definitions

We assume that the proxy is honest-but-curious in the following sense: it will honestly convert the delegator's ciphertexts using the proxy key; however, it might try to actively obtain some information about the plaintexts for the delegator and the delegatee. The delegatee may be curious in a sense that it may try to obtain some information about the plaintexts corresponding to the delegator's ciphertexts which have not been re-encrypted by the proxy.

As a standard practice, we describe a security-game definition for modelling semantic security against an adaptive chosen plaintext attack (IND-ID-PR-CPA). The security-game is carried out between a challenger and an adversary, where the challenger simulates the protocol execution and answers the queries from the adversary. Note that the allowed queries for the adversary reflect the adversary's capability in practice.

Definition. *A* TID-PRE *scheme is said to be secure against an adaptive chosen-plaintext attack* (IND-ID-PR-CPA) *if any polynomial-time adversary $\mathcal{A}$ has only a negligible advantage in the* TID-PRE *security-game defined as follows:*

- *Setup: The challenger takes a security parameter λ as input, runs the* Setup *algorithm to generate public parameters $\mathcal{PP}$ and the master key* msk.

- *Phase 1: The adversary takes $\mathcal{PP}$ as input and has access to the following oracles:*

  - KeyGen *oracle - The adversary queries the oracle with an identifier id. The challenger returns the private key* sk_{id} *corresponding to id.*
  - Pkeygen *oracle - The adversary queries the oracle with (id, id', t). The challenger returns the re-encryption key $rk_{id \to id'}$ for the type t.*

  - Preenc *oracle - The adversary queries the oracle with* (m, t, id, id'). *The challenger first computes* $c = \mathsf{Encrypt}(m, t, id)$ *and then returns a new ciphertext* c' *which is obtained by applying the re-encryption key* $rk_{id \to id'}$ *to* c*, where* $rk_{id \to id'}$ *is issued for type* t.

  Once the adversary decides that Phase 1 is over, it outputs two equal length plaintexts m_0, m_1*, a type* t^**, and an identifier* id^**. At the end of Phase 1, there are three constraints:*

  - id^* *has not been the input to any* KeyGen *query.*
  - *For any* id'*, if* (id^*, id', t^*) *has been the input to a* $\mathsf{Pkeygen}$ *query then* id' *should not have been the input to any* KeyGen *query.*
  - *If there is a* Preenc *query with* (m, t, id, id')*, then* (id, id', t) *should not have been queried to* $\mathsf{Pkeygen}$.

- *Challenge: The challenger picks a random bit* $b \in \{0, 1\}$ *and returns* $c^* = \mathsf{Encrypt}(m_b, t^*, id^*)$ *as the challenge to the adversary.*

- *Phase 2: The adversary is allowed to continue issuing the same types of queries as in Phase 1. At the end of Phase 2, there are the same constraints as at the end of Phase 1.*

- *Guess: the adversary outputs a guess* $b' \in \{0, 1\}$*, and wins the security-game if* $b' = b$.

The advantage of $\mathcal{A}$ *in winning security-game is defined as:*

$$\mathsf{ADV}^{\mathsf{IND-ID-PR-CPA}}_{\mathcal{A},\mathsf{TID-PRE}}(\lambda) \overset{def}{=} \left| \Pr[b' = b] - \frac{1}{2} \right| ,$$

where the probability is over the random values chosen by $\mathcal{A}$ *and the challenger.*

Compared with the chosen-plaintext security formalizations in [51, 69], in our case the categorization of messages is taken into account. The Preenc query reflects the fact that a curious delegatee has access to the delegator's plaintexts.

3.3 Construction of TID-PRE

In this section we propose a construction for TID-PRE. As a building block, the proposed construction uses the modified Boneh-Franklin IBE scheme presented in Section 2.6. Let $(\mathsf{Setup}_1, \mathsf{KeyGen}_1, \mathsf{Encrypt}_1, \mathsf{Decrypt}_1)$ be the algorithms of the modified Boneh-Franklin IBE scheme. The TID-PRE scheme consists of six algorithms (Setup, KeyGen, Pkeygen, Encrypt, Preenc, Decrypt) defined as follows:

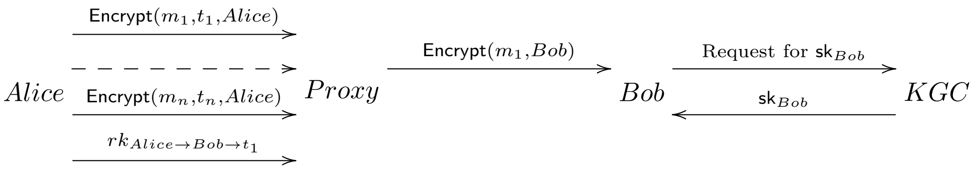
Encrypt(m_1,t_1,$Alice$)
Encrypt(m_n,t_n,$Alice$)
$rk_{Alice\to Bob\to t_1}$
$Alice$
$Proxy$
Encrypt(m_1,Bob)
Bob
Request for sk_{Bob}
sk_{Bob}
KGC

Figure 3.2: *A Type-and-Identity-Based Proxy Re-Encryption.*

- $\mathsf{Setup}(\lambda)$: Given a security parameter λ, the algorithm generates two cyclic groups $\mathbb{G}$ and $\mathbb{G}_T$ of prime order p, a generator g of $\mathbb{G}$, a bilinear map $e : \mathbb{G}\times\mathbb{G} \to \mathbb{G}_T$, a master secret key $\mathsf{msk} = \alpha \in \mathbb{Z}_p$, hash functions $\mathsf{H}_1 : \{0,1\}^* \to \mathbb{G}$ and $\mathsf{H}_2 : \{0,1\}^* \to \mathbb{Z}_p$. Public parameters are $\mathcal{PP} = (\mathbb{G}, \mathbb{G}_T, p, g, \mathsf{H}_1, \mathsf{H}_2, e, \mathsf{pk})$, where $\mathsf{pk} = g^\alpha$ is the public key of the KGC.

- $\mathsf{KeyGen}(\mathsf{msk}, id)$: This algorithm is the same as KeyGen_1.

- $\mathsf{Pkeygen}(id_i, id_j, t, \mathsf{sk}_{id_i})$: The algorithm takes as input the delegator's identifier id_i, the delegatee's identifier id_j, the type t, and the delegator's private key sk_{id_i} as input. The algorithm picks $X \in_R \mathbb{G}_T$ and outputs the re-encryption key:

$$rk_{id_i\to id_j} = (t, \mathsf{sk}_{id_i}^{-\mathsf{H}_2(\mathsf{sk}_{id_i}||t)} \cdot \mathsf{H}_1(X), \mathsf{Encrypt}_1(X, id_j)) \ .$$

- $\mathsf{Encrypt}(m, t, id_i)$: Given a message m, a type t, and an identifier id_i, the algorithm picks $r \in_R \mathbb{Z}_p^*$ and outputs the ciphertext $c_i = (c_{i1}, c_{i2}, c_{i3})$ where:

$$c_{i1} = g^r,\ \ c_{i2} = m \cdot e(\mathsf{pk}_{id_i}, \mathsf{pk})^{r\cdot\mathsf{H}_2(\mathsf{sk}_{id_i}||t)},\ \ c_{i3} = t \ .$$

  Note that apart from the delegator, another party cannot run $\mathsf{Encrypt}$ since only the delegator knows sk_{id_i}.

- $\mathsf{Preenc}(c_i, rk_{id_i\to id_j})$: Run by the proxy, this algorithm takes a ciphertext $c_i = (c_{i1}, c_{i2}, c_{i3})$ and a re-encryption key $rk_{id_i\to id_j}$ where $t = c_{i3}$, and outputs a new ciphertext $c_j = (c_{j1}, c_{j2}, c_{j3})$:

$$\begin{aligned} c_{j1} &= c_{i1}\ , \\ c_{j2} &= c_{i2} \cdot e(c_{i1}, \mathsf{sk}_{id_i}^{-\mathsf{H}_2(\mathsf{sk}_{id_i}||c_{i3})} \cdot \mathsf{H}_1(X)) \\ &= m \cdot e(g^\alpha, \mathsf{pk}_{id_i}^{r\mathsf{H}_2(\mathsf{sk}_{id_i}||t)}) \cdot e(g^r, \mathsf{sk}_{id_i}^{-\mathsf{H}_2(\mathsf{sk}_{id_i}||t)} \cdot \mathsf{H}_1(X)) \\ &= m \cdot e(g^r, \mathsf{H}_1(X))\ , \\ c_{j3} &= \mathsf{Encrypt}_1(X, id_j)\ , \end{aligned}$$

- $\mathsf{Decrypt}(c, \mathsf{sk}_{id})$: This algorithm operates in the following two ways:

1. If $c = c_i$ and $\mathsf{sk}_{id} = \mathsf{sk}_{id_i}$, then parse c_i as (c_{i1}, c_{i2}, c_{i3}) and output:
$$m = \frac{c_{i2}}{e(\mathsf{sk}_{id_i}, c_{i1})^{\mathsf{H}_2(\mathsf{sk}_{id_i}||c_{i3})}} .$$
2. If $c = c_j$ (i.e. c_j is a re-encrypted ciphertext) and $\mathsf{sk}_{id} = \mathsf{sk}_{id_j}$, then parse c_j as (c_{j1}, c_{j2}, c_{j3}) and output:
$$m = \frac{c_{j2}}{e(c_{j1}, \mathsf{H}_1(\mathsf{Decrypt}_1(c_{j3}, \mathsf{sk}_{id_j})))} .$$

Correctness. We first show the correctness for Decrypt when it gets as input the cipheretxt c_i produced by Encrypt. Let $c_i = (c_{i1} = g^r, c_{i2} = m \cdot e(\mathsf{pk}_{id_i}, \mathsf{pk})^{r \cdot \mathsf{H}_2(\mathsf{sk}_{id_i}||t)}$, $c_{i3} = t)$ be an encryption of m under the public key pk_{id_i} and type t and let $sk_{id_i} = \mathsf{pk}^{\alpha}_{id_i}$ be a private key for the identity id_i. The Decrypt outputs the message m as follows:

$$\begin{aligned} & \frac{c_{i2}}{e(\mathsf{sk}_{id_i}, c_{i1})^{\mathsf{H}_2(\mathsf{sk}_{id_i}||c_{i3})}} \\ = & \frac{m \cdot e(\mathsf{pk}_{id_i}, \mathsf{pk})^{r \cdot \mathsf{H}_2(\mathsf{sk}_{id_i}||t)}}{e(\mathsf{pk}^{\alpha}_{id_i}, g^r)^{\mathsf{H}_2(\mathsf{sk}_{id_i}||c_{i3})}} \\ = & m . \end{aligned}$$

Next we show the correctness for Decrypt when it gets as input the ciphertext c_j produced by Preenc. Let $c_j = (c_{j1} = g^r, c_{j2} = m \cdot e(g^r, \mathsf{H}_1(X)), c_{j3} = \mathsf{Encrypt}_1(X, id_j))$ be the re-encrypted ciphertext under the public key pk_{id_j} and let $sk_{id_j} = \mathsf{pk}^{\alpha}_{id_j}$ be a private key for the identity id_j. First we run $\mathsf{Decrypt}_1$ of IBE to decrypt c_{j3}:

$$X = \mathsf{Decrypt}_1(c_{j3}, sk_{id_j}) .$$

Then to compute m the Decrypt proceeds as follows:

$$\begin{aligned} & \frac{c_{j2}}{e(c_{j1}, \mathsf{H}_1(X)))} \\ = & \frac{m \cdot e(g^r, \mathsf{H}_1(X))}{e(g^r, \mathsf{H}_1(X))} \\ = & m . \end{aligned}$$

3.3.1 Efficiency Analysis

In Table 3.1 we count the number of calculations performed in TID-PRE, in particular we count the number of calculations performed during KeyGen, Pkeygen, Encrypt, Preenc and Decrypt. The efficiency of TID-PRE is the same as the efficiency of the Green and Ateniese [51] scheme. Hence, we succeed to have a PRE scheme that enables the delegator to define a fine-grained AC policy without sacrificing the efficiency.

Table 3.1: *Efficiency of* TID-PRE. *We denote by - no computations.*

	Exp.($\mathbb{G}$)	Exp.($\mathbb{G}_T$)	Pairing	H_1	H_2
KeyGen	1	-	-	-	-
Pkeygen	2	-	1	1	1
Encrypt	1	-	1	-	-
Preenc	1	-	1	1	1
Decrypt	1	-	1	-	-
Decrypt (Re-encrypted Ciphertexts)	-	-	2	1	-

KeyGen requires one exponentiation in $\mathbb{G}$. Pkeygen requires two exponentiations in $\mathbb{G}$, one pairing operation, one hash function H_1 and one hash function H_2. Note that the pairing operation is performed in the $\mathsf{Encrypt}_1$ algorithm (which is part of the IBE scheme) and is independent of X and can be computed ahead of time and only once. If the pairing is computed ahead of time, then the pairing operation of the Pkeygen is replaced with one exponentiation in $\mathbb{G}_T$. Encrypt requires one exponentiation in $\mathbb{G}$ and one pairing operation. The pairing operation here is also independent of the message m and can be computed ahead of time and only once. Preenc requires one exponentiation in $\mathbb{G}$, one pairing operation and two hash functions : H_1 and H_2. Decrypt requires one exponentiation in $\mathbb{G}$ and one pairing operation when the ciphertext is not re-encrypted, and it requires two pairing operations and one hash function H_1 when the ciphertext is re-encrypted.

3.3.2 Security Proof

In this section we first prove that the proposed scheme is IND-ID-PR-CPA secure in the random oracle model.

Theorem 1. *For the* TID-PRE *scheme described in Section 3.3, any adversary's advantage in* TID-PRE *security-game is negligible.*

Proof. We suppose that the total number of queries issued to H_1 and H_2 is bounded by the integer q_1 and q_2, respectively[1]. Suppose an adversary $\mathcal{A}$ has the non-negligible advantage ε in the TID-PRE security-game. The security proof is done through a sequence of games.

Game_0: In this security-game, the simulator $\mathcal{B}$ faithfully answers the oracle queries from $\mathcal{A}$. Specifically, $\mathcal{B}$ simulates the random oracle H_1 as follows: $\mathcal{B}$ maintains a list of vectors, each of them containing a request message, an element of $\mathbb{G}$ (the hash-code for this message), and an element of $\mathbb{Z}_p$. After receiving a request message, $\mathcal{B}$

[1] For simplicity of description, it is reasonable to assume that the total number is counted for queries with different inputs.

first checks its list to see whether the request message is already in the list. If the check succeeds, $\mathcal{B}$ returns the stored element of $\mathbb{G}$; otherwise, $\mathcal{B}$ returns g^y, where y is a randomly chosen element from $\mathbb{Z}_p$, and stores the new vector in the list. $\mathcal{B}$ simulates the random oracle H_2 as follows: $\mathcal{B}$ maintains a list of vectors, each of them containing a request message and an element of $\mathbb{Z}_p$ (the hash-code for this message). After receiving a request message, $\mathcal{B}$ first checks its list to see whether the request message is already in the list. If the check succeeds, $\mathcal{B}$ returns the stored element of $\mathbb{Z}_p$; otherwise, $\mathcal{B}$ returns u which is a randomly chosen element of $\mathbb{Z}_p$, and stores the new vector in the list. Let $\delta_0 = \Pr[b' = b]$, as we assumed at the beginning, $|\delta_0 - \frac{1}{2}| = \varepsilon$.

Game_1: In this security-game, $\mathcal{B}$ performs as follows.

- Setup: $\mathcal{B}$ faithfully simulates the setup phase.
- Phase 1: $\mathcal{B}$ randomly selects $j \in \{1, 2, \cdots, q_1 + 1\}$. If $j \leq q_1$, we assume the j-th input to H_1 is $\tilde{id}$. $\mathcal{B}$ faithfully answers the oracle queries from $\mathcal{A}$.
- Challenge: After receiving (m_0, m_1, t^*, id^*) from $\mathcal{A}$, if one of the following events occurs, $\mathcal{B}$ aborts as a failure.
  - id^* has been issued to H_1 but is not the j-th query,
  - id^* has not been issued to H_1 but $j \leq q_1$.

  $\mathcal{B}$ faithfully returns the challenge.
- Phase 2: $\mathcal{B}$ answers the oracle queries faithfully.
- Guess: the adversary outputs a guess $b' \in \{0, 1\}$.

Note that, if the $\mathcal{B}$ does not abort and the j-th query has been made in Phase 1, then $id^* = \tilde{id}$. The probability that $\mathcal{B}$ successfully ends is $\frac{1}{q_1+1}$, i.e. the probability that $\mathcal{B}$ does not abort in its execution is $\frac{1}{q_1+1}$. Let $\delta_1 = \Pr[b' = b]$ when $\mathcal{B}$ successfully ends, in which case $\delta_1 = \delta_0$. Let θ_1 be the probability that $\mathcal{B}$ successfully ends and $b' = b$. We have $\theta_1 = \frac{\delta_1}{q_1+1}$.

Game_2: In this security-game, $\mathcal{B}$ simulates the protocol execution and answers the oracle queries from $\mathcal{A}$ in the following way.

- Setup: $\mathcal{B}$ faithfully simulates the setup phase. Recall that $\mathsf{pk} = g^\alpha$.
- Phase 1: $\mathcal{B}$ randomly selects $j \in \{1, 2, \cdots, q_1 + 1\}$. If $j \leq q_1$, suppose the input of the j-th query to H_1 is $\tilde{id}$. $\mathcal{B}$ answers the oracle queries from $\mathcal{A}$ as follows:
  - Answer KeyGen faithfully, except that $\mathcal{B}$ aborts as a failure when $\tilde{id}$ is the input to a KeyGen query.

- $\mathsf{Pkeygen}$ query with (id, id', t): If $id = \tilde{id}$, $\mathcal{B}$ returns the proxy key $rk_{id \to id'}$, where

  $$g_{t \sim id'} \in_R \mathbb{G},\ \ X_{t \sim id'} \in_R \mathbb{G}_T,\ \ rk_{id \to id'} = (t, g_{t \sim id'}, \mathsf{Encrypt}_2(X_{t \sim id'}, id')).$$

  otherwise, $\mathcal{B}$ answers the query faithfully. If id' has been queried to KeyGen, when $X_{t \sim id'}$ is queried to H_1 then $\mathcal{B}$ returns $g_{t \sim id'} \cdot h^{-1}_{t \sim id'}$ where $h_{t \sim id'} \in_R \mathbb{G}$.

- Preenc query with (m, t, id, id'): If $id = \tilde{id}$, $\mathcal{B}$ returns

  $$r \in_R \mathbb{Z}_p,\ X_{t \sim id'} \in_R \mathbb{G}_T,\ c' = (g^r, e(g^r, \mathsf{H}_1(X_{t \sim id'})), \mathsf{Encrypt}_2(X_{t \sim id'}, id')).$$

  otherwise, $\mathcal{B}$ answers the query faithfully.

- Challenge: After receiving (m_0, m_1, t^*, id^*) from the adversary, if one of the following events occurs, $\mathcal{B}$ aborts as a failure.

  - id^* has been issued to H_1 but is not the j-th query,
  - id^* has not been issued to H_1 but $j \leq q_1$.

  Note that, if $\mathcal{B}$ does not abort then either $j \leq q_1$ and $id^* = \tilde{id}$ is the input to j-th H_1 query or $j = q_1 + 1$ and id^* has not been the input to any H_1 query. In the latter case, $\mathcal{B}$ sets $\mathsf{H}_1(id^*) = g^\beta$ where $\beta \in_R \mathbb{Z}_p$, and returns $c^* = (c_1^*, c_2^*, c_3^*)$ as the challenge to the adversary, where:

  $$b \in_R \{0,1\},\ \ r \in_R \mathbb{Z}_p,\ \ T \in_R \mathbb{G}_T,\ \ c_1^* = g^r,\ \ c_2^* = m_b \cdot T,\ \ c_3^* = t^*.$$

- Phase 2: $\mathcal{B}$ answers the oracle queries from $\mathcal{A}$ as in Phase 1.

- Guess: the adversary outputs a guess $b' \in \{0,1\}$.

Let θ_2 be the probability that $\mathcal{B}$ successfully ends and $b' = b$. We have $\theta_2 = \frac{1}{2(q_1+1)}$ since $T \in_R \mathbb{G}_T$. Let E_1 be the event that, for some id' and t, the adversary issues a H_2 query with the input $g^{\alpha \cdot \beta}||t$ or $X_{t \sim id'}$ is issued to H_1 while id' has not been issued to KeyGen. Compared with Game_1, Game_2 differs when E_1 occurs. From the difference lemma [94], we have $|\delta_2 - \delta_1| \leq \varepsilon_2$ which is negligible in the random oracle model based on the BDH assumption. Note that $(\mathsf{Setup}_1, \mathsf{KeyGen}_1, \mathsf{Encrypt}_1, \mathsf{Decrypt}_1)$ is one-way based on the BDH assumption and BDH implies CDH.

From $|\theta_2 - \theta_1| \leq \varepsilon_2$ and $\theta_2 = \frac{1}{2(q_1+1)}$, we have $|\frac{1}{2(q_1+1)} - \theta_1| \leq \varepsilon_2$. In addition, from $|\delta_0 - \frac{1}{2}| = \varepsilon$, $|\delta_1 - \delta_0| \leq \varepsilon_1$ and $\theta_1 = \frac{\delta_1}{q_1+1}$, we have $\frac{\varepsilon}{q_1+1} \leq \frac{\varepsilon_1}{q_1+1} + \varepsilon_2$. Because ε_i $(1 \leq i \leq 2)$ are negligible and ε is assumed to be non-negligible, we get a contradiction. As a result, the proposed scheme is $\mathsf{IND\text{-}ID\text{-}PR\text{-}CPA}$ secure based on the CDH assumption in the random oracle model, given that $(\mathsf{Setup}_1, \mathsf{KeyGen}_1, \mathsf{Encrypt}_1, \mathsf{Decrypt}_1)$ is one-way. □

3.4 Properties

Recall that Ateniese *et al.* [13] describe a number of properties for proxy re-encryption schemes. Our scheme possesses the following properties:

- Uni-directional. The re-encryption key is generated by the delegator, hence delegation is only from the delegator to the delegatee but not from the delegatee to the delegator.
- Non-Interactive. The delegator creates the re-encryption key by himself, neither the delegatee nor any other party is involved.
- Collusion-Resistant. The delegatee and the proxy together cannot recover the delegator's private key sk_{id_i}; in particular, they cannot compute new re-encryption keys for new types from Theorem 1.

3.5 Conclusion

In this chapter we present a TID-PRE scheme which has been proven to be secure against a chosen plaintext attack. Our scheme enables the delegator to provide different re-encryption capabilities to the proxy while using the same key pair. The re-encryption key is constructed by the delegator such that the proxy can use the re-encryption key to re-encrypt only ciphertext associated with one type. This property helps the delegator to reduce the trust on the server, since even is the proxy and the delegatee collude, at most they can learn is the contents of the messages belonging to one type.

Chapter 4

Efficient Attribute-Based Encryption Schemes

In the previous chapter we presented an encryption scheme where the access to the data depends on the name of the user. In this chapter we present new schemes where the access to the data depends on the attributes of the user. In particular we propose new ciphertext-policy attribute-based encryption (CP-ABE) schemes. In CP-ABE the data is encrypted under an access control policy defined by a user who encrypts the data and a user secret key is associated with a set of attributes which identify the user. A user can decrypt the ciphertext if and only if his attributes satisfy the access control policy. In CP-ABE, since the user enforces the access control policy at the encryption phase, the policy moves with the encrypted data. This is important for data storage servers where data confidentiality must be preserved even if the server is compromised or un-trusted.

We begin this chapter with an introduction and review of the related work on attribute-based encryption (ABE). Then we review the definitions of access structure and access tree, and review two secret sharing schemes. Next we give a formal definition for CP-ABE along with the security definition under which our schemes are proven to be secure. We continue the chapter by proposing two efficient CP-ABE schemes: the first scheme, which we refer to as basic CP-ABE (B-CP-ABE), can express any access control policy represented by a formula involving $\wedge$ (and) and $\vee$ (or) Boolean operators and the second scheme, which we refer to as extended CP-ABE (E-CP-ABE), can express any access control policy represented by a formula involving the *of* (threshold) operator, in addition to the $\wedge$ and $\vee$ operators. We provide a comparison with relevant CP-ABE schemes and show that our schemes are more efficient. We also show that the proposed schemes are secure under the Decision Bilinear Diffie-Hellman assumption (DBDH) in the standard model. Finally we discuss how to update the proposed schemes. The contents of this chapter is adapted from a refereed conference paper [6].

4.1 Introduction

As mentioned in Chapter 1, traditional public-key encryption (PKE) uses two keys: a private key which is kept secret and a public key which is widely distributed. For example, if Alice wants to send a confidential message to Bob, she can encrypt the message with the public key of Bob and only Bob can decrypt the message using his private key. In a Public-Key Infrastructure (PKI), a public key must be obtained from, or at least be certified by the Trusted Third Party (TTP) of the PKI. In Identity-Based Encryption (IBE), as mentioned in Section 2.6, any string (for example `bob@acm.org`) can be used to generate a public key without the involvement of the TTP [27, 89, 38]. IBE thus allows a degree of flexibility that a PKI cannot offer. However, if Alice does not know the identity of the recipient, but instead she only knows certain attributes of the recipient, then neither a PKI nor IBE will work. For example imagine that Alice wishes to communicate with her former classmates, but she does not know their email addresses.

The solution to this problem is provided by Attribute-Based Encryption (ABE), which identifies a user with a set of attributes [104]. In their seminal paper, Sahai and Waters use biometric measurements as attributes in the following way. A secret key based on a set of attributes ω, can decrypt a ciphertext encrypted with a public key based on a set of attributes ω', only if the sets ω and ω' overlap sufficiently as determined by a threshold value t. In the sequel we will refer to the Sahai and Waters scheme as the SW scheme. A more general policy to decide which attributes are required to decrypt a message is provided by a Boolean formula. For example the Boolean formula $\tau = (class1978 \ \wedge \ mycollege \vee \ myteacher)$ states that all students with the attribute $class1978$ who studied at $mycollege$ as well as the teacher possessing the attribute $myteacher$ would satisfy the policy.

There are two variants of ABE: Key-Policy ABE (KP-ABE) [50] and Ciphertext-Policy ABE (CP-ABE) [20, 37]. In CP-ABE the ciphertext is associated with the AC policy - the encrypting party determines the AC policy under which the data can be decrypted, while the secret key is associated with a set of attributes. The secret key can decrypt a ciphertext only if the attribute set of the secret key satisfies the AC policy of the ciphertext. In KP-ABE the idea is reversed. In particular, in KP-ABE the ciphertext is associated with a set of attributes and the secret key is associated with the AC policy. The encryptor does not define the AC policy and has no control over who has access to the data except by defining the set of descriptive attributes necessary to decrypt the ciphertext. The trusted authority who generates the user's secret key defines the combination of attributes (i.e. AC policy) for which the secret key can be used.

4.1.1 Related Work

Pirretti *et al.* [83] give a construction and implementation of a modified SW scheme, which, compared to the original SW scheme, drastically reduces the computational overhead during the encryption and the key generation phase. The Pirretti *et al.* [83] scheme is secure in the random oracle model. Goyal *et al.* [50] introduce the idea of KP-ABE where the secret key associated with the access tree, representing the AC policy, defines which ciphertext a user is able to decrypt. The Goyal *et al.* scheme is an extension of the SW scheme where instead of using the Shamir [88] secret sharing in the key generation phase, the trusted authority uses a more generalized form of secret sharing to enforce a monotonic access tree. Chase [33] constructs a multi-authority ABE scheme, which allows multiple independent authorities to monitor attributes and distribute secret keys. A related work to KP-ABE is predicate encryption or searching in encrypted data [60, 11, 24, 30]. Predicate encryption has the advantage of providing ciphertext anonymity by hiding the access structures, however, the system is less expressive compared to schemes which leave the access structures in the clear. Smart [95] gives an encryption scheme which encrypts the data according to an arbitrary collection of identities. However, the problem of attacks from colluding users is not addressed.

The first CP-ABE scheme proposed by Bethencourt *et al.* [20] uses threshold secret sharing to enforce the policy in the encryption phase. We will henceforth refer to this scheme as the BSW scheme. The main drawback of the BSW scheme is that it requires polynomial interpolation to reconstruct the secret, thus many expensive pairing and exponentiation operations in the decryption phase are required. This scheme is secure in the generic group model. The CP-ABE scheme proposed by Cheung and Newport [37] does not use threshold secret sharing but uses random elements to enforce the policy in the encryption phase. We will henceforth refer to this scheme as the CN scheme. The CN scheme has two drawbacks. First, the CN scheme is not sufficiently expressive since it supports only policies with the $\wedge$ operator. Second, the size of the ciphertext and the secret key increases linearly with the total number of attributes in the system. This makes the CN scheme inefficient. Goyal at el. [49] give a "bounded" CP-ABE construction. The disadvantage of [49] is that the depth of the access trees d under which messages can be encrypted is defined in the Setup phase. Thus, the user who wants to encrypt a message is restricted to use only an access tree which has a depth $d' \leq d$.

High Level Contributions. In this chapter we focus on the efficiency of the CP-ABE scheme with a security proof based on standard complexity assumptions. Previous CP-ABE systems could either support only $\wedge$ nodes in the AC policy [37], or have a security proof only in the generic group model [20] or specify the depth of the access tree in the Setup phase [49]. We propose schemes which compared to existing schemes are more efficient and at least as expressive. Our contribution is twofold:

- We present a new technique for realizing CP-ABE schemes which does not use threshold secret sharing. We will refer to it as the B-CP-ABE scheme. In this scheme the encryptor defines the privacy policy through an access tree which is an n-ary tree represented by $\wedge$ *and/or* $\vee$ nodes. Realizing a scheme which does not use threshold secret sharing is important for resource constraint devices since calculating polynomial interpolations to construct the secret is computationally expensive. We compare the efficiency of B-CP-ABE with the most relevant scheme in the literature, which is the CN scheme, and show that our scheme is more efficient since it requires fewer computations in key generation, encryption and decryption phases.

- We provide a second CP-ABE scheme which uses Shamir's (k, t) threshold secret sharing technique [88]. We will refer to it as the E-CP-ABE scheme. The access tree is an n-ary tree represented by $\wedge$, $\vee$ and *of*(threshold) nodes. We compare the efficiency of the E-CP-ABE scheme with the BSW scheme and show that our scheme requires fewer computations in key generation, encryption and decryption phases.

4.2 Background

First, we give a definition of an access structure. Next, we introduce the concept of the access tree. Finally, we briefly review to relevant secret sharing schemes.

4.2.1 Access Structures

Definition. (Access Structure [17]). *Let $\mathbb{P} = \{P_1, P_2, ..., P_n\}$ be a set of parties. A collection $\mathbb{A} \subseteq 2^{\mathbb{P}}$ is monotone if $\forall B, C$: if $B \in \mathbb{A}$ and $B \subseteq C$ then $C \in \mathbb{A}$. An access structure (respectively, monotone access structure) is a collection (respectively, monotone collection) $\mathbb{A}$ of non-empty subsets of $\mathbb{P}$, i.e. $\mathbb{A} \subseteq 2^{\mathbb{P}} \setminus \{\emptyset\}$. The sets in $\mathbb{A}$ are called the authorized sets and the sets not in $\mathbb{A}$ are called the unauthorized sets.*

In CP-ABE, instead of parties we use attributes and the access structure $\mathbb{A}$ will contain the set of authorized attributes.

4.2.2 Access Tree

In our scheme the access structure is represented by an access tree. The access tree is an n-ary tree, in which leaves are attributes and inner nodes are $\wedge$ and $\vee$ Boolean operators. Intuitively, the access tree is an AC policy which specifies which combination of attributes can decrypt the ciphertext. Consider the following example where a patient wants to specify access restrictions on his medical data. The patient wants to allow his data to be seen by Doctor A who works at Department A or by Doctor B who works at Department B. Using Boolean operators the patient defines the following access structure: $\tau_{Data} = (\mathsf{Doc.A} \wedge \mathsf{Dep.A}) \vee (\mathsf{Doc.B} \wedge \mathsf{Dep.B})$ which is translated into the following access tree:

To decrypt the ciphertext which is encrypted according to τ_{Data}, the decryptor must possess a secret key, which is associated with the attribute set which satisfies τ_{Data}. To decide whether an access tree is satisfied we interpret each attribute as a

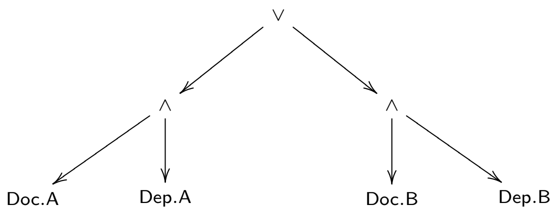

Figure 4.1: *Access tree* τ_{Data} = $(\mathsf{Doc.A} \wedge \mathsf{Dep.A}) \vee (\mathsf{Doc.B} \wedge \mathsf{Dep.B})$.

logical variable. Possession of the secret key for the corresponding attribute makes the logical variable true. If the decryptor does not possess the attribute, the variable is false. For the AC policy above there are several sets of attributes that can satisfy the access tree, such as: the secret key associating with the attribute set $(\mathsf{Doc.A}, \mathsf{Dep.A})$, the secret key associating with the attribute set $(\mathsf{Doc.B}, \mathsf{Dep.B})$, or the secret key associating with all attributes defined in the access tree.

4.2.3 Secret Sharing

When we design our CP-ABE schemes, in the encryption phase we map an AC policy into an encryption key by using two secret-sharing schemes: the Benaloh-Leichter [19] and the Shamir [88] schemes.

Benaloh-Leichter's Scheme

Let the secret domain S be the set $\mathbb{Z}_p$. The function $(s, \mathbb{A})$, where $s \in S$ is the secret to be shared and $\mathbb{A}$ is the monotone access structure, is defined as follows:

- If $(s, \wedge(A_1, A_2, ..., A_n))$ then $\bigcup_{1 \leq i \leq n}(s_i, A_i)$, where $n-1$ shares s_i are random elements of $\mathbb{Z}_p$, such that $1 \leq s_i \leq p-1$, and the last share $s_n = s - \sum_{i=1}^{n-1} s_i$ [1].
- If $(s, \vee(A_1, A_2, ..., A_n))$ then $\bigcup_{1 \leq i \leq n}(s, A_i)$.

We can illustrate the above formal definition with the following simple example. Let the monotone access structure $\mathbb{A}$ be τ_{Data} as defined in Section 4.2.2. To share s according to τ_{Data}, the Benaloh-Leichter scheme proceeds as follows (the sharing of s is also illustrated in Figure 4.2):

- Move s to $\vee$, and assign s to $\mathsf{Doc.A} \wedge \mathsf{Dep.A}$ and $\mathsf{Doc.B} \wedge \mathsf{Dep.B}$.
- Chose at random s_A and s_C and assign s_A to Doc.A and s_C to Doc.B. Then compute $s_B = s - s_A$ and assign it to Dep.A and compute $s_D = s - s_C$ and assign it to Dep.B.

[1] Note that all calculations are done *mod p*. However for presentation purposes we omit the *mod* operation.

To compute s, at least two shares are needed: either shares s_A and s_B, or shares s_C and s_D, since $s = s_A + s_B$ and $s = s_C + s_D$. Note that by only knowing either shares s_A and s_C, or shares s_B and s_D, one cannot compute s, since $s \neq s_A + s_C$ and $s \neq s_B + s_D$. In our schemes, first the decryptor needs to have either Doc. A and Dep. A or Doc. B and Dep. B in order to reconstruct s, and then be able to decrypt the ciphertext. This will be more clear in the following sections when we will present our schemes.

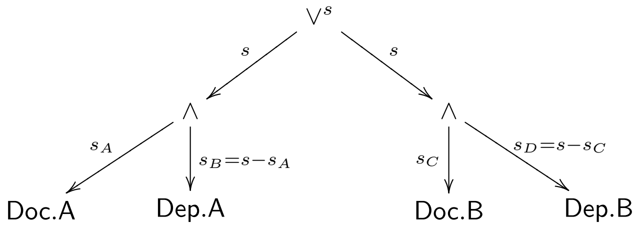

Figure 4.2: *Assigning secret shares to each leaf node in the access tree* $\tau_{Data} =$ (Doc.A $\wedge$ Dep.A) $\vee$ (Doc.B $\wedge$ Dep.B).

Shamir's Scheme

In Shamir's secret sharing scheme [88] a secret s is divided into n shares in such a way that any subset of t shares, where $t \leq n$, can together reconstruct the secret; no subset smaller than t can reconstruct the secret. The technique is based on polynomial interpolation, where a polynomial $y = f(x)$ of degree $t-1$ over a finite field $\mathbb{F}$ is uniquely defined by t points (x_i, y_i). The details of the scheme are as follows:

- Let the secret domain S be the set $\{0, ..., p-1\}$. To share a secret $s \in S$ among n parties, the scheme proceeds as follows:
  - Define $a_0 = s$.
  - Select $t-1$ random coefficients $(a_1, ..., a_{t-1}) \leq p-1$, and construct the polynomial $f(x) = \Sigma_{j=0}^{t-1} a_j x^j$.
  - Compute $s_i = f(i)$ and assign the share (i, s_i) to the party p_i.

Any group of t or more parties can pool their distinct shares together and reconstruct s. Let $(1, s_1), (2, s_2), ..., (t, s_t)$ be shares that can reconstruct the secret s. The secret s is reconstructed as follows:

- Define $(x_0, y_0) = (1, s_1)$, $(x_1, y_1) = (2, s_2), ...,$ $(x_{t-1}, y_{t-1}) = (t, s_t)$.
- Compute the Lagrange coefficients $l_j(x) = \Pi_{1 \leq j \leq t, j \neq i} \frac{x - x_j}{x_i - x_j}$.

- Define the polynomial $f(x) = \Sigma_{j=0}^{t-1} y_j l_j(x)$ and output the secret $f(0) = a_0 = s$.

Note that the following system of equations:

$$\begin{aligned} f(i_1) &= a_0 + a_1 i_1 + ... + a_{t-1} {i_1}^{t-1} \ , \\ f(i_2) &= a_0 + a_1 i_2 + ... + a_{t-1} {i_2}^{t-1} \ , \\ &\vdots \\ f(i_t) &= a_0 + a_1 i_t + ... + a_{t-1} {i_t}^{t-1} \ , \end{aligned}$$

has a unique solution for $(a_0, ..., a_t)$ based on the non-zero Vandermonde matrix:

$$\triangle = \begin{pmatrix} 1 & i_1 & \ldots & i_1^{t-1} \\ 1 & i_2 & \ldots & i_2^{t-1} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & i_t & \ldots & i_t^{t-1} \end{pmatrix} .$$

We can illustrate Shamir's secret sharing with the following example. Let the monotone access structure $\mathbb{A}$ be $\tau'_{Data} =$ 2 of (Doc.B, Dep.B, Specialist). To share s according to $\mathbb{A}$, the Shamir's secret sharing scheme proceeds as follows (the sharing of s is also illustrated in Figure 4.3):

- Select one random coefficient $a_1 \leq p - 1$, and construct the polynomial $f(x) = s + a_1 x$.

- Compute:

$$\begin{aligned} f(A) &= s + a_1 A \ , \\ f(B) &= s + a_1 B \ , \\ f(C) &= s + a_1 C \ , \end{aligned}$$

  and assign the share $(A, f(A))$ to Doc. B, $(B, f(B))$ to Dep. B and $(C, f(C))$ to Specialist.

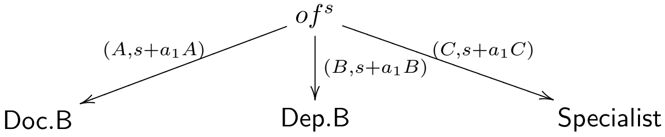

Figure 4.3: *Assigning secret shares to each leaf node in the access tree* $\tau'_{Data} =$ 2 of (Doc.B, Dep.B, Specialist).

Any two shares can reconstruct s. Let $(A, f(A)), (C, f(C))$ be shares to reconstruct the secret s. The secret s is reconstructed as follows:

- Define $(x_0, y_0) = (A, s_A)$ and $(x_1, y_1) = (C, s_C)$.
- Compute the Lagrange coefficients:

$$l_0(x) = \frac{x - C}{A - C} \quad , \quad l_1(x) = \frac{x - A}{C - A} \ .$$

- Define the polynomial $f(x) = f(A) \cdot l_0 + f(C) \cdot l_1$ and output the secret:

$$\begin{aligned} f(0) &= s_A \cdot l_0(0) + s_C \cdot l_1(0) \\ &= (s + a_1 A) \cdot \frac{0 - C}{A - C} + (s + a_1 C) \cdot \frac{0 - A}{C - A} \\ &= s \ . \end{aligned}$$

4.3 Ciphertext-Policy ABE

In this section we formally define the algorithms of CP-ABE. A CP-ABE scheme consists of four algorithms [20]:

- Setup(λ). Run by the trusted authority (TA) this algorithm takes as input a security parameter λ and outputs the public parameters $\mathcal{PP}$ and a master key msk $\in \mathcal{K}$. The public parameters define a universe (set) of attributes Ω, and give descriptions about message space $\mathcal{M}$ and ciphertext space $\mathcal{C}$.
- KeyGen(ω, msk). Run by the trusted authority this algorithm takes as input the master key msk $\in \mathcal{K}$ and a set of attributes $\omega \subseteq \Omega$. The algorithm outputs a secret key $\mathsf{sk}_\omega \in \mathcal{K}$ associated with ω.
- Encrypt(m, τ). Run by the encryptor this algorithm takes as input a message $m \in \mathcal{M}$, and an AC policy τ. The algorithm will return the ciphertext $c_\tau \in \mathcal{C}$ such that only users who have the secret key sk_ω associated with the attributes that ω that satisfies the access tree τ will be able to decrypt the ciphertext.

  To avoid confusion, in the sequel we will refer to τ as the AC policy only.
- Decrypt($c_\tau, \mathsf{sk}_\omega$). Run by the decryptor this algorithm takes as input a ciphertext $c_\tau \in \mathcal{C}$, a secret key $\mathsf{sk}_\omega \in \mathcal{K}$ associated with $\omega \subseteq \Omega$, and it outputs a message $m \in \mathcal{M}$ or an error symbol $\perp$.

We assume that the TA is responsible to publish a universe (set) of attributes Ω. For instance, in a healthcare domain, $TA_{Healthcare}$ may be responsible to publish $\Omega_{Healthcare}$, which may contain attributes such as: doctor, nurse, HIV patient etc, and in a university domain, $TA_{University}$ may be responsible to publish $\Omega_{University}$, which may contain attributes such as: job position, age, research interest etc. We also assume that the process of obtaining a secret key sk_ω associated with a set of attributes ω is straightforward, that is, the user has to go to the TA to apply for sk_ω and prove that he/she indeed possesses the attribute set ω.

Correctness. We call a CP-ABE *correct* if decrypting a ciphertext c_τ with a secret key sk_ω, such that the attribute set ω satisfies the ciphertext's AC policy τ, always results in the same message m as was input by the Encrypt algorithm when it produced the ciphertext c_τ. More precisely:

$$\Pr\left[\begin{array}{cc} (\mathsf{msk}, \mathcal{PP}) \leftarrow \mathsf{Setup}(\lambda), & \mathsf{sk}_\omega \leftarrow \mathsf{KeyGen}(\omega, \mathsf{msk}), \\ c_\tau \leftarrow \mathsf{Encrypt}(m, \tau) \ : & m \leftarrow \mathsf{Decrypt}(c_\tau, \mathsf{sk}_\omega) \end{array}\right] = 1 \ .$$

4.3.1 Security Definitions

Indistinguishability under chosen-plaintext attack is modelled by a security-game. The security-game is carried out between a challenger and an adversary $\mathcal{A}$, where the challenger simulates the protocol execution and answers queries from $\mathcal{A}$.

Definition. *A* CP-ABE *scheme is said to be secure against chosen-plaintext attacks* IND-sAtt-CPA *if any polynomial-time adversary $\mathcal{A}$ has only a negligible advantage in the* CP-ABE *selective security-game defined as follows:*

- Init. *The adversary $\mathcal{A}$ chooses the challenge AC policy τ^* and gives it to the challenger.*

- Setup. *The challenger runs* Setup *to generate* $(\mathcal{PP}, \mathsf{msk})$ *and gives the public parameters $\mathcal{PP}$ to $\mathcal{A}$.*

- Phase1. *$\mathcal{A}$ makes a secret key request to the* KeyGen *oracle for any attribute set $\omega = \{a_j | a_j \in \Omega\}$, with the restriction that $a_j \notin \tau^*$. The challenger runs* $\mathsf{KeyGen}(\omega, \mathsf{msk})$ *and returns* sk_ω *to $\mathcal{A}$.*

- Challenge. *$\mathcal{A}$ sends to the challenger two messages m_0, m_1 such that $|m_0| = |m_1|$. The challenger picks a random bit $b \in \{0,1\}$ and returns to $\mathcal{A}$ the challenge ciphertext* $c_b \leftarrow \mathsf{Encrypt}(m_b, \tau^*, \mathcal{PP})$.

- Phase2. *$\mathcal{A}$ can continue querying* KeyGen *with the same restriction as in* Phase1.

- Guess. *$\mathcal{A}$ outputs a guess $b' \in \{0,1\}$, and wins the game if $b' = b$.*

The advantage of $\mathcal{A}$ in winning the security-game is defined as:

$$\mathsf{ADV}^{\mathsf{IND-sAtt-CPA}}_{\mathcal{A},\mathsf{CP-ABE}}(\lambda) \overset{def}{=} \left|\Pr[b' = b] - \frac{1}{2}\right| \ ,$$

where the probability is over the random values chosen by $\mathcal{A}$ and the challenger.

In IND-sAtt-CPA, the adversary $\mathcal{A}$ must provide the challenge AC policy τ^* he wishes to attack before he receives the public parameters $\mathcal{PP}$ (i.e. the adversary $\mathcal{A}$ must provide τ^* in the Init phase). This means that if $\mathcal{A}$ in the Init phase picks the challenge AC policy $\tau^* = (\mathsf{A} \wedge \mathsf{B}) \vee \mathsf{C}$, then in Phase1 $\mathcal{A}$ can make secret key requests to KeyGen oracle for any attribute set ω with the restriction that attributes $\mathsf{A}, \mathsf{B}, \mathsf{C} \notin \omega$. This model is known as selective-attribute (sAtt) model and can be considered to be analogous to the selective-ID (sID) model [23, 30, 32] used in IBE schemes. In the sID model $\mathcal{A}$ commits ahead of time to the ID^* it will attack. This commitment forces $\mathcal{A}$ to make secret key requests to the KeyGen oracle for any ID such that $\mathsf{ID} \neq \mathsf{ID}^*$.

Recently Okamoto and Takashima [81] and Lewko *et al.* [63] propose fully secure ABE schemes in which the adversary announces τ^* in the Challenge phase.

4.4 Construction of B-CP-ABE

The algorithms of the B-CP-ABE scheme are defined as follows:

- $\mathsf{Setup}(\lambda)$: On input of the security parameter λ, the algorithm operates as follows:
  - Generate a bilinear group $\mathbb{G}$ of prime order p with a generator g and bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$.
  - Generate the attribute set $\Omega = \{a_1, a_2, \dots a_n\}$, for some integer n, and random elements $\alpha, t_1, t_2 \dots t_n \in \mathbb{Z}_p$.

  Let $y = e(g, g)^\alpha$ and $\{T_j = g^{t_j}\}_{1 \le j \le n}$. The public parameters are $\mathcal{PP} = (e, g, y, \{T_j\}_{1 \le j \le n})$ and the master secret key is $\mathsf{msk} = (\alpha, \{t_j\}_{1 \le j \le n})$.

- $\mathsf{KeyGen}(\omega, \mathsf{msk})$: The algorithm operates as follows:
  - Select a random value $r \in \mathbb{Z}_p$ and compute $d_0 = g^{\alpha - r}$.
  - For each attribute a_j in ω compute $d_j = g^{r t_j^{-1}}$.

  The algorithm returns the secret key $\mathsf{sk}_\omega = \big(d_0, \{d_j\}_{a_j \in \omega}\big)$.

- $\mathsf{Encrypt}(m, \tau)$: To encrypt a message $m \in \mathbb{G}_T$ the algorithm proceeds as follows:
  - Select a random element $s \in \mathbb{Z}_p$ and compute:

  $$\begin{aligned} c_0 &= g^s , \\ c_1 &= m \cdot y^s = m \cdot e\,(g, g)^{\alpha s} \ . \end{aligned}$$

  - Set the value of the root node of τ to be s and use the Benaloh-Leichter scheme presented in Section 4.2.3 to assign shares to the leaves of τ. This procedure is defined as follows. First, all child nodes are marked as un-assigned and the root node is marked as assigned. Then, recursively, for each un-assigned non-leaf node, do the following:
    - ∗ If the symbol is $\wedge$, assign a random value s_i where $1 \le s_i \le p - 1$ to each child node except to the last one. Assign the value $s_t = s - \sum_{i=1}^{t-1} s_i$ to the last child node. Mark this node assigned.
    - ∗ If the symbol is $\vee$, set the values of each child node to be s. Mark this node assigned.

- For each leaf attribute $a_{j,i} \in \tau$, compute $c_{j,i} = T_j^{s_i}$ where i denotes the index of the attribute in τ.

Figure 4.4 is an example of the generation of ciphertext components according to the AC policy $\tau_{Data} =$ (Doc.A $\wedge$ Dep.A) $\vee$ (Doc.B $\wedge$ Dep.B). We assume that g^{t_1} is assigned to the attribute Doc.A, g^{t_2} is assigned to the attribute Dep.A, g^{t_3} is assigned to the attribute Doc.B and g^{t_4} is assigned to the attribute Dep.B.

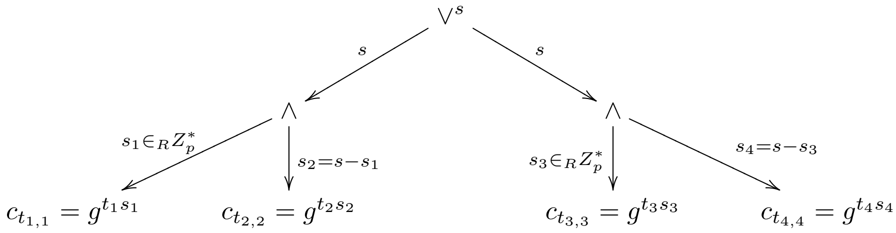
$\vee^s$
s
s
$\wedge$
$\wedge$
$s_1 \in_R Z_p^*$
$s_2 = s - s_1$
$s_3 \in_R Z_p^*$
$s_4 = s - s_3$
$c_{t_{1,1}} = g^{t_1 s_1}$
$c_{t_{2,2}} = g^{t_2 s_2}$
$c_{t_{3,3}} = g^{t_3 s_3}$
$c_{t_{4,4}} = g^{t_4 s_4}$

Figure 4.4: *Generating ciphertext components according to the AC policy* $\tau_{Data} =$ (Doc.A $\wedge$ Dep.A) $\vee$ (Doc.B $\wedge$ Dep.B).

The algorithm returns the ciphertext $c_\tau = \big(\tau, c_0, c_1, \{c_{j,i}\}_{a_{j,i} \in \tau}\big)$.

- Decrypt$(c_\tau, \mathsf{sk}_\omega)$: If ω does not satisfy τ, return $\perp$, otherwise the algorithm chooses the smallest set $\omega' \subseteq \omega$ (we assume that this can be computed efficiently by the decryptor) that satisfies τ and proceeds as follows:
  - compute:
  $$Z^{(1)} = \prod_{a_j \in \omega'} e\left(c_{j,i}, d_j\right) \ ,$$
  - compute:
  $$Z^{(2)} = e\left(c_0, d_0\right) \cdot Z^{(1)} \ ,$$
  - Return m', where
  $$m = \frac{c_1}{Z^{(2)}} \ .$$

In the scheme, when a message sender encrypts a message, along with the encrypted message, the sender specifies in cleartext the AC policy τ used to encrypt the message. However, the AC policy may reveal some sensitive information about the message being encrypted. Suppose, Alice encrypts the message under the AC policy Psychiatrist $\wedge$ Neurologists. From the AC policy, an adversary may conclude that Alice has a mental disorder. The ideal solution to prevent the adversary or unintended decryptor to learn some information about the message being encrypted is to remove the AC policy from the ciphertext. Therefore, to decrypt the ciphertext, the decryptor must try all possible attributes until the AC policy is satisfied. Although this can be computationally inefficient, it ensures that if the decryptor does not posses the right attributes he learns almost nothing about the AC policy which controls access to the message. Moreover, he doesn't learn the attribute list that he has to obtain from the trusted authority in order to decrypt the message.

Collusion Resistant. The most important property of an CP-ABE scheme is collusion resistance. This means that different users cannot combine their secret keys and decrypt a ciphertext that the colluding users separately should not have access to. To prevent collusion, the KeyGen algorithm of our scheme generates a different random value r for each user. Hence, keys generated for different users cannot be combined since they are randomized or personalized. Informally speaking, to decrypt the ciphertext the attacker must know how to recover $e(g,g)^{\alpha s}$. However, to do that the attacker first has to recover $e(g,g)^{rs}$, which would require the attacker to posses the secret key blinded with the same random value r.

Correctness. Let $c_\tau = (\tau, c_0 = g^s, c_1 = m \cdot e(g,g)^{\alpha s}, \{c_{j,i} = g^{t_j s_i}\}_{a_{j,i} \in \tau})$ be a ciphertext generated by $\mathsf{Encrypt}(m, \tau)$ and $\mathsf{sk}_\omega = (d_0 = g^{\alpha - r}, \{d_j = g^{r t_j^{-1}}\}_{a_j \in \omega})$ be a secret key generated by $\mathsf{KeyGen}(\omega, \mathsf{msk})$ such that the attribute set ω satisfy the AC policy τ. Let $\omega' \subseteq \omega$ be the smallest subset which satisfies τ. Note that if $\tau_{Data} = (\mathsf{Doc.A} \land \mathsf{Dep.A}) \lor (\mathsf{Doc.B} \land \mathsf{Dep.B})$ and $\omega = (\mathsf{Doc.A}, \mathsf{Dep.A}, \mathsf{Dep.B})$, then $\omega' = (\mathsf{Doc.A}, \mathsf{Dep.A})$. The reason for chosing ω' is that we need to use the correct shares to reconstruct s. If we use ω, then we cannot reconstruct s since $s \neq s_1 + s_2 + s_4$, however, if we use ω' then we can reconstruct s since $s = s_1 + s_2$.

The correctness of the construction demonstrates that when running Decrypt with key sk_ω on a ciphertext c_τ results in the same message m as was input by the Encrypt algorithm when it produced the ciphertext c_τ. We observe that:

$$\begin{aligned} Z^{(1)} &= \prod_{a_j \in \omega'} e\left(T_j^{s_i}, g^{r t_j^{-1}}\right) \\ &= \prod_{a_j \in \omega'} e\left(g^{t_j s_i}, g^{r t_j^{-1}}\right) \\ &= e\left(g, g\right)^{rs} \end{aligned}$$

and

$$\begin{aligned} Z^{(2)} &= e\left(c_0, d_0\right) \cdot e(g,g)^{rs} \\ &= e\left(g^s, g^{\alpha - r}\right) \cdot e\left(g, g\right)^{rs} \\ &= e\left(g^s, g^\alpha\right) \ . \end{aligned}$$

To output m, Decrypt computes:

$$\begin{aligned} & \frac{c_1}{Z^{(2)}} \\ = & \frac{m \cdot e\left(g, g\right)^{\alpha s}}{e\left(g^s, g^\alpha\right)} \\ = & \ m \ . \end{aligned}$$

4.4.1 Efficiency Analysis

In Table 4.1 we compare our B-CP-ABE scheme to the CN scheme. Our scheme requires fewer computations when running Encrypt, KeyGen and Decrypt than the CN scheme.

In our scheme, the number of calculations for Encrypt depends on the number of attributes in the AC policy τ. Thus, Encrypt requires $|\tau| + 1$ exponentiations in $\mathbb{G}$ and one exponentiation in $\mathbb{G}_T$. The number of calculations for KeyGen depends on the number of attributes in the set ω that the user has. Thus, KeyGen requires $|\omega| + 1$ exponentiations in $\mathbb{G}$. The number of calculations in Decrypt depends on the number of attributes in the attribute set ω'. Thus, Decrypt requires $|\omega'| + 1$ pairing operations. Decrypt also requires $|\omega'|$ multiplications but no exponentiations in $\mathbb{G}_T$. Typically, in a bilinear groups a pairing operation is more expensive than an exponentiation, while an exponentiation is more expensive than a multiplication. Hence, when constructing our scheme we are focused on minimizing the number of pairing operations in the Decrypt phase.

Table 4.1: *Comparison of our* B-CP-ABE *scheme with the* CN *scheme.*

	B-CP-ABE			CN		
	Exp.($\mathbb{G}$)	Exp.($\mathbb{G}_T$)	Pairing	Exp.($\mathbb{G}$)	Exp.($\mathbb{G}_T$)	Pairing
Encrypt	$\|\tau\|$+1	1	-	$\|\Omega\|$+1	1	-
KeyGen	$\|\omega\|$+1	-	-	$\|\Omega\|$+1	-	-
Decrypt	-	-	$\|\omega'\|$+1	-	-	$\|\Omega\|$+1
	Ω is the set of all attributes defined in the Setup phase . τ is the access policy . ω is the set of attributes the user has and $\omega \subseteq \Omega$. ω' the set of attributes satisfying the access policy and $\omega' \subseteq \omega$.					

4.4.2 Security Proof

We prove the following theorem:

Theorem 2. *Suppose the adversary $\mathcal{A}$ can win the* CP-ABE *security-game with a non-negligible advantage ε. Then we construct a polynomial-time reduction $\mathcal{B}$ that is able to solve the DBDH assumption with advantage $\frac{\varepsilon}{2}$.*

Proof. The challenger sets the groups $\mathbb{G}$ and $\mathbb{G}_T$, a generator g of group $\mathbb{G}$, a mapping function e, and selects at random: $a, b, c, \theta \in \mathbb{Z}_p$. The challenger flips a coin μ and sets $Z_\mu = e(g,g)^{abc}$ if $\mu = 0$ and $Z_\mu = e(g,g)^{\theta}$ otherwise. The challenger gives to the simulator $\mathcal{B}$ the DBDH instance: $(g, A, B, C, Z_\mu) = (g, g^a, g^b, g^c, Z_\mu)$. $\mathcal{B}$ will act as $\mathcal{A}$'s challenger in the IND-sAtt-CPA security-game as follows:

- Init. $\mathcal{A}$ chooses the challenge AC policy τ^* and gives it to $\mathcal{B}$.

- Setup. $\mathcal{B}$ selects at random $x' \in \mathbb{Z}_p$ and implicitly sets $\alpha = ab + x'$ by letting $e(g,g)^{\alpha} = e(g,g)^{ab} \cdot e(g,g)^{x'}$. For each $a_j \in \Omega$, $\mathcal{B}$ picks a random $k_j \in \mathbb{Z}_p$ and sets $T_j = B^{1/k_j}$ (thus $t_j = b/k_j$) if $a_j \notin \tau^*$ or $T_j = g^{k_j}$ if $a_j \in \tau^*$ (thus $t_j = k_j$). $\mathcal{B}$ sends the public parameters $\mathcal{PP} = (e, g, y, \{T_j\}_{1 \leq j \leq n})$ to $\mathcal{A}$.

  The distribution of the $\mathcal{PP}$ is identical to the $\mathcal{PP}$ of the scheme since by the DBDH assumption g is a random generator and a, b are random exponents (thus α in the view of $\mathcal{A}$ will be a random variable with the same distribution as in the scheme). Furthermore, the values of t_j are also random in the view of $\mathcal{A}$ since the k_j's are chosen at random from $\mathbb{Z}_p$.

- Phase1. $\mathcal{A}$ makes secret key requests for any set of attributes $\omega_j = \{a_j | a_j \in \Omega\}$ with the restriction that $a_j \notin \tau^*$. On each request $\mathcal{B}$ chooses a random variable $r' \in \mathbb{Z}_p$ and sets $d_0 = g^{x' - r'b}$ (implicitly it sets $r = ab + r'b$). The value of r is random in the view of $\mathcal{A}$ since r' is chosen at random from $\mathbb{Z}_p$. Therefore the distribution of d_0 is the same as in the scheme since:

$$\begin{aligned} d_0 &= g^{x'-r'b} \\ &= g^{\alpha - ab - r'b} \\ &= g^{\alpha - (ab + r'b)} \\ &= g^{\alpha - r} . \end{aligned}$$

  For each $a_j \in \omega_j$, $\mathcal{B}$ has to construct secret key components of the form $d_j = g^{r t_j^{-1}}$. Since $\mathcal{B}$ implicitly sets $r = ab + r'b$ and $t_j = b/k_j$ for each $a_j \notin \tau^*$, the valid form of the secret key component would be $d_j = g^{(ab+r'b)k_j/b}$. For each $a_j \in \omega_j$, $\mathcal{B}$ sets $d_j = A^{k_j} g^{k_j r'}$. This is a valid secret key component and can be computed by $\mathcal{B}$ since:

$$\begin{aligned} d_j &= g^{(ab+r'b)k_j/b} \\ &= g^{a k_j i} g^{k_j r'} \\ &= A^{k_j} g^{k_j r'} . \end{aligned}$$

  $\mathcal{B}$ sends the secret key $\mathsf{sk}_{\omega_j} = (d_0, \{d_j\}_{a_j \in \omega_j})$ to $\mathcal{A}$.

- Challenge. $\mathcal{A}$ submits two messages $m_0, m_1 \in \mathbb{G}_T$ such that $|m_0| = |m_1|$. $\mathcal{B}$ flips a fair binary coin $b \in \{0,1\}$, and returns the encryption of m_b. The encryption of m_b is done as follows:

  – Use g^c and Z_μ from the DBDH instance to compute c_0 and c_1, such that:

$$\begin{aligned} c_0 &= g^c , \\ c_1 &= m_b \cdot Z_\mu \cdot e\left(g, g\right)^{cx'} . \end{aligned}$$

Note that if $Z_\mu = e(g,g)^{abc}$, then c_1 is a valid ciphertext component since:

$$\begin{aligned} c_1 &= m_b \cdot Z_\mu \cdot e(g,g)^{cx'} \\ &= m_b \cdot e(g,g)^{abc} \cdot e(g,g)^{cx'} \\ &= m_b \cdot e(g,g)^{(ab+x')c} \\ &= m_b \cdot e(g,g)^{\alpha c} \ . \end{aligned}$$

- Next, $\mathcal{B}$ sets the value of the root node of τ^* to be g^c and it marks all child nodes as un-assigned and it marks the root node assigned. Recursively, for each un-assigned non-leaf node $\mathcal{B}$ operates as following.
  - $*$ If the symbol is $\wedge$, for each child except the last one $\mathcal{B}$ chooses h_i where $1 \leq h_i \leq p-1$, and assigns g^{h_i} to them, and to the last child it assigns $g^{h_t} = g^c / \sum_{i=1}^{t-1} g^{h_i}$. Mark this node assigned.
  - $*$ If the symbol is $\vee$, set the values of each child node to be g^c. Mark this node assigned.

  Note that $\mathcal{B}$ implicitly uses the Benaloh-Leichter secret sharing scheme to share c in the exponent, without knowing it.
- For every $a_{j,i} \in \tau^*$, compute $c_{j,i} = g^{h_i k_j}$.

The ciphertext $c_{\tau^*} = (\tau^*, c_0, c_1, \{c_{j,i}\}_{a_{j,i} \in \tau^*})$ is sent to $\mathcal{A}$ as a "challenge ciphertext".

- Phase2. $\mathcal{A}$ can continue with the secret key requests with the same restriction as in Phase1.
- Guess. $\mathcal{A}$ outputs a guess $b' \in \{0,1\}$.

If $b' = b$, $\mathcal{B}$ will guess that $\mu = 0$ and $Z_\mu = e(g,g)^{abc}$, otherwise $\mathcal{B}$ will guess that $\mu = 1$ and $Z_\mu = e(g,g)^\theta$. When $Z_\mu = e(g,g)^{abc}$, $\mathcal{B}$ gives a perfect simulation of the scheme since c_{τ^*} is a valid ciphertext. Therefore the advantage of $\mathcal{A}$ is:

$$\Pr\left[b' = b | Z_\mu = e(g,g)^{abc}\right] = \frac{1}{2} + \varepsilon \ .$$

If $\mu = 1$ then $Z_\mu = e(g,g)^\theta$ and c_{τ^*} is a random ciphertext for $\mathcal{A}$. Hence, $\mathcal{A}$ does not gain information about m_b and the advantage of $\mathcal{A}$ is:

$$\Pr\left[b' \neq b | Z_\mu = e(g,g)^{\theta}\right] = \frac{1}{2} \ .$$

Since $\mathcal{B}$ guesses $\mu' = 0$ when $b' = b$ and $\mu' = 1$ when $b' \neq b$, the overall advantage of $\mathcal{B}$ to break the DBDH assumption is:

$$\frac{1}{2}\Pr[\mu' = \mu | \mu = 0] + \frac{1}{2}\Pr[\mu' = \mu | \mu = 1] - \frac{1}{2} = \frac{\varepsilon}{2} \ . \ \square$$

4.5 Construction of E-CP-ABE

In the B-CP-ABE scheme the AC policy is an n-ary tree represented by $\wedge$ and $\vee$ nodes. This allows the user who performs encryption to express any privacy policy using Boolean formulas. Ideally, we would like to have an n-ary AC policy which supports *of* (threshold) operators, similar to the BSW scheme. The essential idea behind the threshold operator is to allow the encryptor to define the minimum number of attributes from a given list of attributes that the decryptor has to possess in order to decrypt the message. For instance, to decrypt the ciphertext encrypted under the policy $\tau'_{Data} = \text{ 2 of (Doc.B, Dep.B, Specialist)}$, the decryptor must have at least two out of three attributes. We now present our version of the four E-CP-ABE algorithms:

- Setup is the same as in the B-CP-ABE scheme.
- KeyGen is the same as in the B-CP-ABE scheme.
- Encrypt(m, τ) : To encrypt a message $m \in \mathbb{G}_T$ the algorithm proceeds as follows:
  - Select a random element $s \in \mathbb{Z}_p$ and compute:

$$\begin{aligned} c_0 &= g^s \ , \\ c_1 &= m \cdot y^s = m \cdot e\left(g, g\right)^{\alpha s} \ . \end{aligned}$$

  - Set the value of the root node to be s, mark all child nodes as un-assigned, and mark the root node assigned. Recursively, for each un-assigned non-leaf node, do the following:
    - ∗ If the symbol is *of*, the secret s is divided using $(t, \mathbf{n})$ Shamir's secret sharing technique where $t \neq \mathbf{n}$, and $\mathbf{n}$ is the total number of child nodes and t is the number of child nodes necessary to reconstruct the secret. To each child node a share $(i, s_i = f(i))$ is assigned. Mark this node assigned.
    - ∗ If the symbol is $\wedge$, the secret s is divided using $(t, \mathbf{n})$ Shamir's secret sharing technique where $t = \mathbf{n}$, and $\mathbf{n}$ is the number of the child nodes. To each child node a share $(i, s_i = f(i))$ is assigned. Mark this node assigned.
    - ∗ If the symbol is $\vee$, the secret s is divided using $(t, \mathbf{n})$ Shamir's secret sharing technique where $t = 1$ and $\mathbf{n}$ is the number of the child nodes. To each child node a share $(i, s_i = f(i))$ is assigned. Mark this node assigned.
  - For each leaf attribute $a_{j,i} \in \tau$, compute $c_{j,i} = T_j^{s_i}$, where i denotes the index of the attribute in the AC policy.

  The algorithm returns the ciphertext: $c_\tau = \left(\tau, c_0, c_1, \{c_{j,i}\}_{a_{j,i} \in \tau}\right)$.

- Decrypt$(c_\tau, \mathsf{sk}_\omega)$: If ω does not satisfy τ, return $\perp$, otherwise the algorithm chooses the smallest set $\omega' \subseteq \omega$ that satisfies τ and proceeds as follows:

- compute:

$$Z^{(1)} = \prod_{a_j \in \omega'} e\left(c_{j,i}, d_j\right)^{l_i(0)} \ ,$$

  where $l_i(0)$ for $i \in \mathbb{Z}_p$ is a Lagrange coefficient that can be computed by everyone who knows the index of the attribute in the AC policy.

- Compute:

$$Z^{(2)} = e\left(c_0, d_0\right) \cdot Z^{(1)} \ ,$$

- Return m', where

$$m = \frac{c_1}{Z^{(2)}} \ .$$

Correctness. The correctness of E-CP-ABE is similar to the correctness of B-CP-ABE. The only difference is in Decrypt when computing $Z^{(1)}$. We observe that:

$$\begin{aligned} Z^{(1)} &= e(T_j^{s_i}, g^{rt_j^{-1}})^{l_i(0)} \\ &= \prod_{a_j \in \omega'} e(g^{t_j s_i}, g^{rt_j^{-1}})^{l_i(0)} \\ &= \prod_{a_j \in \omega'} e(g, g)^{rs_i l_i(0)} \\ &= e(g, g)^{rs} \ . \end{aligned}$$

Since the value of $Z^{(1)}$ is the same as the value of $Z^{(1)}$ of the B-CP-ABE scheme, the values of $Z^{(2)}$ and m are computed in the same way as in the B-CP-ABE scheme.

4.5.1 Efficiency Analysis

In Table 4.2 we give a comparison of the efficiency of our E-CP-ABE scheme to the BSW scheme. Compared to the BSW scheme, our scheme is more efficient since it requires fewer computations for Encrypt, KeyGen and Decrypt.

In E-CP-ABE the number of calculations for Encrypt depends on the number of attributes in the AC policy τ and the number of calculations for KeyGen depends on the number of attributes in the set ω that the user has. Thus, Encrypt requires $|\tau| + 1$ exponentiations in $\mathbb{G}$ and one exponentiation in $\mathbb{G}_T$ and KeyGen requires $|\omega| + 1$ exponentiations in $\mathbb{G}$. E-CP-ABE differs from B-CP-ABE in the number of calculations in the Decrypt. E-CP-ABE uses Shamir secret sharing to enforce the AC policy in Encrypt. Hence, to reconstruct the secret s in the exponent, in Decrypt the user needs to compute Lagrange coefficients and use them as exponents for elements in group $\mathbb{G}_T$. In total Decrypt requires $|\omega'|$ exponentiations in $\mathbb{G}_T$ and $|\omega'| + 1$ pairing operations.

Table 4.2: *Comparison of our* E-CP-ABE *scheme with* BSW *scheme.*

	E-CP-ABE			BSW		
	Exp.($\mathbb{G}$)	Exp.($\mathbb{G}_T$)	Pairing	Exp.($\mathbb{G}$)	Exp.($\mathbb{G}_T$)	Pairing
Encrypt	$\lvert\tau\rvert$+1	1	-	2$\lvert\tau\rvert$+1	1	-
KeyGen	$\lvert\omega\rvert$+1	-	-	2$\lvert\omega\rvert$+1	-	-
Decrypt	-	$\lvert\omega'\rvert$	$\lvert\omega'\rvert$+1	-	$\lvert\omega'\rvert$	2$\lvert\omega'\rvert$
(Note:)	Ω is the set of all attributes defined in the Setup phase . τ is the access policy . ω is the set of attributes the user has and $\omega \subseteq \Omega$. ω' the set of attributes satisfying the access policy and $\omega' \subseteq \omega$.					

4.5.2 Security Proof

Theorem 3. *Suppose the adversary $\mathcal{A}$ can win the* CP-ABE *security-game with a non-negligible advantage ε. Then we construct a polynomial-time reduction $\mathcal{B}$ that is able to solve the DBDH assumption with advantage $\frac{\varepsilon}{2}$.*

Proof Sketch. The security proof of B-CP-ABE can be used as a base for the security proof for E-CP-ABE scheme. The security-game played between the simulator $\mathcal{B}$ and the adversary $\mathcal{A}$ is the same as in Section 7.5 with a small difference in the generation of the challenge ciphertext components where $\mathcal{B}$ uses a different approach to assign shares to leave nodes. This is necessary because the AC policy contains an additional operator, i.e. the *of* (threshold) operator. The simulation of this part of the encryption is as follows:

The simulator $\mathcal{B}$ sets the value of the root node of τ^* to be g^c, it marks all child nodes as un-assigned, and marks the root node assigned. Recursively, for each un-assigned non-leaf child node do the following:

- If the symbol is *of* (threshold operator), $\mathcal{B}$ assigns $g^{f(i)}$ to each child node, where $f(i)$ is a polynomial of degree $t-1$, t is the number of child nodes to reconstruct the secret, i is the index (order) of the attributes in τ^* and $f(0) = c$. Mark this node assigned.
- If the symbol is $\wedge$, $\mathcal{B}$ assigns $g^{f(i)}$ to each child node, where $f(i)$ is a polynomial of degree $\mathbf{n}-1$, $\mathbf{n}$ is the total number of the child nodes, i is the index (order) of the attributes in τ^* and $f(0) = c$. Mark this node assigned.
- If the symbol is $\vee$, $\mathcal{B}$ assigns $g^{f(i)}$ to each child node, where $f(i)$ is a polynomial of degree 0, i is the index (order) of the attributes in τ^* and $f(0) = c$. Mark this node assigned.

For each leaf attribute $a_{j,i} \in \tau$, compute $c_{j,i} = g^{f(i)k_j}$. Since we have the same probability measures as in Section 7.5, we define the overall advantage of $\mathcal{B}$ in solving the DBDH assumption to be: $\frac{\varepsilon}{2}$. □

4.6 Updates

Updating User's Attribute Set

In CP-ABE granting or revoking an attribute from a user is a challenging task. Revocation is difficult since there is no way to prevent the user from using the issued attribute secret key since the attribute is not connected solely with one user. Pirretti *et al.* [83] propose to use time framed attributes where each attribute would be valid for a specific time frame. However, this would require the trusted authority to refresh the list of attributes regularly. In Chapter 5, we provide a new CP-ABE scheme which supports revocation without requiring the trusted authority to update the attribute list.

Granting additional attributes is less difficult than revoking. There are two options for granting attributes. One option is to keep a list of users and their corresponding random values r generated during KeyGen phase. The trusted authority needs the random variable r to update the attribute set for each user, since for each attribute a_j the KeyGen algorithm computes $d_j = g^{rt_j^{-1}}$. Another option, which would not require maintaining a list of users, is to do everything from the beginning, namely to issue secret keys again for each attribute for the updated user attribute set.

Updating the AC policy

In a CP-ABE scheme the encryptor may update his AC policy without entirely decrypting the ciphertext. Suppose the user wants to update his AC policy from $\tau_{Data} =$ (Doc.A $\wedge$ Dep.A) $\vee$ (Doc.B $\wedge$ Dep.B) represented in Figure 4.2 to a different AC policy $\tau' =$ (Doc.A $\wedge$ Dep.A) $\vee$ $(T_5 \vee T_6)$.

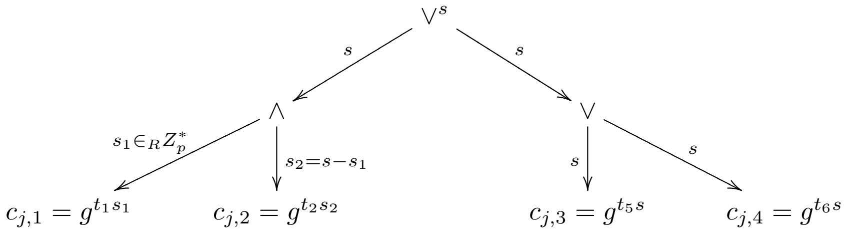

Figure 4.5: *Generating ciphertext components according to the AC policy* $\tau' =$ (Doc.A $\wedge$ Dep.A) $\vee$ $(T_5 \vee T_6)$.

Recall from B-CP-ABE constructions, to encrypt a message $m \in \mathbb{G}_T$ under τ_{Data}, Encrypt selects a random element $s \in \mathbb{Z}_p$ and it computes: $c_0 = g^s$ and $c_1 = m \cdot y^s = m \cdot e(g,g)^{\alpha s}$. The other components of the ciphertext depend on τ_{Data}: $\forall a_{j,i} \in \tau_{Data} : c_{t_{1,1}} = T_1^{s_1}$, $c_{t_{2,2}} = T_2^{s_2}$, $c_{t_{3,3}} = T_3^{s_3}$, $c_{t_{4,4}} = T_4^{s_4}$. The final ciphertext is $c_{\tau_{Data}} = (\tau_{Data}, c_0, c_1, \forall a_{j,i} \in \tau : c_{t_{1,1}}, c_{t_{2,2}}, c_{t_{3,3}}, c_{t_{4,4}})$. To update the privacy from τ_{Data} to τ', there is no reason to modify c_0 and c_1, since the $c_{j,i}$'s component of the ciphertext enforces the AC policy. Therefore, the updated ciphertext will be different only at $c_{t_{3,3}}$ and $c_{t_{4,4}}$. The new ciphertext elements are: $\forall a_{j,i} \in \tau' : c_{t_{1,1}} = T_1^{s_1}$, $c_{t_{2,2}} = T_2^{s_2}$, $c_{t_{5,3}} = T_5^s$, $c_{t_{6,4}} = T_6^s$. In Figure 4.5 we illustrate how to generate ciphertext components according to the AC policy $\tau' = (\mathsf{Doc.A} \wedge \mathsf{Dep.A}) \vee (T_5 \vee T_6)$. We assume that g^{t_1} is assigned to the attribute $\mathsf{Doc.A}$, g^{t_2} is assigned to the attribute $\mathsf{Dep.A}$, g^{t_5} is assigned to the attribute T_5 and g^{t_6} is assigned to the attribute T_6. In a similar way we can show the update of AC policies in E-CP-ABE.

Updating the AC policy without totally decrypting the ciphertext, requires the user to know the random value s, which is chosen in Encrypt. This is the trade-off that the encryptor has to accept since it has to keep a list of random variables used during each encryption. We solve this problem in Chapter 6, where we present a new construction which enables users that satisfy the AC policy to update the encrypted data without having to remember any random variable.

4.7 Conclusion

In this chapter we present two CP-ABE schemes: B-CP-ABE and E-CP-ABE. In B-CP-ABE, the encryptor specifies the policy in the encryption phase using an n-ary tree which consists from $\vee$ and $\wedge$ nodes. Then, we show a modified version of B-CP-ABE, which we call E-CP-ABE, which is more expressive compared to B-CP-ABE but uses threshold secret sharing. In E-CP-ABE scheme, the policy is expressed as an n-ary tree access tree which consists of $\vee$, $\wedge$ and *of* nodes. Finally, we discuss how to update a user attribute set and the AC policy of the ciphertext.

Chapter 5

Key Revocation in Attribute-Based Encryption

In the previous chapter, we proposed new Ciphertext-Policy Attribute-Based Encryption (CP-ABE) schemes that do not support attribute revocation. In this chapter we propose a mediated Ciphertext-Policy Attribute-Based Encryption (mCP-ABE) scheme with instantaneous attribute revocation. In mCP-ABE only users who have a secret key associated with a set of attributes which satisfy the access control policy and whose *attributes are not revoked* will be able to decrypt the ciphertext.

We start with a motivation about attribute revocation and briefly review the related work. We continue by giving a formal definition for the mCP-ABE scheme along with its security definition. In Section 5.3 we introduce the construction of mCP-ABE. In Section 5.4 we apply the mCP-ABE scheme and describe a general architecture for secure management of Personal Health Records (PHRs). The last section concludes the chapter. This chapter extends the work of a refereed paper [4].

5.1 Introduction

Modern distributed information systems require flexible access control (AC) models which go beyond discretionary, mandatory and role-based AC. Recently proposed models, such as attribute-based AC, define AC policies based on different attributes of the requester, environment, or the data object. On the other hand, the current trend of service-based information systems and storage outsourcing require increased protection of data including AC methods that are cryptographically enforced. The concept of Attribute-Based Encryption (ABE) fulfills the aforementioned requirements. As described in Chapter 4, ABE provides an elegant way of encrypting data

such that the encryptor defines the attribute set that the decryptor needs to posses in order to decrypt the ciphertext.
The state-of-the-art CP-ABE schemes provide limited support for key revocation, a feature, which is becoming increasingly important in modern AC systems. In CP-ABE a user secret key is associated with a set of attributes. In general, secret key revocation may happen when: a) an attribute is not valid because it has expired, b) when a user misuses her secret key by giving it to unauthorized users or c) when a user loses her secret key.

When we talk about key revocation in CP-ABE, we define the following cases: i. revoking a subset of attributes from a user secret key without influencing other users, ii. revoking all the attribute set for a user secret key without influencing other users and iii. revoking a subset of attributes from every secret key. Based on these cases we observe that in CP-ABE we can have a partial key revocation (such as i. and iii.) when a user still can use some of her attributes, or a total key revocation (such as ii.) when a user cannot use at all her secret key. These cases are different from a traditional public-key encryption (PKE) where the key revocation is all-or-nothing and there is only one key per user to be revoked.

Our Contributions

In this chapter, we propose a new scheme for key revocation in CP-ABE, called mediated Ciphertext-Policy Attribute-Based Encryption (mCP-ABE). Previous CP-ABE systems propose to use a system where secret keys are valid within a specific time frame [83]. However, the drawback of this approach is that there is no way to revoke a key before the expiration date.

In our scheme the secret key consists of two components, one component for the mediator and the other one for the user. To decrypt the data, the user must contact the mediator to receive a decryption token. The mediator keeps an attribute revocation list (R) and refuses to issue the decryption token for revoked attributes. Without the token, the user cannot decrypt the ciphertext, therefore the attribute is implicitly revoked. Decryption tokens can be used only once and depend on the randomness used to produce the ciphertext (i.e. tokens depend on the ciphertext). In our scheme we assume that each user has a unique identifier I_u and may have many attributes. Different users having different identifiers, may have the same attribute set. For example, Alice with an identifier I_{Alice}, and Bob with an identifier I_{Bob}, may have the same attribute set $\omega = (att_1, att_2)$. The identifier is used by the mediator to check whether there are revoked attributes related to a specific user without affecting other users.

5.1.1 Related Work

Mediated Cryptography. Boneh *et al.* [26, 25] have introduced a method for fast revocation of public key certificates and security capabilities in a RSA cryptosystem called mediated RSA (mRSA). The method uses an online semi-trusted mediator (SEM) which has a share of each user's secret key, while the user has the remaining share of the secret key. To decrypt or sign a message, a user must first contact SEM and receive a message-specific token. Without the token, the user cannot decrypt or sign a message. Instantaneous user revocation is obtained by instructing SEM to stop issuing tokens for future decrypt/sign requests. Thus, in mediated cryptography the Trusted Authority (TA) responsible to generate a user key pair, does not deliver the full decryption key to users, but it delivers only a share of it. This method achieves faster revocation of user's security capabilities compared to previous certification techniques such as Certificate Revocation List (CRL) and Online Certificate Status Protocol (OCSP). Libert and Quisquater [66] have shown that the architecture for revoking security capabilities can be applied to several existing schemes including the Boneh-Franklin IBE scheme [27]. Nali *et al.* [75] have presented a mediated hierarchical identity-based encryption (IBE) and signature scheme. The hierarchical nature of the schemes and the instant revocation capability offered by the SEM architecture allows to enforce access control cryptographically in hierarchically structured communities of users whose access privileges change dynamically. Nali *et al.* [74] also have shown how to extend the Libert and Quisquater scheme to allow the enforcement of role-based AC.

Broadcast ABE. In a broadcast encryption scheme, the sender (broadcaster) sends a ciphertext to a group of recipients such that only non-revoked users inside the group can decrypt the broadcasted content. Such a scheme allows the broadcaster to specify the list of revoked users who are not allowed to decrypt the digital content that is broadcasted. Revocation of users in broadcast encryption schemes have achieved significant attraction over the year as the revocation of the user is necessary and inevitable. For example, when broadcasting encrypted TV programs only subscribed users should be able to decrypt the content, while unsubscribed users (i.e. revoked users) should not be able to decrypt the content. In a broadcast CP-ABE scheme the encryptor encrypts the data according to the AC policy τ and the list of the identities of revoked users R. Only users with the attribute set that satisfy the AC policy τ and their identities is not in R would be able to decrypt the ciphertext.

The revocable broadcast encryption schemes can be divided into two broad categories: a) tree based revocation schemes and, b) secret sharing revocation schemes. The tree based revocation schemes were first proposed by Naor *et al.* [76], where they present the *subset cover* framework. The proposed framework assigns the user to leaves of the tree, such that they belong to different subsets. The encryptor encrypts the data to the minimum disjoint subsets such that only non-revoked users are covered. The second categories of revocable broadcast encryption schemes are based on the secret sharing (or polynomial interpolation) in the exponents of group elements, given by Kusrosawa and Desmedt [62] and Naor and Pinkas [77]. In the proposed systems, a polynomial of degree r is selected at setup phase, where r is the maximum number of the users that can be revoked. These systems can be extended to handle n revoked user by using $\log(n)$ parallel systems with the private key and message length of $O(r)$ and public key size of $O(n)$[106]. Both schemes in a) and b) are restricted in the sense that the schemes work only for a single broadcaster, where only a single entity (broadcasting center) can broadcast messages to a group of users.

Ostrovsky *et al.* [82] proposed an ABE with non-monotonic access structures. Hence to negate an attribute in an access structure one applies revocation scheme using attribute as an identity to be revoked. Ostrovsky *et al.* gives a particular example by adjusting the revocation scheme proposed by Naor and Pinkas [77] for the broadcast encryption scenarios. The main drawback of the scheme is that the private key size blows up by a multiplicative factor of $\log(n)$, where n represents the maximum number of attributes in the system.
Lubiz and Sirvent [67] proposed a broadcast encryption scheme in the context of ABE. As the authors point out, the main drawback of the scheme is the expressivity of the AC policies associated with the ciphertext, since the scheme supports only AC policies expressed by the Boolean operator $\wedge$.

The difference between the work presented in this chapter and *Broadcast* ABE is that in our work the revocation list $\mathcal{R}$ is not defined by the encryptor, hence the Encrypt algorithm is not influenced by, and does not get as input, the revocation list $\mathcal{R}$. In our case the revocation list $\mathcal{R}$ is maintained by the mediator and either the trusted authority or the secret key holders can add entries to it.

Revocation in ABE. The paper close to our setting is presented by Pirretti *et al.* [83]. In their system an attribute can be used for a limited period of time - after a specific time the attribute would become invalid. However, the drawback of this approach is that an attribute cannot be revoked before the expiration date. This approach also requires the list of secret keys that correspond to attributes to be updated regularly.

5.2 Mediated CP-ABE (mCP-ABE)

The mCP-ABE scheme consists of three entities: a trusted authority (TA) (also know as a Key-Generation Center), a mediator and users. The TA uses the master key to generate a user secret key, which consists of two component: one component is sent to the mediator and the other component is sent to a user. The mediator has to stay online all the time, while the TA can go off-line once it has generated secret keys for all users. The mCP-ABE scheme consists of five algorithms: Setup, KeyGen, Encrypt, m-Decrypt, and Decrypt, which are defined as follows:

- Setup and Encrypt are defined respectively in the same way as Setup and Encrypt of the CP-ABE defined in Section 4.3.

- $\mathsf{KeyGen}(\mathsf{msk}, \omega, I_u)$: run by the TA, this algorithm takes as input the master key $\mathsf{msk} \in \mathcal{K}$, the user attribute set $\omega \subseteq \Omega$, and the unique user identifier I_u. The algorithm outputs two secret key components associated with ω and I_u : $sk_{\omega I_u,1} \in \mathcal{K}$ and $sk_{\omega I_u,2} \in \mathcal{K}$. The first component $sk_{\omega I_u,1} \in \mathcal{K}$ is delivered to the mediator and the second component $sk_{\omega I_u,2} \in \mathcal{K}$ is delivered to the user. The secret key components are delivered through a secure channel.

- $\mathsf{m\text{-}Decrypt}(c_\tau, I_u, sk_{\omega I_u,1}, \mathsf{R})$: run by the mediator, this algorithm takes as input a ciphertext $c_\tau \in \mathcal{C}$, an identifier I_u and the secret key $sk_{\omega I_u,1} \in \mathcal{K}$, and outputs a ciphertext $\hat{c}_\tau \in \mathcal{C}$ if $\omega \subseteq \Omega$ satisfies τ and there are no attributes from ω in

R. The algorithm returns an error symbol $\perp$ if $\omega \subseteq \Omega$ does not satisfy the AC policy τ or if $\omega \subseteq \Omega$ satisfies the AC policy τ but it contains revoked attributes that are listed in R.

- Decrypt($\hat{c}_\mathcal{T}, sk_{I_u\omega,2}$): run by the message receiver, this algorithm takes as input a ciphertext $\hat{c}_\mathcal{T} \in \mathcal{C}$, and a secret key $sk_{I_u\omega,2} \in \mathcal{K}$, and outputs a message $m \in \mathcal{M}$, or an error symbol $\perp$ when $\omega \subseteq \Omega$ does not satisfy the AC policy τ.

In practice, there might be multiple entities acting as mediators, and a global entity acting as TA. For example, a healthcare organization may choose Proxy$_1$ as its mediator and a government organization may choose Proxy$_2$ as its mediator, where each mediator has the first component of the secret key for registered users in the hospital organization, respectively in the government organization. Vanrenen *et al.* [100] propose the use of peer-to-peer networking (P2P) which would allow users to require a decryption token from every mediator, such that the mediator either tries to compute a decryption token by itself, or forwards the request to its neighbors.

Correctness. We call a mCP-ABE *correct* if first decrypting a ciphertext $c_\mathcal{T}$ with a secret key $sk_{\omega I_u,1}$, and then decrypting a ciphertext $\hat{c}_\mathcal{T}$ with a secret key $sk_{\omega I_u,2}$, such that the attribute set ω does not have attributes in R and satisfies the AC policy τ, always results in the same message m as was input by the Encrypt algorithm when it produced the ciphertext $c_\mathcal{T}$. More precisely we show that if the attribute set ω satisfies the AC policy τ then:

$$\Pr\left[\begin{array}{c}(\mathsf{msk}, \mathcal{PP}) \leftarrow \mathsf{Setup}(\lambda), \quad (sk_{\omega I_u,1}, sk_{\omega I_u,2}) \leftarrow \mathsf{KeyGen}(\mathsf{msk}, \omega, I_u), \\ c_\mathcal{T} \leftarrow \mathsf{Encrypt}(m, \tau), \quad \hat{c}_\mathcal{T} \leftarrow \mathsf{m\text{-}Decrypt}(c_\mathcal{T}, I_u, sk_{\omega I_u,1}, \mathsf{R}) \ : \\ m \leftarrow \mathsf{Decrypt}(\hat{c}_\mathcal{T}, sk_{I_u\omega,2})\end{array}\right] = 1 \ .$$

5.2.1 Security Definitions

In our scheme the TA is a fully trusted entity which securely stores the master key. We skip the discussions about the key escrow problem, since different existing threshold schemes [88, 42] can be applied to mitigate this problem. A mediator is a semi-trusted entity, namely, it should issue decryption tokens to users, but it is not trusted in the sense that it might be curious about the plaintext. We define indistinguishability under chosen-plaintext attack (IND-CPA) following the security model of Libert and Quisquater [66].

Definition. *The* mCP-ABE *scheme is said to be indistinguishable under chosen-plaintext attacks* (IND-CPA) *if any polynomial-time adversary $\mathcal{A}$ has only a negligible advantage in the* mCP-ABE *security-game defines as follows:*

- Setup. *The challenger runs the* Setup *algorithm to generate* ($\mathcal{PP}$, msk) *and gives the public parameters $\mathcal{PP}$ to the adversary $\mathcal{A}$.*

- Phase1. *$\mathcal{A}$ performs a polynomially bounded number of queries:*

  - KeyGen^1. *$\mathcal{A}$ asks for a secret key for the attribute set ω and identifier I_u, and receives the mediator component of the secret key $sk_{\omega I_u,1}$.*
  - KeyGen^2. *$\mathcal{A}$ asks for a secret key for the attribute set ω and identifier I_u, and receives the user component of the secret key $sk_{\omega I_u,2}$.*

- Challenge. *$\mathcal{A}$ sends to the challenger two messages m_0, m_1, such that $|m_0| = |m_1|$, and the challenge AC policy τ^*. $\mathcal{A}$ is restricted in his queries such that it should not have obtained both secret key components $sk_{\omega I_u,1}$ and $sk_{\omega I_u,2}$ associated with the attribute set ω that satisfies τ^*. Note that $\mathcal{A}$ is allowed to obtain either $sk_{\omega I_u,1}$ or $sk_{\omega I_u,2}$.*

  The challenger picks a random bit $b \in \{0,1\}$ and returns $c_{\tau^*} = \mathsf{Encrypt}(m_b, \tau^*)$.

- Phase2. *$\mathcal{A}$ can continue querying with the same restriction as in the* Challenge *phase.*

- Guess. *$\mathcal{A}$ outputs a guess $b' \in \{0,1\}$, and wins the game if $b' = b$.*

The advantage of $\mathcal{A}$ in winning security-game is defined as:

$$\mathsf{ADV}^{\mathsf{IND-CPA}}_{\mathcal{A},\mathsf{mCP-ABE}}(\lambda) \overset{def}{=} \left| \Pr[b' = b] - \frac{1}{2} \right| ,$$

where the probability is over the random values chosen by $\mathcal{A}$ and the challenger.

The definition formally captures the following security requirements:

- Resistance against secret key collusion, where different users cannot combine their attribute sets to extend their decryption power. For example, suppose there is a message encrypted under the AC policy $\tau = a_1 \wedge a_2 \wedge a_3$. Suppose Alice has a secret key $sk_{\omega_A I_A}$ associated with an attribute set $\omega_A = (a_1, a_2)$, and Bob has a secret key $sk_{\omega_B I_B}$ associated with an attribute set $\omega_B = (a_3, a_4)$. Neither Alice's secret key, nor Bob's secret key satisfies the AC policy τ. But, if Alice and Bob combine their attribute sets $\omega_A \cup \omega_B = (a_1, a_2, a_3, a_4)$, then the combined attribute sets satisfies the AC policy τ. In the security-game therefore we allow the adversary to make secret key queries associated with different attribute sets, say ω_1 and ω_2, such that neither ω_1, nor ω_2 alone can satisfy the challenge AC policy τ^*, but $\omega_1 \cup \omega_2$ can satisfy τ^*.

- Resistance against malicious cooperation between the mediator and some users to decrypt the ciphertext associated with an AC policy, when a user's secret key does not satisfy the AC policy. For example, even if a user with attribute set $\omega = (a_1, a_2)$ colludes with the mediator, the user should not be capable to decrypt a ciphertext encrypted under a challenge AC policy $\tau^* = (a_1 \wedge a_2 \wedge a_3)$, since ω does not satisfy τ^*. Therefore in the security-game the adversary is allowed to ask the mediator component (first component of the secret key $sk_{\omega I_u,1}$) and the user component of a secret key (second component of the secret key $sk_{\omega I_u,2}$) for any set of attributes which does not satisfy the challenge AC policy τ^*.

Conceptual differences with the security definition of CP-ABE presented in Chapter 4

The security definition of the CP-ABE scheme presented in Chapter 4 is weaker than this definition. In Chapter 4, we use a security notion known as *selective security.* In selective security, the adversary must commit to the challenge AC policy τ^* at the initial phase of the security-game. Then, in the challenge phase, the adversary has to commit to two messages only: m_0 and m_1. Finally, the challenger picks at random one of the messages and encrypts it according to τ^*. The crux of the selective security proof is that the simulator uses τ^* to generate public parameters $\mathcal{PP}$ and is able to answer all adversary queries.

In this chapter we use a stronger security notion known as *full security.* With full security, the adversary commits to the challenge AC policy τ^* in the challenge phase, at the same time when it commits to m_0 and m_1. The difficulty of the full security proof is that the challenger (in the security game the challenger is named as simulator) does not know at the beginning of the security-game the AC policy τ^* the adversary is going to commit to. Hence, the challenger has to generate public parameters $\mathcal{PP}$ without knowing τ^* and should still be able to answer all adversary queries. The full security notion better reflects a real world adversaries' behavior than the selective security notion. However, this comes at a price. Unlike in Chapter 4 where we give a security proof in the standard model, in this chapter we are able to prove the security of the mCP-ABE scheme only in the generic group model (see Sections 2.4 and 2.5 for the differences between the standard model and the generic group model). The generic group model is used as a tool to prove the security of other CP-ABE schemes [20].

5.3 Construction of mCP-ABE

Intuition. Before introducing the scheme, we give some intuition about its construction. The construction is based on the B-CP-ABE schemes presented in Chapter 4 and re-uses its Setup and Encrypt algorithms. The main technical challenge when constructing mCP-ABE is to generate the secret key in such a way that users among them should not be able to collude, but they should "collude" with the mediator in order to receive a decryption token. Therefore, in KeyGen for a single user we generate two secret key components that are randomized using the same random value. One secret key component is delivered to the mediator and the other to the user.

In KeyGen users are identified with a unique identity I_u and an attribute set ω. Unlike in CP-ABE, in mCP-ABE we need the unique identity I_u to separate users who have the same attribute set. Note that, Alice with an identifier I_{Alice}, and Bob with an identifier I_{Bob}, may have the same attribute set $\omega = (att_1, att_2)$. The identifier I_u helps the mediator to revoke attributes from one user without affecting the attribute set of the other users. For example, the mediator can revoke the attribute att_1 from Bob without affecting Alice's attribute set.

Key revocation happens in the m-Decrypt phase. This is the phase when the user requests a decryption token. When generating decryption tokens, the mediator constructs them such that they can be only used once (i.e. to make them useless for users when they get revoked). Therefore, m-Decrypt produces decryption tokens that are ciphertext-specific and depend on the randomness generated in the Encrypt phase.

As already mentioned in Section 5.1, there can be many reasons why an attribute can be revoked. We assume that the mediator maintains an attribute revocation list (R) which simply has information about attributes revoked from a universe (set) of attributes Ω, and attributes revoked from a user attribute set ω. The basic idea is that, when an attribute a_j is revoked from Ω, the TA notifies the mediator to stop generating decryption tokens for all users whose attribute secret key involves a_j. When an attribute a_j is revoked from ω, which is associated with an identity I_u, the TA notifies the mediator to stop generating decryption tokens that include a_j for the user I_u. Therefore, the attribute revocation is achieved immediately after the revocation decision is made. Note that a user with an identity I_u can request decryption tokens for attributes other than a_j. We assume that there is a policy of revocation authorization maintained by the mediator that describes who is responsible to revoke attributes from Ω or ω. At least, the TA should be able to revoke any attribute (from Ω and ω), and the owner of the attribute should be able to revoke its attributes (i.e. be able to revoke from ω) because the owner may be the first to notice the compromise of her secret key.

The Scheme. We now present our mCP-ABE scheme.

- Setup(λ) : is the same as the Setup of the B-CP-ABE scheme presented in Chapter 4.

- KeyGen$(\mathsf{msk}, \omega, I_u)$: To generate a secret key for the user with an attribute set ω and an identifier I_u, the KeyGen algorithm works as follows:

  - Compute the base component of the secret key: $d_0 = g^{\alpha - u_{id}}$ where $u_{id} \in_R \mathbb{Z}_p$ (for each user with an identifier I_u a unique random value u_{id} is generated).
  - Compute the attribute component of the secret key. For each attribute $a_j \in \omega$, choose $u_j \in_R \mathbb{Z}_p$ and compute:

  $$d_{j,1} = g^{\frac{u_j}{t_j}} \qquad , \qquad d_{j,2} = g^{\frac{u_{id} - u_j}{t_j}} \quad .$$

  The secret key of the form: $sk_{\omega I_u,1} = \{d_{j,1}\}_{a_j \in \omega}$ is delivered to the mediator, and the secret key of the form: $sk_{\omega I_u,2} = \big(d_0, \{d_{j,2}\}_{a_j \in \omega}\big)$ is delivered to the user. As it will be shown later, the user will need the help of the mediator to use his attribute component of the secret key in the decryption phase. The denial of help for an attribute, would imply the revocation of that attribute.

- Encrypt(m, τ) : is the same as the Encrypt of the B-CP-ABE scheme presented in Chapter 4.

- $\mathsf{m\text{-}Decrypt}(c_\mathcal{T}, I_u, sk_{\omega I_u,1}, \mathsf{R})$: when receiving the ciphertext $c_\mathcal{T}$, the recipient I_u first chooses the smallest set $\omega' \subseteq \omega$ that satisfies τ and forwards to the mediator $(c_\mathcal{T}, \omega', I_u)$. The mediator checks the Attribute Revocation List (R) if any $a_j \in \omega'$ is revoked either from system attribute set Ω or from the user associated with the identifier I_u.
  - If an attribute is revoked, the mediator returns an error symbol $\perp$ and does not perform further computations.
  - If no attribute is revoked, the mediator computes $\hat{c}_\mathcal{T}$ as follows:

    $$\hat{c}_\mathcal{T} = \prod_{a_j \in \omega'} e\left(T_j^{s_i}, g^{\frac{u_j}{t_j}}\right)$$

    and sends the token $\hat{c}_\mathcal{T}$ to the recipient.

- $\mathsf{Decrypt}(\hat{c}_\mathcal{T}, sk_{\omega I_u,2})$: To decrypt the ciphertext the recipient proceeds as follows:
  - compute:

    $$c''_\mathcal{T} = \prod_{a_j \in \omega'} e\left(T_j^{s_i}, g^{\frac{u_{id}-u_j}{t_j}}\right)$$

  - return m, where:

    $$m = \frac{c_1}{e\left(c_0, d_0\right) \cdot \hat{c}_\mathcal{T} \cdot c''_\mathcal{T}} \ .$$

Note. For the sake of simplicity, we use only AC policies with $\wedge$ (and) and $\vee$ (or). However, our scheme, in addition to $\wedge$ and $\vee$ nodes, can support threshold nodes nodes. For example, the encryptor may specify the AC policy 2 *of* (a_1, a_2, a_3), which implies that the user must have at least two attributes to satisfy the AC policy and to be able to decrypt. If the AC policy contains threshold nodes, then the attribute shares s_i can be generated using Shamir's secret sharing in the same way as in the $\mathsf{Encrypt}$ of the $\mathsf{E\text{-}CP\text{-}ABE}$ scheme presented in Chapter 4. This would require to use Lagrange coefficients in $\mathsf{Decrypt}$ in order to reconstruct the value s.

Correctness. We show that $\mathsf{Decrypt}$ returns the message m on input: a) the secret key $sk_{\omega I_u,2}$, which is created as a result of running $\mathsf{KeyGen}(\mathsf{msk}, \omega, I_u)$ and b) the ciphertext $\hat{c}_\mathcal{T}$, which is created as a result of running $\mathsf{m\text{-}Decrypt}(c_\mathcal{T},\ I_u, sk_{\omega I_u,1}, \mathsf{R})$, where $c_\mathcal{T}$ is created as a result of running $\mathsf{Encrypt}(m, \tau)$.

Let $c_\mathcal{T} = \left(\tau, c_0 = g^s, c_1 = m \cdot e(g,g)^{\alpha s}, \{c_{j,i} = g^{t_j s_i}\}_{a_{j,i} \in \tau}\right)$ be a ciphertext generated by $\mathsf{Encrypt}(m, \tau)$ and let:

$$sk_{\omega I_u,1} = \{d_{j,1} = g^{\frac{u_j}{t_j}}\}_{a_j \in \omega} \quad , \quad sk_{\omega I_u,2} = \left(d_0 = g^{\alpha - u_{id}}, \{d_{j,2} = g^{\frac{u_{id}-u_j}{t_j}}\}_{a_j \in \omega}\right)$$

Table 5.1: *Efficiency of* mCP-ABE.

	Exp.($\mathbb{G}$)	Exp.($\mathbb{G}_T$)	Pairing
KeyGen	$2\|\omega\|+1$	-	-
Encrypt	$\|\tau\|+1$	1	-
m-Decrypt	-	-	$\|\omega'\|$
Decrypt	-	-	$\|\omega'\|+1$

be secret key components generated by $\mathsf{KeyGen}(\mathsf{msk}, \omega, I_u)$ such that the attribute set ω satisfies the AC policy τ. Let $\omega' \subseteq \omega$ be the smallest subset which satisfies τ. We observe that:

$$\hat{c}_\tau = e\left(g,g\right)^{\sum_{a_j \in \omega'} u_j s_i}$$

and

$$\begin{aligned} c''_\tau &= \prod_{a_j \in \omega'} e\left(g^{t_j s_i}, g^{\frac{u_{id}-u_j}{t_j}}\right) \\ &= e\left(g,g\right)^{\sum_{a_j \in \omega'} (u_{id}-u_j)s_i} . \end{aligned}$$

To output m, Decrypt computes:

$$\begin{aligned} & \frac{c_1}{e\left(c_0, d_0\right) \cdot \hat{c}_\tau \cdot c''_\tau} \\ = & \frac{m \cdot e(g,g)^{\alpha s}}{e\left(g^s, g^{\alpha - u_{id}}\right) \cdot e\left(g,g\right)^{\sum_{a_j \in \omega'} u_j s_i} \cdot e\left(g,g\right)^{\sum_{a_j \in \omega'} (u_{id}-u_j)s_i}} \\ = & \frac{m \cdot e(g,g)^{\alpha s}}{e\left(g^s, g^{\alpha - u_{id}}\right) \cdot e\left(g,g\right)^{u_{id} s}} \\ = & \frac{m \cdot e\left(g,g\right)^{\alpha s}}{e\left(g^s, g^{\alpha}\right)} \\ = & m. \end{aligned}$$

5.3.1 Efficiency Analysis

In Table 5.1 we count the number of calculations in mCP-ABE. In mCP-ABE, the size of the components of the secret key $sk_{\omega I_u,1}$ and $sk_{\omega I_u,2}$ depend on the number of attributes in the set ω. Thus, KeyGen requires in total $2|\omega|+1$ exponentiations in $\mathbb{G}$. The size of the ciphertext c_τ depends on the size of the AC policy τ and has $|\tau|+1$ group elements in $\mathbb{G}$, and one group element in $\mathbb{G}_T$. Thus, Encrypt requires $|\tau|+1$ exponentiations in $\mathbb{G}$ and one exponentiation in $\mathbb{G}_T$. In the m-Decrypt phase, the mediator has to compute $|\omega'|$ pairing operations, where $\omega' \subseteq \omega$ is the attribute set which satisfies the AC policy τ. In Decrypt, to reveal the message, the user has to compute $|\omega'|+1$ pairing operations.

5.3.2 Security Proof

We provide a brief informal explanation about the security of the proposed scheme. A full formal security proof starts from Theorem 4 and closely follows the security proof given by Bethencourt *et al.* [20].

To decrypt a ciphertext without satisfying the AC policy, the adversary has to construct $e(g^s, g^\alpha)$, and then divide c_1 with $e(g^s, g^\alpha)$ to obtain m. To obtain $e(g^s, g^\alpha)$, the adversary must first obtain $e(g,g)^{su_{id}}$, which can be calculated by pairing the components of the secret key $g^{\frac{u_{id}-u_j}{t_j}}$ with the components of the ciphertext $g^{t_j s_i}$, and then multiply the result with the decryption token $e(g,g)^{\sum_{a_j \in \omega'} u_j s_i}$ received from the mediator. However, $e(g,g)^{su_{id}}$ can be computed only if the adversary has enough attributes which satisfy the AC policy, otherwise this would not be possible. Also note that, if a user acting as an adversary is revoked, then the user will not get the decryption token $e(g,g)^{\sum_{a_j \in \omega'} u_j s_i}$ from the mediator, and as a result the user cannot reconstruct $e(g,g)^{su_{id}}$ even if the user has a secret key associated with the attribute set which satisfies the AC policy.

If we assume that the adversary is able to compromise the mediator but not a legitimate user, then the adversary will be able to learn the mediator component of the user secret key $sk_{\omega I_u,1}$, and will be able to compute the decryption token $e(g,g)^{\sum_{a_j \in \omega'} u_j s_i}$. However, the decryption token will not help the adversary to decrypt the ciphertext without having $sk_{\omega I_u,2}$.

Theorem 4. *Let γ_0 and γ_1 be a random encoding for the group $\mathbb{G}$ and $\mathbb{G}_T$, respectively. A random encoding maps elements of the additive group $\mathbb{Z}_p$ into a bit string. The advantage of the adversary in the* mCP-ABE *security-game issuing at most q queries to the oracles for computing: i) a group operation in $\mathbb{G}$ and $\mathbb{G}_T$, ii) a pairing operation e, iii) a key generation, and iv) an encryption, is bounded by $O(q^2/p)$.*

Proof. Consider groups $\mathbb{G}$ and $\mathbb{G}_T$, and generators g of the group $\mathbb{G}$ and $e(g,g)$ of the group $\mathbb{G}_T$. In generic group model, group elements are encoded as unique random strings, in such a way that the adversary $\mathcal{A}$ cannot test any property other than equality. In our proof, we use γ_0 as a random encoding for the group $\mathbb{G}$ (generic bilinear group), and γ_1 as a random encoding for the group $\mathbb{G}_T$. Thus, for example, the group element $g^s \in \mathbb{G}$ will be encoded as $\gamma_0(s)$, and the group element $e(g,g)^\alpha \in \mathbb{G}_T$ will be encoded as $\gamma_1(\alpha)$. Each random encoding is associated with a rational function $f = \frac{\xi}{\varpi}$ over the variables:

$$\Upsilon = \{\alpha, s, s_i, u_{id}, \{u_j\}_{a_j \in \omega}, \{t_j\}_{1 \le j \le n}\},$$

where each variable is an element picked at random in the scheme.

We now give the simulation of the security-game. Following the proof from [20], in the simulation we slightly modify the security-game given in Section 5.2.1, and simulate a game in which the component c_1 of the challenge phase is either $\gamma_1(\alpha s)$ or $\gamma_1(\theta)$, where $\theta \in_R \mathbb{Z}_p$, and $\mathcal{A}$ has to decide whether $c_1 = \gamma_1(\alpha s)$ or $c_1 = \gamma_1(\theta)$. It can be easily shown that if there is an $\mathcal{A}$ who has a negligible advantage in a modified

game, then there is an algorithm who has a negligible advantage in the security-game given in Section 5.2.1.
In the security game, the simulator maintains a table L_1 to store information about the values generated from the interaction of $\mathcal{A}$ with the KeyGen^1 and KeyGen^2 oracles. The security-game is simulated as follows:

- Setup. The simulator chooses a group $\mathbb{G}$ of prime order p with a generator g and a bilinear map $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$, a random value $\alpha \in \mathbb{Z}_p$, and for each attribute $a_j \in \Omega$, chooses random values $t_j \in \mathbb{Z}_p$. In addition to that, the simulator chooses two encoding functions γ_0 and γ_1, and two oracles for computing group operations in $\mathbb{G}$ and $\mathbb{G}_T$, and one oracle for computing the bilinear map e. The following encodings are sent to $\mathcal{A}$:

  1. $\gamma_0(1)$ representing the generator g.
  2. $\gamma_1(1)$ representing the generator $e(g, g)$.
  3. $\gamma_1(\alpha)$ representing $e(g, g)^{\alpha}$.
  4. $\{\gamma_0(t_j)\}_{1 \leq j \leq n}$ representing $\{T_j = g^{t_j}\}_{1 \leq j \leq n}$.

- $\mathsf{Phase1}$. $\mathcal{A}$ performs a polynomially bounded number of queries:

  KeyGen^1. $\mathcal{A}$ makes a request for the first component (mediator component) of the secret key for associated with attribute set ω and an identifier I_u. The simulator checks whether L_1 already contains a record for the attribute set ω and the identifier I_u. If such record exists, the simulator fetches $d_{j,1}$ from the record and sends the encoding of $sk_{\omega I_u,1} = (\{d_{j,1}\}_{a_j \in \omega})$ to $\mathcal{A}$. If such record does not exist, the simulator chooses a random value $u_{id} \in \mathbb{Z}_p$, and for each $a_j \in \omega$ chooses a random value $u_j \in \mathbb{Z}_p$. The simulator generates the following encodings:

  1. $\gamma_0(\alpha - u_{id})$ representing $d_0 = g^{\alpha - u_{id}}$.
  2. $\{\gamma_0(\frac{u_j}{t_j})\}_{a_j \in \omega}$ representing $\{d_{j,1} = g^{\frac{u_j}{t_j}}\}_{a_j \in \omega}$.
  3. $\{\gamma_0(\frac{u_{id} - u_j}{t_j})\}_{a_j \in \omega}$ representing $\{d_{j,2} = g^{\frac{u_{id} - u_j}{t_j}}\}_{a_j \in \omega}$.

  The simulator sends the encoding of $sk_{\omega I_u,1} = (\{d_{j,1}\}_{a_j \in \omega})$ to $\mathcal{A}$, and inserts a new record with the encoding of $sk_{\omega I_u} = (d_0, \{d_{j,1}, d_{j,2}\}_{a_j \in \omega})$ into L_1.

  KeyGen^2. $\mathcal{A}$ makes a request for the second component (user component) of the secret key associated with an attribute set ω and an identifier I_u. The simulator checks whether L_1 already contains a record for the attribute set ω and the identifier I_u. If such an entry exists, the simulator sends the encoding of $sk_{I_u \omega,2} = (d_0, \{d_{j,2}\}_{a_j \in \omega})$ to $\mathcal{A}$. If such an entry does not exist, the simulator calculates the encodings of d_0, $d_{j,1,}$ and $d_{j,2}$ as explained under KeyGen^1 and updates the table L_1 with the new encoding of $sk_{\omega I_u} = (d_0, \{d_{j,1}, d_{j,2}\}_{a_j \in \omega})$.

- Challenge. $\mathcal{A}$ submits two messages m_0, $m_1 \in \mathbb{G}_T$ and the challenge AC policy τ^*. $\mathcal{A}$ is not allowed to ask for a challenge AC policy τ^* if in Phase1 has obtained secret key components $sk_{\omega I_u,1}$ and $sk_{\omega I_u,2}$ associated with the attribute set ω that satisfies τ^*. Note that $\mathcal{A}$ is allowed to obtain either $sk_{\omega I_u,1}$ or $sk_{\omega I_u,2}$.

  The simulation chooses a random $s \in \mathbb{Z}_p$, and for each $a_{j,i} \in \tau^*$ it constructs a value s_i as explained in Section 5.3. The following encodings are sent to $\mathcal{A}$:

  1. $\gamma_0(s)$ representing $c_0 = g^s$.
  2. $\gamma_1(\theta)$ representing $c_1 = e(g,g)^\theta$.
  3. $\{\gamma_0(t_j s_i)\}_{a_{j,i} \in \tau^*}$ representing $\{c_{j,i} = g^{t_j s_i}\}_{a_{j,i} \in \tau^*}$.

- Phase2. $\mathcal{A}$ can continue querying with same restrictions mentioned in the previous phase.

$\mathcal{A}$ uses the group elements received from the interaction with the simulator to perform generic group operations and equality tests. The simulator provides $\mathcal{A}$ with two oracles to compute the group operation in $\mathbb{G}$ and $\mathbb{G}_T$ and one oracle to compute the pairing operations e. $\mathcal{A}$ can make queries to perform group operations as follows:

- Queries to the oracles for group operations in $\mathbb{G}$ and $\mathbb{G}_T$: $\mathcal{A}$ asks for multiplying or dividing group elements represented with their random encodings, and associated with a rational function. The oracle returns $\gamma_0(a+b)$ or $\gamma_1(a+b)$ when $\mathcal{A}$ asks for multiplying $\gamma_0(a)$ and $\gamma_0(b)$, respectively $\gamma_1(a)$ and $\gamma_1(b)$, and returns $\gamma_0(a-b)$ or $\gamma_1(a-b)$ when $\mathcal{A}$ asks for dividing $\gamma_0(a)$ and $\gamma_0(b)$, respectively $\gamma_1(a)$ and $\gamma_1(b)$.

- Queries to the oracle for computing pairing operation e. $\mathcal{A}$ asks for pairing of group elements represented with their random encoding, and associated with a rational function. The oracle returns $\gamma_1(ab)$ when $\mathcal{A}$ asks for pairing $\gamma_1(a)$ and $\gamma_1(b)$.

We show that $\mathcal{A}$ can distinguish the simulation of the game where the challenge ciphertext is set $c_1 = \gamma_1(\theta)$ with the simulation of the game where the challenge ciphertext would have been set $c_1 = \gamma_1(\alpha s)$ with probability $O(\frac{q^2}{p})$.
First, we show $\mathcal{A}$'s view when the challenge ciphertext is $\gamma_1(\theta)$. The $\mathcal{A}$'s view can change when an "unexpected collision" happens due to the random choice of the formal variables $\Upsilon = \{\alpha, s, s_i, u_{id}, \{u_j\}_{a_j \in \omega}, \{t_j\}_{1 \leq j \leq n}\}$ chosen uniformly from $\mathbb{Z}_p$. A collision happens when two queries corresponding to two different rational functions map to the same string representation. It can be calculated that for any two distinct queries the probability of such a collision to happen is at most $O(q^2/p)$, where q is the total number of queries done by $\mathcal{A}$. We ignore this situation, since the probability of such collision is negligible.
Second, we show what would have been the $\mathcal{A}$'s view if the challenge ciphertext had been set $\gamma_1(\theta)$, when $\theta = \alpha s$. Again, $\mathcal{A}$'s view can change when a collision happens, such that the values of two different encodings coincide. In the following we show that the probability of such collision is negligible.

Table 5.2: *Possible pairing operations.*

s	α	t_j	$\alpha - u_{id}$	$\frac{u_j s}{t_j}$
$\frac{u_j}{t_j}$	$\frac{u_{id}-u_j}{t_j}$	$t_j s$	u_j	$u_j s_i$
$u_{id} - u_j$	$t_j^2 s_i$	$t_j\alpha - t_j u_{id}$	$\frac{\alpha u_j - u_{id}u_j}{t_j}$	$\frac{su_{id}-su_j}{t_j}$
$\frac{\alpha u_{id}-\alpha u_j - u_{id}^2 + u_{id}u_j}{t_j}$	$\alpha s - su_{id}$	$\alpha t_j s_i - u_{id} t_j s_i$	$\frac{u_j u_{id} - u_j^2}{t_j^2}$	$s_i u_{id} - s_i u_j$
$st_j s_i$				

We observe that $\mathcal{A}$ cannot pair $\gamma_1(\theta)$ with other elements (since γ_1 is the encoding for $\mathbb{G}_T$). However, $\mathcal{A}$ can make oracle queries to perform a group operation in $\mathbb{G}_T$. Therefore, $\mathcal{A}$ can ask to multiply θ and δ and obtain $\delta\alpha s$. Let $v_1 = \delta_1\theta$ and $v_2 = \delta_2\theta$. If we subtract v_2 from v_1 we have the following equation:

$$v_1 - v_2 = (\delta_1 - \delta_2)\theta = \delta'\theta = \delta'\alpha s$$

Therefore we say that $\mathcal{A}$ can make a query $\delta'\alpha s$. But, we will show that if $\mathcal{A}$ does not have sufficient attributes to satisfy the challenge AC policy τ^*, $\mathcal{A}$ cannot make a polynomial query which would be equal to $\delta'\alpha s$ (thus the collision cannot happen), which would prove the theorem. In Table 5.2 we list possible values that $\mathcal{A}$ can get using group elements received from the interaction with the simulator in the security-game. $\mathcal{A}$ can get these values by querying the oracle for computing pairing operation e. Next, we observe that $\mathcal{A}$ can get $\alpha s - su_{id}$ by pairing $\alpha - u_{id}$ and s. Thus, $\mathcal{A}$ can make a query to the oracle that performs a group operation in $\mathbb{G}_T$ to get $\delta'\alpha s - \delta' su_{id}$, for some δ'. To get only $\delta'\alpha s$, $\mathcal{A}$ has to combine group elements received from the interaction with the simulator and from the generic group oracles in order to cancel $\delta' su_{id}$. From Table 5.2 we can see that $\mathcal{A}$ can construct a query polynomial of the form:

$$\underbrace{\delta'\alpha s}_{A} - \underbrace{\delta' su_{id}}_{B} + \underbrace{\delta' \sum_{a_j \in \omega} u_j s_i}_{C} + \underbrace{\delta' \sum_{a_j \in \omega} (u_{id} - u_j)s_i}_{D}$$

for some δ'.
The term B can be cancelled only if the sum of C and D is equal to $\delta' su_{id}$. Therefore, $\mathcal{A}$ must have all necessary secret key components $\frac{(u_{id}-u_j)}{t_j}$ to pair them with $t_j s_i$, and all secret key component $\frac{u_j}{t_j}$ to pair them with $t_j s_i$, and later obtain su_{id} or $\delta' su_{id}$ for some constant δ'. However, this is not possible since in $\mathsf{Phase1}$ and $\mathsf{Phase2}$ $\mathcal{A}$ is not allowed to make queries to KeyGen^1 and KeyGen^2 oracles and obtain both secret key components $sk_{\omega I_u,1}$ and $sk_{\omega I_u,2}$ associated with an attribute set ω that satisfies τ^* (there must be at least one $s_i u_{id}$ which $\mathcal{A}$ cannot compute). Thus, the sum of terms C and D cannot be used to construct $\delta' su_{id}$. Therefore $\mathcal{A}$ cannot cancel term B, and as a result of this $\mathcal{A}$ cannot construct a query of the form $\delta'\alpha s$. We conclude the proof by making the following observations:

- We analyze the case when $\mathcal{A}$ has the component of a user secret key $sk_{\omega I_u,2}$ which satisfies the AC policy τ^*. We also assume that there are some attributes from ω in the revocation list R ($\mathcal{A}$ does not receive a decryption token $\sum_{a_j \in \omega} u_j s_i$ from the mediator). $\mathcal{A}$ can make a polynomial query which has the form:

$$\underbrace{\delta' \alpha s}_{A} - \underbrace{\delta' s u_{id}}_{B} + \underbrace{\delta' \sum_{a_j \in \omega} (u_{id} - u_j) s_i}_{D}$$

$$= \underbrace{\delta' \alpha s}_{A} - \underbrace{\delta' s u_{id}}_{B} + \underbrace{\delta' s u_{id}}_{D_1} - \underbrace{\delta' \sum_{a_j \in \omega} u_j s_i}_{D_2} \ .$$

  From this we can see that $\mathcal{A}$ can cancel B and D_1. However, $\mathcal{A}$ cannot cancel the term D_2 without having the decryption token $\sum_{a_j \in \omega} u_j s_i$ (note that $\mathcal{A}$ is not allowed to learn $sk_{\omega I_u,1}$). As a result, $\mathcal{A}$ cannot make a polynomial query which has the form $\delta' \alpha s$.

- We analyze the case when $\mathcal{A}$ corrupts the mediator and obtains the mediator component of the user secret key $sk_{\omega I_u,1}$ (i.e. $\mathcal{A}$ can compute the decryption token). We assume that $\mathcal{A}$ does not have the user component of the secret key $sk_{\omega I_u,2}$. $\mathcal{A}$ can make a polynomial query which has the form:

$$\underbrace{\delta' \alpha s}_{A} - \underbrace{\delta' s u_{id}}_{B} + \underbrace{\delta' \sum_{a_j \in \omega} u_j s_i}_{C} \ .$$

  As we can see $\mathcal{A}$ cannot cancel B and C, since the user component of the secret key $\sum_{a_j \in \omega} (u_{id} - u_j) s_i$ is missing. As a result, we conclude that $\mathcal{A}$ cannot make a polynomial query which has the form $\delta' \alpha s$. □

5.3.3 Multi-Authority mCP-ABE

Ideally, we would like to have multiple independent authorities which would manage user attributes and distribute secret keys. Assume that the Attribute Authority (AA) from Hospital A manages the attribute set $\Omega_{HospitalA}$, and that the AA from Hospital B manages the attribute set $\Omega_{HospitalB}$. In the multi-authority setting, the encryptor has the flexibility to choose different attributes from different authorities in the AC policy of the ciphertext, such that only users who have attributes from the given authority can decrypt the ciphertext. For instance, a patient may want to encrypt her health data, such that a user who has the attribute General Practitioner received from Hospital A or the attributes General Practitioner and Pediatrician received from Hospital B can decrypt the ciphertext.

Chase [34] gives the construction of the first multi-authority ABE, which allows multiple independent authorities to monitor user attributes. We can apply the same idea to extend the scheme presented in Section 5.3 to support multi-authority mCP-ABE. The main requirement that we have is that each AA should use the same function which takes as input I_u and outputs u_{id}, where u_{id} is used to connect the components of the secret key. Note that there should be an entity who will manage α. Thus, in addition to AA, a central authority (CA) is needed. We extend the single-authority mCP-ABE scheme presented in Section 5.3 to a multi-authority mCP-ABE as follows (only changes from the scheme in Section 5.3 are presented):

1. Setup :
   (a) Central Authority: Generates a group $\mathbb{G}$ of prime order p with a generator g and a bilinear map $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$. Set the component of the master secret key $\alpha \in_R \mathbb{Z}_p^*$, and the component of the public key $e(g,g)^\alpha$.
   (b) Attribute Authority(AA)$-l$: Generate the attribute set $\Omega_l = \{a_{l,1}, a_{l,2}...a_{l,n}\}$. For each $a_{l,j} \in \Omega_l$ set the attribute secret key: $t_{l,1}...t_{l,n}$, and the attribute public key $\{T_{l,j} = g^{t_{l,j}}\}_{1 \leq j \leq n}$.
2. KeyGen
   (a) Central Authority : Compute the base component of the secret key: $d_0 = g^{\alpha - u_{id}}$.
   (b) Attribute Authority(AA)$-l$: Suppose a user with an identifier I_u applies for the set of attributes ω to the AA l. The AA l computes the attribute secret key as follows: for each $a_{l,j} \in \omega$, compute $d_{l,j,1} = g^{\frac{u_{l,j}}{t_{l,j}}}$ and $d_{l,j,2} = g^{\frac{u_{id} - u_{l,j}}{t_{l,j}}}$, where $u_{l,j} \in_R \mathbb{Z}_p$.

5.4 Applying mCP-ABE in Practice

In this section we describe an application of mCP-ABE. We propose to use mCP-ABE to securely manage Personal Health Records (PHRs). This application demonstrates the practicality and usefulness of our scheme.

Using mCP-ABE to Securely Manage PHRs

Figure 5.2 illustrates a general architecture of a PHR system that uses mCP-ABE. The architecture consists of a publishing server, a data repository that includes a security mediator (Proxy), TA and the data user. The publishing server can be implemented on a home PC of the data source (a patient) or as a trusted service. Its role is to protect and publish health records. The data repository stores encrypted health records, while the Proxy is used in the data consumption phase for revocation. The TA is used to set up the keys. Note that the TA and the publishing server do not always have to be online (the TA is needed only in the set-up phase while the publishing server can upload the protected data in an ad hoc way).

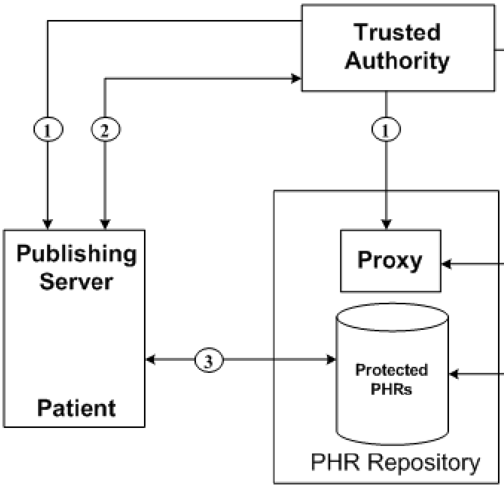

Figure 5.2: *Secure Management of PHRs.*

There are five basic processes in the management of PHRs:

1. Setup: In this phase, the trusted authority (TA) distributes the keys to the patient, the user and the Proxy (1).
2. Data protection (upload of data to the PHR): When a patient wants to upload protected data to the repository, she contacts the TA to check which attributes can be used in the AC policy (2). Then she creates her AC policy and encrypts the data with the keys corresponding to that policy. Then the data is uploaded to the repository. If she wants to change the policy she can re-encrypt the data and update the repository. All this can be done by a publishing server on behalf of the patient who specifies the AC policy (3).
3. Data consumption (doctor's request-response) and revocation: When a user wants to use patient data he contacts the PHR repository and downloads the encrypted data (4). The user makes a request to the Proxy for a decryption token. The request contains the encrypted data and a set of user attributes which satisfy the AC policy associated with the encrypted data. The Proxy checks if any attribute from the user request is not revoked, and, if so, the Proxy generates the decryption token and sends it to the user. After receiving the decryption token the user decrypts the patient data using the keys corresponding to the appropriate attributes which satisfy the AC policy (5).

An additional advantage of an online semi-trusted mediator (Proxy) is that the mCP-ABE scheme can be used to enforce context attributes such as: system date and time or the location where the request comes from. This is useful for healthcare applications which require context-aware AC where access to patients' data depends not only on user roles, but also on the context information. Suppose there is an

AC policy $\tau = (Location = Hospital \wedge Doctor)$ which says that a doctor can view patient's health records only if the doctor's request comes from inside the hospital. Outside the hospital, no user should be able to decrypt the ciphertext encrypted under the AC policy τ, even if the user may own a secret key associated with the attribute *Doctor*. Using mCP-ABE scheme, the enforcement of τ, which contains context attributes, is done as follows:

- A patient encrypts her health record according to the AC policy which contains only the attribute *Doctor*, and then uploads his data to a PHR repository. Hence, part of τ is enforced in the Encrypt phase by the patient.

- A doctor downloads encrypted data and requests a decryption token from the Proxy.

- The Proxy checks the context attribute inside τ and issues a decryption token only if the request comes from inside the hospital (e.g. only if the request comes from a specific IP address), therefore the context part of τ is enforced by the Proxy in the m-Decrypt phase.

Note that the involvement of an online semi-trusted mediator (Proxy) plays a crucial role in the enforcement of context attributes, as it is hard to enforce these attributes without the involvement of an online component.

The mCP-ABE scheme can also support the off-line use of data. Then the architecture is slightly changed in a way that the Proxy is distributed to the users or their domains in which the data will be used. As a consequence there will be a number of proxies which will be coordinated by the central Proxy. The above defined process will not fundamentally change, except that the central Proxy will update the local ones and that in the data consumption phase, the user will contact only the local Proxy.

5.5 Conclusion

In this chapter we propose a mediated Ciphertext-Policy Attribute-Based Encryption scheme that supports revocation of user attributes. If an attribute is revoked, the user cannot use it in the decryption phase. The scheme allows the encryptor to encrypt a message according to an AC policy over a set of attributes. Only users who satisfy the AC policy and whose attributes are not revoked can decrypt the ciphertext. Finally, we demonstrate how to use the proposed scheme to manage Personal-Health Records (PHRs).

Chapter 6

Updating Access Control Policies in Attribute-Based Encryption

As mentioned in Chapter 1, one of the drawbacks of existing CP-ABE schemes (including schemes presented in chapters 4 and 5) is that they do not support updating access control policies without decrypting the ciphertext. In this chapter we introduce a new variant of CP-ABE called Ciphertext-Policy Attribute-Based Proxy Re-encryption (CP-ABPRE). The proposed scheme allows updating the access control policy of the ciphertext without decrypting it. The scheme uses a semi-trusted proxy to re-encrypt the encrypted data according to a new access control policy such that only users who satisfy the updated access control policy can decrypt the ciphertext.

We begin this chapter by giving a motivation for updating access control policies. In Section 6.2 we give a formal definition of the Ciphertext-Policy Attribute-Based Proxy Re-Encryption (CP-ABPRE) scheme and its security definition. Section 6.3 describes the construction of CP-ABPRE and its security proof. The last section concludes the chapter. This chapter builds on previous work presented in a patent application and a refereed paper [1].

6.1 Introduction

As we mentioned in Section 4.3, the ciphertext-policy attribute-based encryption (CP-ABE) scheme allows data to be encrypted according to an access control (AC) policy over a set of descriptive attributes. Once the data is encrypted, it can be stored in an honest-but-curious server, such that everyone can download the encrypted data (even a malicious user), but only users who have the right secret key associated with a set of attributes which satisfy the AC policy can decrypt it. Therefore, when the data is encrypted using a CP-ABE, the AC policy moves with the encrypted data and there is no need for the use of other entities, such as a fully-trusted access-control manager, to enforce the AC policy.

CP-ABE does not support updating AC policies without decrypting the ciphertext. The only way to update an AC policy is to decrypt the ciphertext and then re-encrypt it according to a new AC policy. For instance, if Bob wants to change the AC policy from $\tau_1 = \mathsf{Bob} \vee (\mathsf{GP} \wedge \mathsf{Hospital1})$ to $\tau_2 = \mathsf{Bob} \vee (\mathsf{GP} \wedge (\mathsf{Hospital1} \vee \mathsf{Hospital2}))$, Bob has to send his secret key to the server. Once the server receives the secret key, it decrypts the data and then uses the CP-ABE scheme to re-encrypt the data according to the new policy τ_2. However, the drawback of this approach is that the server accesses sensitive data in plaintext. To avoid this drawback Bob might perform the re-encryption process by himself. To do so, Bob has to download the encrypted data from the server, decrypt the data locally using his secret key, and then re-encrypt the data under the new AC policy. The drawback of this approach is that Bob has to be online during each re-encryption process which is not efficient both from communication and processing point of view.

In Chapter 3, we presented a proxy re-encryption (PRE) scheme which uses a semi-trusted proxy to transform an encryption computed under Alice's (the delegator) public-key to an encryption computed under Bob's (the delegatee) public-key, without decrypting the ciphertext. However, this solution cannot be applied in attribute-base systems, such as CP-ABE. In the scheme of Chapter 3, the mapping *ciphertext-user* is *one-to-one*, therefore, the ciphertext is intended only for one delegatee since there is only one user who is in possession of the secret key and who can decrypt the ciphertext. The case is different in CP-ABE, where the mapping *ciphertext-user* is *one-to-many* and there are many users who have a secret key associated with the attribute set that satisfies the AC policy and decrypt the ciphertext. In addition, what makes the re-encrypting ciphertexts even more challenging in CP-ABE is the requirement of collusion resistance. The scheme must guarantee that different users cannot collude and combine their secret key in order to extend their decryption power.

Our Contribution

In this chapter we overcome the aforementioned drawbacks of updating AC policies in CP-ABE by proposing a new scheme, called ciphertext-policy attribute-based proxy re-encryption (CP-ABPRE). In the proposed scheme Bob has to compute a re-encryption key $rk_{\tau_1 \to \tau_2}$ which is then used by a semi-trusted proxy to re-encrypt all ciphertexts encrypted according to policy τ_1 (without decrypting them) into ciphertexts encrypted according to policy τ_2. The proxy is a semi-trusted entity in the sense that it does not have access to the plain data, but it needs to re-encrypt the ciphertexts. One of the distinctive features of the proposed scheme is that the proxy and the delegatee cannot collude to generate a new secret key.

The construction of the proposed scheme is based on prime order bilinear groups. We give a formal definition of indistinguishability under chosen-plaintext attack (IND-CPA) and provide a security proof in the generic group model.

6.1.1 Related Work

The work presented in this chapter is related to both PRE and ABE. Indeed, the proposed scheme incorporates the idea of re-encrypting the ciphertext from PRE into ABE in order to support updating AC policies. The related work about PRE is presented in Section 3.1.1 and the related work about ABE is presented in Section 4.1.1. In the following we present the most relevant related work as the combination of the two approaches.

Guo *et al.* [91] propose a proxy re-encryption scheme based on the Goyal *et al.*[50] KP-ABE scheme. The proposed scheme can transform a ciphertext associated with a set of attributes into a new ciphertext associated with another set of attributes. In general, adapting CP-ABE to proxy re-encryption is more suitable than adapting KP-ABE to proxy re-encryption since CP-ABE allows the encryptor to express the AC policy in the encryption phase, while in KP-ABE the AC policy is associated with the secret key and is defined in the key generation phase. Lliang *et al.*[65] proposed an attribute-based proxy re-encryption scheme, which is based on the Cheung and Newport CP-ABE scheme [37]. The proposed scheme inherits the same limitations that [37] has: it supports only AC policies with the Boolean operator $\wedge$, and the size of the ciphertext increases linearly with the number of attributes in the system.

6.2 Ciphertext-Policy Attribute-Based Proxy Re-Encryption

A CP-ABPRE scheme extends the CP-ABE by adding a proxy component to the existing components: the trusted authority (TA) and users. Extensions have been made to the number of algorithms as well - the Pkeygen algorithm to generate a re-encryption key and the Preenc algorithm to re-encrypt a ciphertext. A CP-ABPRE scheme is a tuple of six algorithms (Setup, KeyGen, Encrypt, Pkeygen, Preenc, Decrypt) defined as follows:

- Setup, KeyGen and Encrypt are defined respectively in the same way as Setup, KeyGen and Encrypt of the CP-ABE defined in Section 4.3.
- $\mathsf{Pkeygen}(\mathsf{sk}_\omega, \tau_1, \tau_2)$. Run by the delegator, this algorithm takes as input the secret key $\mathsf{sk}_\omega \in \mathcal{K}$ and AC policies τ_1 and τ_2. The algorithm outputs a unidirectional re-encryption key $rk_{\tau_1 \to \tau_2} \in \mathcal{K}$ if $\omega \subseteq \Omega$ satisfies τ_1, or an error symbol $\perp$ if $\omega \subseteq \Omega$ does not satisfy τ_1.
- $\mathsf{Preenc}(c_{\tau_1}, rk_{\tau_1 \to \tau_2})$. Run by the proxy, this algorithm takes as input the ciphertext $c_{\tau_1} \in \mathcal{C}$ and the re-encryption key $rk_{\tau_1 \to \tau_2} \in \mathcal{K}$, and outputs the ciphertext $c_{\tau_2} \in \mathcal{C}$ associated with the AC policy τ_2.
- $\mathsf{Decrypt}(c_\tau, \mathsf{sk}_\omega)$. Run by the decryptor, the algorithm takes as input the ciphertext $c_\tau \in \mathcal{C}$ and the secret key $\mathsf{sk}_\omega \in \mathcal{K}$, and outputs a message $m \in \mathcal{M}$ if $\omega \subseteq \Omega$ satisfies τ, or an error symbol $\perp$ if $\omega \subseteq \Omega$ does not satisfy τ. The ciphertext $c \in \mathcal{C}$ can be either equal to $c_{\tau_1} \in \mathcal{C}$ (produced by Encrypt) or equal to $c_{\tau_2} \in \mathcal{C}$ (produced by Preenc).

Correctness. We say that CP-ABPRE is *correct* if for all security parameters $\lambda \in \mathbb{N}$, for all master secret keys msk and public parameters $\mathcal{PP}$ produced by Setup, for all private keys sk_ω produced by KeyGen, for all ciphertexts c_{τ_1} produced by Encrypt, for all re-encryption keys $rk_{id_{\tau_1} \to id_{\tau_2}}$ produced by Pkeygen, for all re-encrypted ciphertexts c_{τ_2} produced by Preenc, if ω satisfies either τ_1 or τ_2 we should have:

$$\Pr\left[\begin{array}{c} (\mathsf{msk}, \mathcal{PP}) \leftarrow \mathsf{Setup}(\lambda), \mathsf{sk}_\omega \leftarrow \mathsf{KeyGen}(\mathsf{msk}, \omega), c_{\tau_1} \leftarrow \mathsf{Encrypt}(m, \tau_1), \\ rk_{\tau_1 \to \tau_2} \leftarrow \mathsf{Pkeygen}(\mathsf{sk}_\omega, \tau_1, \tau_2), c_{\tau_2} \leftarrow \mathsf{Preenc}(c_{\tau_1}, rk_{\tau_1 \to \tau_2}) : \\ m \leftarrow \mathsf{Decrypt}(c_{\tau_1}, \mathsf{sk}_\omega) \vee m \leftarrow \mathsf{Decrypt}(c_{\tau_2}, \mathsf{sk}_\omega) \end{array}\right] = 1 \ .$$

6.2.1 Security Definitions

In the following we present the game-based security definition of the CP-ABPRE scheme. Informally, the definition guarantees that: a) a user who does not have enough attributes to satisfy the AC policy τ^* of the ciphertext cannot learn any information about the plaintext being encrypted, b) two users cannot combine their attributes to extend their decryption power, for instance two users cannot combine their secret keys and decrypt a ciphertext associated with τ^* if none of users' secret keys satisfy τ^*, and c) the proxy and a user cannot combine the re-encryption key and the secret key in order to compute a new secret key. Therefore in the security-game, played between the adversary $\mathcal{A}$ and the challenger (the challenger simulates the game and answers $\mathcal{A}$'s queries) we allow $\mathcal{A}$ to compromise a user secret key except the secret keys which satisfy the challenge AC policy τ^*. In addition, $\mathcal{A}$ is also allowed to compromise proxy keys or re-encryption keys with the following restriction:

- $\mathcal{A}$ is not allowed to ask secret key queries for the attribute set ω which satisfies τ_2 if $\mathcal{A}$ has a re-encryption key $rk_{\tau^* \to \tau_2}$. The reason for this restriction is that $\mathcal{A}$ can use the re-encryption key to re-encrypt the challenge ciphertext associated with τ^* to a ciphertext associated with τ_2 and decrypt the re-encrypted ciphertext using his secret key which satisfies τ_2. In the sequel we will refer to τ_2 as a challenge derivative AC policy if $\mathcal{A}$ has the re-encryption key $rk_{\tau^* \to \tau_2}$.

At one point of the security-game $\mathcal{A}$ gives to the challenger two messages and the challenge AC policy τ^*, and the challenger returns to $\mathcal{A}$ a ciphertext of one of the two messages encrypted under τ^*. $\mathcal{A}$ has to guess which of the messages was encrypted. If the guess is correct, then $\mathcal{A}$ wins the security-game. The following definition formally captures these attacks.

Definition. *A* CP-ABPRE *scheme is said to be indistinguishable under chosen-plaintext attacks* (IND-CPA) *if any polynomial-time adversary $\mathcal{A}$ has only a negligible advantage in winning the* CP-ABPRE *security-game defined as follows:*

- Setup. *The challenger runs* $\mathsf{Setup}(\lambda)$ *to generate* $(\mathcal{PP}, \mathsf{msk})$*, and gives* $\mathcal{PP}$ *to* $\mathcal{A}$*.*
- Phase 1. $\mathcal{A}$ *performs a polynomially bounded number of queries:*

  - KeyGen. *$\mathcal{A}$ asks for a secret key associated with the attribute set ω_j. The challenger runs* KeyGen(ω_j, msk) *and gives* sk_{ω_j} *to $\mathcal{A}$.*
  - Pkeygen. *$\mathcal{A}$ asks for a re-encryption key for $rk_{\tau_1 \to \tau_2}$, where $\tau_1 \neq \tau_2$. The challenger first runs* KeyGen(ω, msk) *to obtain* sk_ω *and then runs* Pkeygen$(\mathsf{sk}_\omega, \tau_1, \tau_2)$ *to obtain $rk_{\tau_1 \to \tau_2}$. The challenger sends $rk_{\tau_1 \to \tau_2}$ to $\mathcal{A}$.*

- Challenge. *$\mathcal{A}$ sends to the challenger two messages m_0, m_1, such that $|m_0| = |m_1|$, and the challenge AC policy τ^*. $\mathcal{A}$ is not allowed to choose as a challenge AC policy τ^* if it has made the following queries in* Phase 1*:*

  - KeyGen *queries such that* sk_{ω_j} *satisfies a challenge AC policy τ^*.*
  - KeyGen *queries such that* sk_{ω_j} *satisfies any challenge derivative AC policies.*
  - Pkeygen *queries if $\mathcal{A}$ previously has issued* KeyGen *queries such that it has receiver* sk_{ω_j} *which satisfies τ_2, and τ_1 is a challenge derivative AC policy.*

  The challenger picks at random $b \in \{0,1\}$ and returns $c_{\tau^*} =$ Encrypt(m_b, τ^*)*.*

- Phase 2. *$\mathcal{A}$ can continue querying* KeyGen *and* Pkeygen*. $\mathcal{A}$ is not allowed to ask for restricted queries specified in the* Challenge *phase.*

- Guess. *$\mathcal{A}$ outputs a guess $b' \in \{0,1\}$, and wins the game if $b' = b$.*

The advantage of $\mathcal{A}$ in winning the security-game is defined as:

$$\mathsf{ADV}^{\mathsf{IND-CPA}}_{\mathcal{A},\mathsf{CP-ABPRE}}(\lambda) \overset{def}{=} \left| \Pr[b' = b] - \frac{1}{2} \right| ,$$

where the probability is over the random values chosen by $\mathcal{A}$ and the challenger.

6.3 A Construction of CP-ABPRE Scheme

Intuition. First we give some intuition about the construction. The Setup, KeyGen and Encrypt algorithms are based on the CP-ABE schemes presented in Chapter 4. The challenging part in constructing the CP-ABPRE scheme is to compute a re-encryption key $rk_{\tau_1 \to \tau_2}$ such that the proxy is not able to extract the delegator's secret key and to collude with the delegatee in order to create another re-encryption key, say $rk_{\tau_1 \to \tau_3}$. Note that the proxy is honest-but-curious, which implies that the proxy might be interested to know the contents of the ciphertext that it is re-encrypting. Therefore, we have to find a way to enable the proxy to "somehow" partially decrypt the ciphertext associated with the AC policy τ_1, but without accessing the plaintext in clear. Afterwards, we have to allow the proxy to re-encrypt the data and to create a new ciphertext associated with the AC policy τ_2.

To compute the re-encryption key $rk_{\tau_1 \to \tau_2}$, the Pkeygen algorithm randomizes a part of the delegator's secret key (with the random value l) and sends it to the proxy. The delegator also sends in clear the part of the secret key associated with the attribute set ω' which satisfies the access policy τ_1 (it sends in clear $\{d_j\}_{a_j \in \omega'}$). Note that even if the proxy gets these components in clear, it cannot decrypt any ciphertext without de-randomizing the randomized component. In addition, the random value, which is used to randomize the components of the secret key, is first blinded (with g^x) and then encrypted under the AC policy τ_2. The random value is blinded because we want to prevent the delegatee to have it in clear. This technique prevents the proxy to collude with the delegatee and to extract the delegator's secret key. In addition to preventing the collusion between the proxy and the delegatee, we also have to prevent the collusion among delegators such that they should not combine their secret keys in order to extend their decryption power. For this reason, the KeyGen algorithm generates a random value r for each user, which is embedded in each component of the secret key. Therefore, users cannot combine secret keys since different users have different r values.

The Encrypt algorithm enforces the AC policy in a similar way as the Encrypt algorithm of the CP-ABE schemes presented in Chapter 4. In the Preenc phase, the proxy produces a re-encrypted ciphertext, which indeed is a ciphertext created as a result of encrypting the data under a key created from the blinded random value. In the Decrypt phase, first, the delegatee reveals the blinded random value by decrypting the ciphertext which is produced by Pkeygen (i.e. the ciphertext associated with the AC policy τ_2). Finally, the delegatee reveals the data by dividing the ciphertext produced by Preenc with the key generated from the blinded random value.

The Scheme. In this section we describe the construction of the proposed CP-ABPRE scheme. The scheme consists of the following algorithms:

- $\mathsf{Setup}(\lambda)$. The algorithm selects a bilinear group $\mathbb{G}$ of a prime order p and a generator g, and the bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$. Next to this, the algorithm generates a universe (set) of attributes $\Omega = \{a_1, a_2, ..., a_n\}$, picks randomly $\alpha, \beta, f, t_1, t_2, ..., t_n \in \mathbb{Z}_p$, and sets $\{T_j = g^{t_j}\}_{(1 \leq j \leq n)}$. Note that for each $a_j \in \Omega$ there is a $t_j \in \mathbb{Z}_p$. The algorithm also defines the hash function $\mathsf{H}_1 : \mathbb{G}_T \to \mathbb{G}$. The master secret key is : $\mathsf{msk} = \big(\alpha, \beta, f, \{t_j\}_{(1 \leq j \leq n)}\big)$. The public parameters are published as:

$$\mathcal{PP} = \Big(g, e(g,g)^{\alpha+\beta}, g^f, \{T_j\}_{(1 \leq j \leq n)}, \mathsf{H}_1\Big).$$

- $\mathsf{KeyGen}(\mathsf{msk}, \omega)$. The algorithm takes as input the attribute set ω which characterizes the user. The algorithm operates as follows:

  1. Pick at random $r \in \mathbb{Z}_p$ and compute $d_0 = g^{\alpha - r}$.
  2. For each attribute $a_j \in \omega$ compute $d_j = g^{\frac{r+\beta}{t_j}}$.

  The algorithm outputs the secret key $\mathsf{sk}_\omega = \big(d_0, \{d_j\}_{a_j \in \omega}\big)$.

- $\mathsf{Encrypt}(m, \tau_1)$. To encrypt a message $m \in \mathbb{G}_T$ under the AC policy τ_1 over the set of attributes from Ω, the encryption algorithm picks at random $s \in \mathbb{Z}_p$ and assigns s_i values to attributes in τ_1 (s_i values are shares of s and are generated using the Benaloh and Leichter scheme in the same way as in $\mathsf{Encrypt}$ of the $\mathsf{B\text{-}CP\text{-}ABE}$ scheme presented in Chapter 3). The resulted ciphertext consists of the following components:

$$c_0 = g^s \quad , \quad c_1 = m \cdot e\left(g, g\right)^{(\alpha+\beta)s} \ ,$$
$$c_2 = g^{fs} \quad , \quad \{c_{j,i} = g^{t_j s_i}\}_{a_{j,i} \in \tau_1} \ .$$

The algorithm outputs the ciphertext $c_{\tau_1} = \big(c_0, c_1, c_2, \{c_{j,i}\}_{a_{j,i} \in \tau_1}\big)$.

- $\mathsf{Pkeygen}(\mathsf{sk}_\omega, \tau_1, \tau_2)$: The algorithm outputs a re-encryption key which is used by the proxy to update the ciphertext associated with τ_1 to a ciphertext associated with τ_2. Let $\omega' \subseteq \omega$ be the smallest set which satisfies the AC policy τ_1. The algorithm first parses sk_ω as $(d_0, \{d_j^{(2)}\}_{a_j \in \omega})$, picks at random $l, x' \in \mathbb{Z}_p$, it sets $(g^f)^{x'} = g^x$ and computes the re-ecnryption key $rk_{\tau_1 \to \tau_2}$ which consists of the following components:

$$\bar{d_0} = d_0 \cdot g^l \quad , \quad \bar{d_1} = \mathsf{Encrypt}(g^{x-l}, \tau_2) \ ,$$
$$\bar{d_2} = g^{x'} = g^{\frac{x}{f}} \quad , \quad \{\bar{d_j} = d_j\}_{a_j \in \omega'} \ .$$

The algorithm outputs the re-encryption key $rk_{\tau_1 \to \tau_2} = \big(\bar{d_0}, \bar{d_1}, \bar{d_2}, \{\bar{d_j}\}_{a_j \in \omega'}\big)$.
Note: The message g^{x-l} encrypted in this phase (component $\bar{d_1}$) belongs to group $\mathbb{G}$, while the message m encrypted in the $\mathsf{Encrypt}$ phase belongs to group $\mathbb{G}_T$. The encryption of g^{x-l} is done in the same way as the encryption of m with a small change in the computation of c_1. The only purpose of this change is to keep g^{x-l} in the group $\mathbb{G}$. In the $\mathsf{Encrypt}$ phase the c_1 has the form:

$$c_1 \quad = \quad m \cdot e\left(g, g\right)^{(\alpha+\beta)s} \ ,$$

with $s \in_R \mathbb{Z}_p$. While in the $\mathsf{Pkeygen}$ phase the c_1 has the form:

$$c_1 \quad = \quad g^{x-l} \ \cdot \mathsf{H}_1\left(e(g, g)^{(\alpha+\beta)z}\right) \ ,$$

with $z \in_R \mathbb{Z}_p$. All the other components are computed in the same way as in the $\mathsf{Encrypt}$ phase.

- $\mathsf{Preenc}(c_{\tau_1}, rk_{\tau_1 \to \tau_2})$. The algorithm parses c_{τ_1} as $(c_0, c_1, c_2, \{c_{j,i}\}_{a_{j,i} \in \tau_1})$ and $rk_{\tau_1 \to \tau_2}$ as $(\bar{d_0}, \bar{d_1}, \bar{d_2}, \{\bar{d_j}\}_{a_j \in \omega'})$, and operates as follows:

  – Compute:

$$I^{(1)} = \prod_{a_j \in \omega'} e\left(\bar{d_j}, c_{j,i}\right) \quad , \quad I^{(2)} = e\left(c_0, \bar{d_0}\right) \cdot I^{(1)} \quad , \quad I^{(3)} = \frac{c_1}{I^{(2)}}$$

- Define:

$$\bar{c}_0 = c_0 \qquad , \qquad \bar{c}_1 = e\left(c_2, \bar{d}_2\right) \cdot I^{(3)} \qquad , \qquad \bar{c}_2 = \bar{d}_1 \ .$$

The algorithm outputs the re-encrypted ciphertext $c_{\mathcal{T}_2} = (\bar{c}_0, \bar{c}_1, \bar{c}_2)$.

- $\mathsf{Decrypt}(c_{\mathcal{T}}, \mathsf{sk}_\omega)$: The decryption algorithm takes as input a ciphertext $c_{\mathcal{T}}$ and a secret key sk_ω. It checks if the secret key sk_ω related to the attribute set ω satisfies the AC policy τ. If not, the algorithm outputs $\perp$. Otherwise:

1. If ω satisfies the AC policy τ and $c_{\mathcal{T}}$ is equal to $c_{\mathcal{T}_1}$ ($c_{\mathcal{T}_1}$ is a regular ciphertext created by $\mathsf{Encrypt}$), then the decryption algorithm performs the following operations:

   - In the first step, the algorithm chooses the smallest set $\omega' \subseteq \omega$ which satisfies τ_1 and parses $c_{\mathcal{T}_1}$ as $\left(c_0, c_1, c_2, \{c_{j,i}\}_{a_j \in \tau_1}\right)$ and sk_ω as $(d_0, \{d_j\}_{a_j \in \omega})$.
   - In the second step, it computes:

$$Z^{(1)} = \prod_{a_j \in \omega'} e\left(d_j, c_{j,i}\right) \ .$$

   - In the third step, it computes:

$$Z^{(2)} = e\left(d_0, c_0\right) \cdot Z^{(1)} \ .$$

   - In the final step, the message is obtained by computing:

$$m = \frac{c_1}{Z^{(2)}}.$$

   We do not use the c_2 element of the ciphertext in the $\mathsf{Decrypt}$ phase. The c_2 element is used in Preenc only. This phase of the $\mathsf{Decrypt}$ algorithm is almost the same as the $\mathsf{Decrypt}$ algorithm of the $\mathsf{CP\text{-}ABE}$ schemes presented in Chapter 4 (i.e. it considers ciphertexts created by $\mathsf{Encrypt}$ only). The following phase of $\mathsf{Decrypt}$ is unique for the proxy re-encryption since it considers the case when the ciphertext is re-encrypted.

2. If ω satisfies the AC policy τ and $c_{\mathcal{T}}$ is equal to $c_{\mathcal{T}_2}$ ($c_{\mathcal{T}_2}$ is a re-encrypted ciphertext created by Preenc), then the decryption algorithm computes the following:

   - In the first step it parses $c_{\mathcal{T}_2}$ as $(\bar{c}_0, \bar{c}_1, \bar{c}_2)$.
   - In the second step it recovers the message m in the following way:

$$m = \frac{\bar{c}_1}{e\left(\bar{c}_0, \mathsf{Decrypt}\left(\bar{c}_2, \mathsf{sk}_\omega\right)\right)} \ .$$

Note: The operation $\mathsf{Decrypt}(\bar{c}_2, \mathsf{sk}_\omega) = g^{x-l}$ (where g^{x-l} belongs to the group $\mathbb{G}$) is done in similar way as $\mathsf{Decrypt}$ explained under (1.):

$$g^{x-l} = \frac{c_1}{\mathsf{H}_1(Z^{(2)})} \ .$$

Correctness. We show that $\mathsf{Decrypt}$ returns the message m on input of the secret key sk_ω, which is created as a result of running $\mathsf{KeyGen}(\mathsf{msk}, \omega)$, and the ciphertext c_τ. The c_τ can be either equal to c_{τ_1}, which is created as a result of running $\mathsf{Encrypt}(m, \tau_1)$, or equal to c_{τ_2}, which is created as a result of running $\mathsf{Preenc}(c_{\tau_1}, rk_{\tau_1 \to \tau_2})$. Without loss of generality, let $\omega' \subseteq \omega$ be the smallest subset which satisfies both τ_1 and τ_2.

First we show the correctness for $\mathsf{Decrypt}$ when c_τ is equal to c_{τ_1}. We observe that:

$$\begin{aligned} Z^{(1)} &= \prod_{a_j \in \omega'} e\left(g^{\frac{r+\beta}{t_j}}, g^{x_j s_i}\right) \\ &= e\left(g^{r+\beta}, g^s\right) \end{aligned}$$

and

$$\begin{aligned} Z^{(2)} &= e\left(g^{\alpha-r}, g^s\right) \cdot e\left(g^{r+\beta}, g^s\right) \\ &= e\left(g, g\right)^{(\alpha+\beta)s} \ . \end{aligned}$$

Finally, $\mathsf{Decrypt}$ computes m as follows:

$$\begin{aligned} &\frac{c_1}{Z^{(2)}} \\ =\ &\frac{m \cdot e\left(g, g\right)^{(\alpha+\beta)s}}{e\left(g, g\right)^{(\alpha+\beta)s}} \\ =\ &m \ . \end{aligned}$$

Next we show the correctness for $\mathsf{Decrypt}$ when c_τ is equal to c_{τ_2}. First, we observe the form of the ciphertext $c_{\tau_2} = (\bar{c}_0, \bar{c}_1, \bar{c}_2)$:

$$\bar{c}_0 = g^s \qquad , \qquad \bar{c}_2 = \mathsf{Encrypt}(g^{x-l}, \tau_2) \ .$$

The form of $\bar{c}_1$ depends on $I^{(1)}$, $I^{(2)}$, $I^{(3)}$, where:

$$\begin{aligned} I^{(1)} &= \prod_{a_j \in \omega'} e\left(g^{\frac{r+\beta}{t_j}}, g^{t_j s_i}\right) \\ &= e\left(g^{r+\beta}, g^s\right) \ , \end{aligned}$$

$$\begin{aligned} I^{(2)} &= e\left(g^s, g^{\alpha-r} \cdot g^l\right) \cdot e\left(g, g\right)^{(r+\beta)s} \\ &= e\left(g^s, g^{\alpha+\beta} \cdot g^l\right) \ , \end{aligned}$$

$$\begin{aligned} I^{(3)} &= \frac{m \cdot e\left(g^s, g^{\alpha+\beta}\right)}{e\left(g^s, g^{\alpha+\beta} \cdot g^l\right)} \\ &= \frac{m}{e\left(g^s, g^l\right)}\ , \end{aligned}$$

$$\begin{aligned} \bar{c}_1 &= e\left(g^{sf}, g^{\frac{x}{f}}\right) \cdot \frac{m}{e\left(g^s, g^l\right)} \\ &= m \cdot e\left(g^s, g^{x-l}\right)\ . \end{aligned}$$

To compute m, the Decrypt first runs :

$$\mathsf{Decrypt}(\bar{c}_2, \mathsf{sk}_\omega) = g^{x-l}$$

and finally:

$$\begin{aligned} & \frac{\bar{c}_1}{e\left(\bar{c}_0, g^{x-l}\right)} \\ =\ & \frac{m \cdot e\left(g^s, g^{x-l}\right)}{e\left(g^s, g^{x-l}\right)} \\ =\ & m. \end{aligned}$$

Properties

In the following we present the properties of our proposed scheme:

- **Uni-directional.** The re-encryption key $rk_{\tau_1 \to \tau_2}$ only allows the proxy to re-encrypt ciphertexts encrypted under the policy τ_1 into ciphertexts encrypted under policy τ_2, and not the other way around. For instance, the re-encryption key $rk_{\tau_1 \to \tau_2}$ can be used to re-encrypt ciphertexts associated with a policy $\tau_1 = Patient \wedge Bob$ into ciphertext associated with a policy $\tau_2 = General\ Practitioner(GP)$. The idea is that a GP should access his patients' health data, however, individual patients should not be able to access GP's data since a GP possesses data of different patients.

- **Non-Interactive.** The re-encryption key $rk_{\tau_1 \to \tau_2}$ is computed by the delegator without interacting with the delegatee, the TA authority or the proxy. To compute $rk_{\tau_1 \to \tau_2}$, the delegator uses his secret key and the master public key. Therefore the delegator remains off-line while computing the re-encryption key and the proxy performs a re-encryption process to update (or re-encrypt) the ciphertext without any interaction with the delegator.

- **Key Optimal.** The delegator and the delegatee do not need to store extra secrets in addition to their original secret keys associated with a set of attributes, regardless of how many delegations he/she gives (or accepts).

- **Non-transitivity.** The proxy cannot re-delegate the decryption rights. Alternatively it can be said that the proxy cannot combine re-encryption keys to create new delegations. For example, the proxy cannot construct a re-encryption key $rk_{\tau_1 \to \tau_3}$ from other two re-encryption keys $rk_{\tau_1 \to \tau_2}$ and $rk_{\tau_2 \to \tau_3}$.

- **Collusion Safe.** The proxy and the delegatee cannot combine their secret keys to derive a new key. For example, the proxy should not be able to combine the re-encryption key $rk_{\tau_1 \to \tau_2}$, where $\tau_1 = GP \wedge Hospital\ 1$ and $\tau_2 = GP \wedge (Hospital\ 1 \vee Hospital\ 2)$, with a delegatee who has a secret key sk_ω associated with the attribute set $\omega = (GP,\ Hospital\ 2)$ in order to compute the delegator's secret key $\mathsf{sk}_{\omega'}$ associated with the attribute set $\omega' = (GP, Hospital\ 1)$. Collusion safety also implies that two users cannot combine their secret keys. For instance, Alice who has a secret key sk_ω associated with the attribute set $\omega = (Nurse, Hospital\ 1)$ should not be able to combine her secret key with Charlie who has a secret key $sk_{\omega'}$ associated with the attribute set $\omega' = (GP, Hospital\ 2)$ and should not be able to decrypt a ciphertext encrypted under the policy $\tau = Nurse \wedge Hospital\ 2$, which cannot be satisfied neither by Alice nor by Charlie.

- **Multi-User Decryption.** In existing proxy re-encryption schemes, once the proxy performs the re-encryption, the delegator loses the decryption power, thus the delegator cannot use his secret key to decrypt the re-encrypted data. The reason is that the mapping ciphertext-public key is one-to-one, which implies that one ciphertext can be decrypted only by one secret key, thus after the re-encryption is performed only the delegatee has the power to decrypt the ciphertext. One can argue that the proxy can keep a copy of the original ciphertext and enable the delegator to decrypt the original ciphertext. However, this solution requires for the proxy to keep the original ciphertext for each re-encrypted data, which is not efficient.

  The CP-ABPRE scheme has a property which allows the delegator to generate a re-encryption key in such a way that the delegator does not loose his decryption power after the proxy performs the re-encryption, and the re-encrypted ciphertext can be decrypted by many users whose secret key satisfies the AC policy. Thus the server does not have to keep the original ciphertext. As an example, suppose there is a ciphertext associated with the AC policy $\tau_1 = (a \wedge b) \vee (c \wedge d)$. Bob has a secret key $\mathsf{sk}_{\omega_{Bob}}$ associated with a set of attributes $\omega_{Bob} = (a, b, f)$. Since Bob satisfies the AC policy τ_1, Bob is capable to compute a re-encryption key that can update the AC policy τ_1 into another policy $\tau_2 = c \wedge f$. If Bob updates the AC policy τ_1 into τ_2, Bob looses his decryption power because Bob does not satisfy the AC policy τ_2. However, using CP-ABPRE, Bob can retain his decryption power by defining the AC policy as $\widetilde{\tau} = \tau_1 \vee \tau_2$.

- **Multi-User & Single-User Delegation.** In CP-ABPRE many users may have a secret key associated with an attribute set that may satisfy the AC policy associated with the ciphertext. Hence, many users can compute re-encryption keys. However, this property may not always be of potential interest and might

Table 6.1: *Efficiency of* CP-ABPRE.

	Exp.($\mathbb{G}$)	Exp.($\mathbb{G}_T$)	Pairing	H_1
KeyGen	$\|\omega\| + 1$	-	-	-
Pkeygen	$\|\tau_2\| + 4$	1	-	1
Encrypt	$\|\tau_1\| + 2$	1	-	-
Preenc	-	-	$\|\omega'\| + 2$	-
Decrypt	-	-	$\|\omega'\| + 1$	-
Decrypt (Re-encrypted Ciphertexts)	-	-	$\|\omega'\| + 2$	1

be undesirable in some scenarios. In practice we can overcome this problem by defining attributes that are unique to an individual, in addition to the attributes that may be possessed by multiple users. For example, consider Alice who has a secret key sk_ω associated with the attribute set $\omega = (Alice, Patient)$ and a ciphertext encrypted under the AC policy $\tau_1 = Alice \wedge Patient$. Here *Alice* is an individual attribute which can be possessed solely by Alice and *Patient* is an attribute which can be possessed by many users. It is obvious that only Alice satisfies the AC policy τ_1 and only Alice can compute the re-encryption key $rk_{\tau_1 \to \tau_2}$, for any τ_2.

6.3.1 Efficiency Analysis

The size of the secret key sk_ω depends on the number of attributes the user possesses and consists of $|\omega| + 1$ group elements in $\mathbb{G}$. Thus, KeyGen requires $|\omega| + 1$ exponentiations in $\mathbb{G}$. The size of the ciphertext c_{τ_1} depends on the size of the AC policy τ_1 and has $|\tau_1| + 2$ group elements in $\mathbb{G}$, and one group element in $\mathbb{G}_T$. Thus, Encrypt requires $|\tau| + 2$ exponentiations in $\mathbb{G}$ and one exponentiation in $\mathbb{G}_T$. The size of the re-encryption key $rk_{\tau_1 \to \tau_2}$ computed by Pkeygen depends on the size of the attribute set ω' that satisfies τ_1 and on the size of τ_2. However, the computations performed in Pkeygen are independent of ω'. This is because Pkeygen re-uses some components computed by KeyGen. For example, the $\bar{d}_j$ components computed by Pkeygen are the same as the d_j components computed by KeyGen. In total, Pkeygen requires $|\tau_2| + 4$ exponentiations in $\mathbb{G}$, one exponentiation in $\mathbb{G}_T$ and one hash function H_1. Preenc requires $|\omega'| + 2$ pairings and the size of the re-encrypted ciphertext c_{τ_2} is two components from group $\mathbb{G}$ and one component from group $\mathbb{G}_T$.

To decrypt the original ciphertext (i.e. not the re-encrypted ciphertext), Decrypt requires $|\omega'| + 1$ pairing operations, where $|\omega'|$ is the attribute set that satisfies τ_1. If the ciphertext is re-encrypted, then Decrypt requires $|\omega'| + 2$ pairing operations and one hash function H_1, where $|\omega'|$ is the attribute set that satisfies τ_2.

6.3.2 Security Proof

We provide a security proof in the generic group model, described in Section 2.5.2. Informally, the model relies on the fact that it is hard to find the discrete logarithm in a group (including a group with bilinear pairing) when the order of the group is a large prime number. In this model group elements are encoded as unique random strings, in such a way that the adversary $\mathcal{A}$ can manipulate group elements using canonical group operations in $\mathbb{G}$ and $\mathbb{G}_T$ and cannot test any property other than equality.

Theorem 5. *The advantage of any adversary $\mathcal{A}$ in the* CP-ABPRE *security game receiving q group elements from queries it makes to the oracles for computing a group operation in $\mathbb{G}$ and $\mathbb{G}_T$, pairing operation e and from the interaction with the* CP-ABPRE *security-game is bounded by $O(q^2/p)$.*

Proof. In the same was as we did to prove the security of mCP-ABE in Chapter 5, to prove the security of CP-ABPRE we bound the advantage of $\mathcal{A}$ in a modified game in which the challenge ciphertext is either $c_0 = e(g,g)^{(\alpha+\beta)s}$ or $c_0 = e(g,g)^{\theta}$, instead of giving a challenge ciphertext as defined in the security-game of Section 6.2 as $c_0 = m_b \cdot e(g,g)^{(\alpha+\beta)s}$ where $b \in \{0,1\}$. We show that $\mathcal{A}$ cannot distinguish which game is playing. Note that if there is an $\mathcal{A}$ that has an advantage ε in the security-game of Section 6.2 then there can be another $\mathcal{A}$ which has advantage $\frac{\varepsilon}{2}$ in the modified security-game.

We will write $\gamma_0(x)$ as a random encoding for the group element $g^x \in \mathbb{G}$, and $\gamma_1(x)$ as a random encoding for group element $e(g,g)^x \in \mathbb{G}_T$. Each random encoding is associated with a rational function (a function written as a division of two polynomial functions). Let f be a rational function over the variables $\{\alpha, \beta, \theta, s, s_{\hat{i}}, \{t_j\}_{1\leq j\leq n}, r, f, l\}$, where each variable is an element picked in the scheme at random. $\mathcal{A}$ receives the following encodings from the interaction with the simulator in the security-game:

- Components generated by the Setup algorithm:

  1. $\gamma_0(1)$ representing the group generator g.
  2. $\gamma_0(f)$ representing the group element g^f.
  3. $\{\gamma_0(t_j)\}_{1\leq j\leq n}$ representing $\{T_j = g^{t_j}\}_{1\leq j\leq n}$.
  4. $\gamma_1(\alpha+\beta)$ representing $e(g,g)^{\alpha+\beta}$.

- Components generated by the KeyGen oracle in Phase 1 and Phase 2 of the security game. We assume that $\mathcal{A}$ asks for a secret key associated with the attribute set ω.

  1. $\gamma_0(\alpha - r)$ representing $d_0 = g^{\alpha - r}$.
  2. $\{\gamma_0(\frac{r+\beta}{t_j})\}_{a_j\in\omega}$ representing $\{d_j = g^{\frac{r+\beta}{t_j}}\}_{a_j\in\omega}$.

- Components generated by the Pkeygen oracle in Phase 1 and Phase 2 of the security game. Let Pkeygen(τ_1, τ_2) be the re-encryption query used to re-encrypt messages encrypted under the AC policy τ_1 into messages encrypted under the AC policy τ_2. Let ω' be the set of attributes that satisfies the AC policy τ_1.

1. $\gamma_0(\alpha - r + l)$ representing $\bar{d}_0 = g^{\alpha - r + l}$.
2. $\gamma_0(z)$, $\gamma_0(R)$, $\gamma_0(fz)$ and $\{\gamma_0(t_j z_{\hat{i}})\}_{a_{j,\hat{i}} \in \tau_2}$ representing $\bar{d}_1 = \mathsf{Encrypt}(g^{x-l}, \tau_2)$.
3. $\gamma_0(x')$ representing $\bar{d}_2 = g^{x'} = g^{\frac{x}{f}}$.
4. $\{\gamma_0(\frac{r+\beta}{t_j})\}_{a_j \in \omega}$ representing $\{\bar{d}_j = g^{\frac{r+\beta}{t_j}}\}_{a_j \in \omega'}$.

- Components generated by the $\mathsf{Encrypt}$ oracle in the $\mathsf{Challenge}$ phase of the security-game. Let $\mathcal{A}$ asks for a challenge for messages $m_0, m_1 \in \mathbb{G}_T$ and the AC policy τ^*.

   1. $\gamma_0(s)$ representing $c_0 = g^s$.
   2. $\gamma_1(\theta)$ representing $c_1 = e(g, g)^{\theta}$.
   3. $\gamma_0(fs)$ representing $c_2 = g^{fs}$.
   4. $\{\gamma_0(t_j s_{\hat{i}})\}_{a_{j,\hat{i}} \in \tau^*}$ representing $\{c_{j,\hat{i}} = g^{t_j s_{\hat{i}}}\}_{a_{j,\hat{i}} \in \tau^*}$.

$\mathcal{A}$ uses the group elements received from the interaction with the simulator to perform queries to the oracles for group operation in $\mathbb{G}$ and $\mathbb{G}_T$. For instance, the oracle returns $f + f'$ when $\mathcal{A}$ asks for multiplying f and f', or $f - f'$ when $\mathcal{A}$ asks for dividing f and f'. $\mathcal{A}$ can also make queries to the oracle for computing pairing operation e. For instance, the oracle returns $f \cdot f'$ when $\mathcal{A}$ asks for pairing f and f'. We show that $\mathcal{A}$ cannot distinguish with non-negligible advantage the simulation of the modified game where the challenge ciphertext is $c_1 = e(g, g)^{\theta}$, with the simulation of the real game where the challenge ciphertext would have been $c_1 = e(g, g)^{(\alpha+\beta)s}$. First, we show the $\mathcal{A}$'s view when the challenge ciphertext is $\gamma_1(\theta)$. Following the standard approach for security in the generic group model (we follow the same approach in Chapter 5 to prove the security of $\mathsf{mCP\text{-}ABE}$), $\mathcal{A}$'s view can change when an unexpected collision occurs due to the random choice of the formal variables $\{\alpha, \beta, \theta, s, s_{\hat{i}}, \{t_j\}_{1 \le j \le n}, r, f, l\}$. For any two distinct queries the probability of such a collision occurs is $O(\frac{q^2}{p})$. Since for large p the probability of such a collision is negligible we ignore this case.
Second, we show what $\mathcal{A}$'s view would have been if the challenge ciphertext had been set $\gamma_1((\alpha + \beta)s)$. We show that $\mathcal{A}$ cannot make a polynomial query which would be equal to $(\alpha + \beta)s$, and therefore a collision cannot occur. This would also prove our theorem. In Table 6.2 we list possible queries that $\mathcal{A}$ can make into $\mathbb{G}_T$ using the group elements received from interaction with the simulator in the security-game.
As shown in Table 6.2 (the highlighted cell), $\mathcal{A}$ can pair s with $\alpha - r$, and $\frac{r+\beta}{t_j}$ with $s_i t_j$, and then sum the results to get $s(\alpha - r) + \sum_{a_i \in \omega} r s_i + \sum_{a_i \in \omega} \beta s_i$. In order to get only $(\alpha + \beta)s$, $\mathcal{A}$ has to create polynomial requests to cancel sr and to compute βs. We observe that in order $\mathcal{A}$ to obtain βs and sr has to pair $\frac{r+\beta}{t_j}$ with $s_{\hat{i}} t_j$. From

Table 6.2: *Possible queries into $\mathbb{G}_T$.*

1	$\alpha+\beta$	t_j
$(\alpha-r)s$	$(r+\beta)s_i$	$\frac{r+\beta}{t_j}s$
fz	xs	x
$s(\alpha-r)+(r+\beta)s_i$	$r+\beta$	$(r+\beta)s_i$
x_js_i	$(\alpha-r)(x_js_i)$	z
$\alpha-r\pm(r+\beta)s_i$	$s(\alpha-r+l)$	R
$(\alpha+\beta)\pm s$	$(\alpha-r+l)$	

the Table 6.2 we can see that $\mathcal{A}$ can construct a query polynomial of the form:

$$\underbrace{s\alpha}_{A} - \underbrace{sr}_{B} + \sum_{a_i\in\omega} \underbrace{rs_i}_{C} + \sum_{a_i\in\omega} \underbrace{\beta s_i}_{D}.$$

However, $\mathcal{A}$ cannot construct a query polynomial of the form $(\alpha+\beta)s = \alpha s + \beta s$ if $\mathcal{A}$ does not have a secret key which satisfies the AC policy. First, there has to be at least one rs_i missing (there must be one ciphertext component $g^{x_js_i}$ for which $\mathcal{A}$ does not have a secret key component $g^{\frac{\beta+r}{t_j}}$ to pair, therefore $\mathcal{A}$ cannot cancel t_j), therefore $\mathcal{A}$ cannot reconstruct rs under the term C, and as a sequence cannot cancel the term B and C. Second, there must be at least one βs_i missing, hence $\mathcal{A}$ cannot reconstruct βs under the term D. As a result of the above analysis, we conclude that $\mathcal{A}$ cannot make a polynomial query which has the form $(\alpha+\beta)s$. $\square$

6.4 Conclusion

In this chapter we present a new proxy re-encryption scheme in the CP-ABE setting. The proposed scheme (CP-ABPRE) allows a user (the delegator) to change the AC policy associated with the ciphertext dynamically, without decrypting it. To reduce the computations performed at the delegator's side, and to avoid the need for the delegator to be online, the delegator has to compute a re-encryption key which allows the proxy to re-encrypt the ciphertext and to update its AC policy. The re-encrypted ciphertext can be decrypted by users who posses a secret key associated with a set of attributes that satisfies the updated AC policy.

Chapter 7

Public-Key Encryption with Delegated Search

In chapters 3-6 we propose schemes that control the access to the outsourced data by using encryption techniques. In this chapter we consider a scenario when encryption is used for malicious purposes. For instance, an attacker can use traditional public-key encryption (PKE) to encrypt malware, which once decrypted by the private-key holder can compromise all private-key holder's data regardless whether the data is stored in a local computer or is outsourced to another party. To avoid this attack, we construct a mechanism, called *public-key encryption with delegated search* ($\mathcal{PKEDS}$), that enables the private-key holder to provide trapdoors to the server such that the server, given an encrypted data and a malware signature, is able to check whether the encrypted data contains the malware signature, without decrypting it.

We begin the chapter with a motivation and review the related work. In Section 7.2 we define the algorithms of the $\mathcal{PKEDS}$ scheme and formalize the security requirements. In Section 7.4 we present our $\mathcal{PKEDS}$ construction. In Section 7.5 we prove its security and in Section 7.6 we discuss its applications. The last section concludes the chapter. The contents of this chapter is based on a refereed paper [3].

7.1 Introduction

Consider an organization, Carol, whose employees use public-key encryption (PKE) to communicate with other users from outside the organization. All organizational incoming encrypted emails are stored in a server which is managed by Carol. Alice (an employee) can download her encrypted email from the server and decrypt it locally using her private key. Encryption prevents an attacker from learning confidential information, but it opens another problem: the server cannot search the ciphertext for malware. While encryption helps Alice to protect her sensitive data, the hardness of processing the ciphertext without decrypting it, helps the attacker to hide malicious content from the server. Suppose Bob, who resides outside the organization, encrypts a message with Alice's public key, so that only Alice will be able to learn the contents of the message. Unbeknown to Bob, his computer is infected and embeds malware into the message. Since the malware is encrypted, the server is unable to scan the ciphertext for malicious code. A naive solution to detect encrypted malware for Alice is to send her private key to the server. Once the server gets the key, it decrypts the ciphertext and then scans the plain data for a malicious content. However this solution is too risky since the server accesses the plain data and a compromise of the server reveals all Alice's data. Another solution would be to force Alice to scan her email for malicious content. However this approach is not efficient.

If the infected ciphertext is not properly scanned, the malware will get installed on Alice's computer. Once installed, the malware can steal sensitive information from Alice, including her secret keys that she uses to control the access to her outsourced data. This makes all schemes presented in chapters 3-6 useless since the attacker has access to all information needed to decrypt the encrypted data.

In this chapter we focus on finding mechanisms which allow Alice to delegate the power to search her ciphertexts (i.e. ciphertexts that are not created by Alice) to the server, without decrypting it. Searching encrypted data [24] is an attractive technique that might address the aforementioned problem. It allows the server to search encrypted data without learning information about the plain data or the search query. Boneh *et al.* [24] were the first to propose *public-key encryption with keyword search* (a.k.a PEKS or searchable encryption). It works as follows. Bob creates a ciphertext c_w which encrypts the keyword w and Alice creates a trapdoor t_w for a keyword w. The trapdoor t_w is sent to the server, which on receipt of the searchable ciphertext c_w and the trapdoor t_w, runs the Test function which returns *true* only if both the searchable ciphertext c_w and the trapdoor t_w are associated with the same keyword, otherwise it outputs *false*. PEKS is only used to encrypt keywords (meta-data) describing the document, whereas to encrypt the entire document Bob must use a traditional PKE scheme, where the ciphertext is decryptable but not searchable. This approach is not suitable for some applications, such as detecting encrypted malware, for the following reasons: a) the server can search only inside the PEKS ciphertext, the other part of the ciphertext created by the PKE scheme is not searchable, and b) Alice has to stay online - the malware signature database maintained by the server might get updated frequently, therefore Alice has to create trapdoors and send them to the server.

Our Contribution

In this chapter we construct a public-key encryption scheme with delegated search ($\mathcal{PKEDS}$) which has the following properties:

1. Each part of the encrypted data is both searchable and decryptable, unlike in PEKS where only the metadata part of the ciphertext is searchable. Hence, our

scheme can be used on its own, without employing an additional PKE scheme, to provide end-to-end security.

2. Once delegated by Alice, the server is allowed to create any trapdoor without contacting Alice, thus, once the delegation is done, Alice can go offline. We construct a mechanism that enables Alice to provide the server with a master trapdoor t_* such that, given the encrypted data and a word w picked by the server, the server can test whether the word w is in the encrypted data, without decrypting it.

3. The server can answer queries made by Alice. In the proposed scheme, Alice can provide the server with a trapdoor t_w associated with a specific word w such that the server can test whether the word w occurs in the encrypted data, without allowing the server to learn the word w.

We provide a security proof in the standard model and show that the scheme is *ciphertext indistinguishable* and *trapdoor indistinguishable* under the Symmetric External Diffie-Hellman (SXDH) assumption. Note that in our scheme it is *inherently* impossible to achieve these properties with respect to the server (i.e. we can achieve these properties against any adversary excluding the server). The first limitation comes from the fact that we allow the server to hold the master trapdoor t_*, from which the server can create any trapdoor associated to any word and break the ciphertext indistinguishability. The second limitation comes from the nature of the PKE, where an entity (i.e. the server) which holds a trapdoor t_w associated with a specific word w and knows the public key of the receiver can create a valid ciphertext and thus can break the trapdoor indistinguishability. This is also observed by Shen, Shi and Waters [92] and to date the property of the trapdoor indistinguishability is only achieved in the symmetric key setting, where only the secret key holder can create a valid ciphertext. The best that we can achieve with respect to the server is *ciphertext one-wayness* and we show that our scheme is secure in this respect under the modified Computation Diffie-Hellman (mCDH) assumption in the standard model. The construction of the scheme is based on ElGamal private-key encryption (PKE) [45]. Indeed, our scheme can be viewed as an extension of ElGamal, with the additional features: it allows the receiver to create trapdoors and the server to search the encrypted data. The proposed construction uses Type-3 pairings [48] which are employed by the server to search the ciphertext and by the receiver to run the trapdoor generation functions in order to generate the trapdoor. The use of Type-3 pairing is crucial, both for running the testing functions and for preventing an adversary (other than the server) to break the security of the scheme.

7.1.1 Related Work

The idea of searching in encrypted data was first introduced by Song, Wagner and Perrig [96] in the symmetric key setting where the owner of the secret key creates both a searchable ciphertext and a trapdoor, while a server can test whether a given ciphertext and a trapdoor are associated with the same word.

In the public key setting, Boneh *et al.* [24] introduced the first *private-key encryption with keyword search* (PEKS) in which everyone can create a searchable ciphertext but only the owner of a private key can create a trapdoor. The proposed PEKS scheme [24] is based on anonymous identity-based encryption (IBE) as introduced in [27]. Abdalla *et al.* [10] have fixed a consistency flaw from [24] and have provided a transform of an anonymous IBE scheme from Boyen and Waters [30] to construct a PEKS scheme in the standard model. In addition, they have shown how to extend the PEKS scheme to design a public-key encryption with temporary keyword search.

There are a number of improvements to the initial concept of PEKS in which the search is only done by comparing the keyword of the ciphertext with the keyword of the trapdoor. Boneh and Waters [29] have proposed a scheme which supports conjunctive, subset, and range queries over the keywords. Hwang and Lee [54] have proposed a PEKS scheme which works in the multiuser setting, where the keyword is encrypted under many public keys for many receivers. Fuhr and Paillier [47] have proposed a decryptable PEKS scheme with a security proof in the heuristic random oracle model, and Hofheinz and Weinreb [53] propose a decryptable PEKS scheme with a security proof in the standard model.

A concept similar to the decryptable PEKS is the *hybrid model* [14, 107] which integrates PKE and PEKS into a single scheme by allowing both schemes to share the same key pair $(\mathsf{pk}, \mathsf{sk})$. The difference between the hybrid model and the original PEKS scheme is, that the first integrates both PEKS and PKE schemes into a single scheme, while the latter assumes that in addition to PEKS scheme there is a separate PKE scheme. While the hybrid model ties PEKS and PKE, it does not guarantee any relation between messages encrypted under PEKS and messages encrypted under PKE scheme. Particularly, an attacker can always encrypt one message using PEKS scheme and encrypt a different message using PKE scheme, in this way causing the server to send emails in which the receivers are not interested. Our scheme guarantees this relation since it allows the receiver to decrypt the searchable ciphertext and check whether the keywords indeed describe the original message.

7.2 Description and Security Model of $\mathcal{PKEDS}$ Scheme

A public-key encryption scheme with delegated search ($\mathcal{PKEDS}$) consists of the following nine algorithms (Setup, KeyGen_S, KeyGen_R, Encrypt, Delegate, TrapGen, Test_1, Test_2, Decrypt):

- $\mathsf{Setup}(\lambda)$: The setup algorithm is run by a system administrator and takes as input a security parameter λ and outputs public parameters $\mathcal{PP}$.
- $\mathsf{KeyGen}_S(\mathcal{PP})$: This key generation algorithm is run by the server and takes as input public parameters $\mathcal{PP}$ and outputs the server's public/private key pair $(\mathsf{pk}_S, \mathsf{sk}_S)$.
- $\mathsf{KeyGen}_R(\mathcal{PP})$: This key generation algorithm is run by Alice (the receiver) and takes as input public parameters $\mathcal{PP}$ and outputs Alice's public/private key pair $(\mathsf{pk}_R, \mathsf{sk}_R)$.

- $\mathsf{Encrypt}(\mathsf{pk}_R, w)$: The encryption algorithm is run by Bob (the message sender) and takes as input Alice's public key pk_R and a word w, and outputs a ciphertext c_w.

- $\mathsf{Delegate}(\mathsf{sk}_R, \mathsf{pk}_S)$: The delegate algorithm is run by Alice and takes as input Alice's private key sk_R, the server's public key pk_S, and outputs the master trapdoor t_*.

- $\mathsf{TrapGen}(\mathsf{sk}_R, \mathsf{pk}_S, w)$: The trapdoor generation algorithm is run by Alice and takes as input Alice's private key sk_R, the server's public key pk_S and a word w, and outputs the trapdoor t_w.

- $\mathsf{Test}_1(c_w, t_*, t_w, \mathsf{sk}_S)$: The first testing algorithm is run by the server and takes as input a ciphertext c_w, a master trapdoor t_*, a trapdoor t_w associated with the word w, and the server's private key sk_S, and outputs *true* if the ciphertext and the trapdoor are associated with the same word, otherwise outputs $\perp$.

- $\mathsf{Test}_2(c_w, t_*, w, \mathsf{sk}_S)$: The second testing algorithm is run by the server and takes as input a ciphertext c_w, a master trapdoor t_*, a word w, and the server's private key sk_S, and outputs *true* if the ciphertext contains the word w, otherwise outputs $\perp$.

- $\mathsf{Decrypt}(c_w, \mathsf{sk}_R)$: The decryption algorithm is run by Alice and takes as input a ciphertext c_w and Alice's private key sk_R, and outputs the word w or $\perp$ if c_w is invalid.

Correctness. We say that $\mathcal{PKEDS}$ is *correct* if for all security parameters $\lambda \in \mathbb{N}$, for all server public/private key pairs produced by KeyGen_S, for all receiver public/private key pairs produced by KeyGen_R, for all ciphertexts c_w produced by $\mathsf{Encrypt}$, for all master trapdoors t_* produced by $\mathsf{Delegate}$ and for all trapdoors t_w produced by $\mathsf{TrapGen}$, we should have:

$$\Pr\left[\begin{array}{c} \mathcal{PP} \leftarrow \mathsf{Setup}(\lambda), (\mathsf{pk}_S, \mathsf{sk}_S) \leftarrow \mathsf{KeyGen}_S(\mathcal{PP}), (\mathsf{pk}_R, \mathsf{sk}_R) \leftarrow \mathsf{KeyGen}_R(\mathcal{PP}), \\ c_w \leftarrow \mathsf{Encrypt}(\mathsf{pk}_R, w), t_* \leftarrow \mathsf{Delegate}(\mathsf{sk}_R, \mathsf{pk}_S), t_w \leftarrow \mathsf{TrapGen}(\mathsf{sk}_R, \mathsf{pk}_S, w) : \\ w \leftarrow \mathsf{Decrypt}(c_w, \mathsf{sk}_R) \wedge \mathit{true} \leftarrow \mathsf{Test}_1(c_w, t_*, t_w, \mathsf{sk}_S) \\ \wedge\ \mathit{true} \leftarrow \mathsf{Test}_2(c_w, t_*, w, \mathsf{sk}_S) \end{array}\right] = 1 \ .$$

7.3 Security Definitions

7.3.1 Ciphertext Indistinguishability

In the following we describe the basic security property of a $\mathcal{PKEDS}$ scheme which is ciphertext indistinguishability. This property guarantees that it is infeasible for an adversary (other than the server) to learn any information about any word from the ciphertext. The following definition formally captures this property.

Definition. *A $\mathcal{PKEDS}$ is said to be ciphertext indistinguishable under chosen-plaintext attacks if any polynomial-time adversary $\mathcal{A}$ has only a negligible advantage in the chosen-ciphertext indistinguishability attack $\mathcal{CI}-\mathcal{ATK}$ security-game defined as follows:*

- Setup. *The challenger runs* $\mathsf{Setup}(\lambda)$ *and forwards $\mathcal{PP}$ to $\mathcal{A}$.*
- $\mathsf{KeyGen_S}$ Query. *$\mathcal{A}$ requests the server's public key* pk_S. *The challenger runs* $\mathsf{KeyGen}_S(\mathcal{PP})$ *and forwards* pk_S *to $\mathcal{A}$.*
- $\mathsf{KeyGen_R}$ Query. *$\mathcal{A}$ requests the receiver's key pair* $(\mathsf{pk}_R, \mathsf{sk}_R)$. *The challenger runs* $\mathsf{KeyGen}_R(\mathcal{PP})$ *and forwards* $(\mathsf{pk}_R, \mathsf{sk}_R)$ *to $\mathcal{A}$.*
- $\mathsf{Delegate}$ Query. *$\mathcal{A}$ requests the master trapdoor t_* associated with the receiver R and server S. The challenger runs* $\mathsf{Delegate}(\mathsf{sk}_R, \mathsf{pk}_S)$ *and forwards t_* to $\mathcal{A}$.*
- $\mathsf{TrapGen}$ Query. *$\mathcal{A}$ requests a trapdoor for the message w. The challenger runs* $\mathsf{TrapGen}(\mathsf{sk}_R, \mathsf{pk}_S, w)$ *and forwards t_w to $\mathcal{A}$.*
- $\mathsf{Challenge}$. *$\mathcal{A}$ sends to the challenger two messages w_0, w_1 such that $|w_0| = |w_1|$, and the identity of the receiver R^*. We restrict $\mathcal{A}$ such that if $\mathcal{A}$ performs a* KeyGen_R *Query for the receiver key pair R^*, then the challenger returns only the public key* pk_{R^*}. *The challenger randomly picks $b \in \{0,1\}$, runs $c_{w_b} \leftarrow$* $\mathsf{Encrypt}(\mathsf{pk}_{R^*}, w_b)$ *and forwards c_{w_b} to $\mathcal{A}$.*
- $\mathsf{Final\ Phase}$. *$\mathcal{A}$ outputs a bit $b' \in \{0,1\}$, and wins the security-game if $b' = b$.*

The advantage of $\mathcal{A}$ in winning the above security-game is defined as:

$$\mathsf{ADV}^{\mathcal{CI}-ATK}_{\mathcal{A},\mathcal{PKEDS}}(\lambda) \overset{def}{=} \left|\Pr[b' = b] - \frac{1}{2}\right| ,$$

where the probability is over the random values chosen by $\mathcal{A}$ and the challenger.

The scheme does not achieve $\mathcal{CI}$ against the server. Given the challenge ciphertext c_{w_b}, the master trapdoor t_* and the words (w_0, w_1), the server runs $\mathsf{Test}_2(c_{w_b}, t_*, w_0, \mathsf{sk}_S)$ to check whether the trapdoor and the ciphertext are associated with the same word. If the output of Test_2 is *true* then the server learns that $b = 0$, otherwise it learns that $b = 1$. As $\mathcal{CI}$ against the server is inherently not possible (remember that our focus is to allow the server to search the ciphertext); the best we can achieve against the server is ciphertext one-wayness.

Note. The security model that we consider in this chapter is *weaker* than the original security model considered in PEKS [24]. Our model gives more power to the server since the scheme allows the server to generate any trapdoor. This is risky for low entropy messages since the server can find the message by trial and error. Nevertheless, for high entropy messages this attack is hard. For instance, if the server scans a ciphertext which contains user fingerprints, then it is hard for the server to guess the fingerprint by running the trial and error attack. This makes our scheme suitable for situations when the server is managed by an organization which wants to protect their employees from potential malicious senders while avoiding the need of giving the private key to the server.

7.3.2 Trapdoor Indistinguishability

The *trapdoor indistinguishability* property guarantees that it is infeasible for an adversary (except the server) to learn any information about any word from the trapdoor. Baek *et al.* [15] observe that PEKS scheme presented by Boneh *et al.* [24] assumes a secure channel between the server and the receiver. If there is no secure channel, then everyone can break the *trapdoor indistinguishability* property since everyone can play the role of the server. To remove the secure channel between the receiver and the server, Baek *et al.* [15] propose a scheme where the sender encrypts the PEKS ciphertext with the public key of the server, in such a way that only the server who knows the private key can reveal the PEKS ciphertext. In this chapter we take a different approach to achieve *trapdoor indistinguishability* against outside adversaries. The Baek *et al.* [15] solution is not suitable in our setting since we allow the receiver to decrypt the $\mathcal{PKEDS}$ ciphertext, otherwise if we encrypt the $\mathcal{PKEDS}$ ciphertext with the server's public key, then the receiver cannot decrypt the ciphertext without getting help from the server. Instead, the role of the server is to search the encrypted data and not to help the receiver to decrypt the ciphertext. To achieve *trapdoor indistinguishability*, we need the secure channel established between the receiver and the server, as assumed in [24]. This implies that, instead of encrypting the communication between the sender and the receiver, we encrypt the communication between the receiver and the server. Namely, before sending the trapdoor to the server, the receiver encrypts the trapdoor under the server's public key. Since only the server has the private key, only the server can reveal the trapdoor and search the encrypted data. We capture the property of *trapdoor indistinguishability* through the following definition.

Definition. *A $\mathcal{PKEDS}$ is said to have the property of the trapdoor indistinguishability if any polynomial-time adversary $\mathcal{A}$ has only a negligible advantage in the trapdoor indistinguishability attack $\mathcal{TI} - ATK$ security-game defined as follows:*

- Setup. *The challenger runs* $\mathsf{Setup}(\lambda)$ *and forwards* $\mathcal{PP}$ *to* $\mathcal{A}$.

- $\mathsf{KeyGen_S}$ Query. $\mathcal{A}$ *requests the server's public key* pk_S. *The challenger runs* $\mathsf{KeyGen}_S(\mathcal{PP})$ *and forwards* pk_S *to* $\mathcal{A}$

- $\mathsf{KeyGen_R}$ Query. $\mathcal{A}$ *requests the receiver's public key* pk_R. *The challenger runs* $\mathsf{KeyGen}_R(\mathcal{PP})$ *and forwards* $(\mathsf{pk}_R, \mathsf{sk}_R)$ *to* $\mathcal{A}$.

- Delegate Query. $\mathcal{A}$ *requests the master trapdoor* t_* *associated with the receiver* R *and server* S *. The challenger runs* $\mathsf{Delegate}(\mathsf{sk}_R, \mathsf{pk}_S)$ *and forwards* t_* *to* $\mathcal{A}$.

- TrapGen Query. $\mathcal{A}$ *requests a trapdoor for the message* w. *The challenger runs* $\mathsf{TrapGen}(\mathsf{sk}_R, \mathsf{pk}_S, w)$ *and forwards* t_w *to* $\mathcal{A}$.

- Challenge. $\mathcal{A}$ *sends to the challenger two messages* w_0, w_1 *such that* $|w_0| = |w_1|$, *and the identity of the receiver* R^*. *The challenger randomly picks* $b \in \{0,1\}$, *runs* $t_{w_b} \leftarrow \mathsf{TrapGen}(\mathsf{sk}_R, \mathsf{pk}_S, w_b)$ *and forwards* t_{w_b} *to* $\mathcal{A}$.

- Guess. *$\mathcal{A}$ outputs a bit $b' \in \{0,1\}$, and wins the security-game if $b' = b$.*

The advantage of $\mathcal{A}$ in winning the trapdoor indistinguishability attack $\mathcal{TI} - ATK$ security-game is defined as:

$$\mathsf{ADV}^{\mathcal{TI}-ATK}_{\mathcal{A},\mathcal{PKEDS}}(\lambda) \overset{def}{=} \left|\Pr[b' = b] - \frac{1}{2}\right| ,$$

where the probability is over the random values chosen by $\mathcal{A}$ and the challenger.

Under this definition we achieve $\mathcal{TI}$ only for adversaries other than the server. Informally speaking, we achieve this property by not allowing an adversary to search the encrypted data. In particular, we cannot achieve $\mathcal{TI}$ from the server who runs the Test_1 function since the server can guess the word in the following way: the server sends to the challenger two words (w_0, w_1) and the challenger replies to the server by sending t_{w_v} for a random bit $v \in \{0,1\}$. Next, the server chooses a random bit $v' \in \{0,1\}$ and runs $c_{v'} \leftarrow \mathsf{Encrypt}(\mathsf{pk}_R, w_{v'})$. Finally, the server run $\mathsf{Test}_1(c_{v'}, t_*, t_{w_v}, \mathsf{sk}_S)$ and outputs $v' = v$ if the output of Test_1 is *true*.

7.3.3 Ciphertext One-Wayness

The property of *ciphertext one-wayness* guarantees that it is hard for an adversary to invert the ciphertext and to learn the word even if the adversary holds the server's private key, the master trapdoor and the trapdoor associated with that word, but the adversary does not hold the receiver's private key. The following definition formally captures this property.

Definition. *A $\mathcal{PKEDS}$ is said to have the property of the ciphertext one-wayness if any polynomial-time adversary $\mathcal{A}$ has only a negligible advantage in the ciphertext one-wayness attack $\mathcal{CTOW} - ATK$ security-game defined as follows:*

- Setup. *The challenger runs $\mathsf{Setup}(\lambda)$ and forwards $\mathcal{PP}$ to $\mathcal{A}$.*
- $\mathsf{KeyGen_S}$ Query. *$\mathcal{A}$ requests the server's key pair $(\mathsf{pk}_S, \mathsf{sk}_S)$. The challenger runs $\mathsf{KeyGen}_S(\mathcal{PP})$ and forwards $(\mathsf{pk}_S, \mathsf{sk}_S)$ to $\mathcal{A}$.*
- $\mathsf{KeyGen_R}$ Query. *$\mathcal{A}$ requests the receiver's key pair $(\mathsf{pk}_R, \mathsf{sk}_R)$. The challenger runs $\mathsf{KeyGen}_R(\mathcal{PP})$ and forwards $(\mathsf{pk}_R, \mathsf{sk}_R)$ to $\mathcal{A}$.*
- $\mathsf{Delegate}$ Query. *$\mathcal{A}$ requests the master trapdoor t_* associated with the receiver R and server S . The challenger runs $\mathsf{Delegate}(\mathsf{sk}_R, \mathsf{pk}_S)$ and forwards t_* to $\mathcal{A}$.*
- $\mathsf{TrapGen}$ Query. *$\mathcal{A}$ requests a trapdoor for the word w. The challenger runs $\mathsf{TrapGen}(\mathsf{sk}_R, \mathsf{pk}_S, w)$ and forwards t_w to $\mathcal{A}$.*
- $\mathsf{Challenge}$. *$\mathcal{A}$ sends the identity of the receiver R^* to the challenger. The challenger picks a random word w^*, it runs $c_{w^*} \leftarrow \mathsf{Encrypt}(\mathsf{pk}_{R^*}, w^*)$ and $t_{w^*} \leftarrow \mathsf{Encrypt}(\mathsf{sk}_{R^*}, \mathsf{pk}_S, w^*)$, and it forwards c_{w^*} and t_{w^*} to $\mathcal{A}$. Note that $\mathcal{A}$ is not allowed to learn sk_{R^*} from the* $\mathsf{KeyGen_R}$ Query.

- Final Phase. $\mathcal{A}$ *outputs* w', *and wins the security-game if* $w' = w^*$.

The advantage of $\mathcal{A}$ *in winning the ciphertext one-wayness attack* $\mathcal{CTOW} - ATK$ *security-game is defined as:*

$$\mathsf{ADV}^{\mathcal{CTOW}-ATK}_{\mathcal{A},\mathcal{PKEDS}}(\lambda) \overset{def}{=} \Pr[w' = w] \ ,$$

where the probability is over the random values chosen by $\mathcal{A}$ *and the challenger.*

7.4 Construction of the $\mathcal{PKEDS}$ Scheme

Intuition. Existing searchable public-key encryption schemes (such as PEKS) are based on anonymous identity-based encryption (anonymous IBE) schemes. In an anonymous IBE, by looking only at the ciphertext one cannot reveal the identity of the receiver. The same applies to PEKS - by looking at the searchable ciphertext one cannot reveal the keyword. In PEKS, the property of anonymity prevents the receiver to know the keyword for which the searchable ciphertext is created, unless the receiver guesses the keyword. It is exactly the property of anonymity that prevents PEKS to be used on its own. When PEKS is used, there is always a need for an additional PKE scheme to encrypt the word (i.e. not only keywords), which can easily be decrypted by the receiver [1].

Our scheme works quite differently and does not rely on the traditional anonymous IBE transformation, which we identify as a main source that does not allow the receiver to decrypt the encrypted keywords. At a high level, in our construction both the ciphertext and the trapdoor are somehow *encryptions* of the same word w [2].The server runs testing functions to check whether both encryptions contain the same word, and if so, outputs *true* to indicate that the ciphertext contains the word specified by the receiver when creating the trapdoor, otherwise outputs *false.* Note that this is different from Naor-Yung [78] *double encryption paradigm* in which the message sender generates a proof that two ciphertexts generated from the same public-key encrypt the same message. In our construction, the sender does not generate any proof and the ciphertext and the trapdoor are generated from different public keys. More specifically, the ciphertext is created from the receiver's public-key using the Encrypt algorithm of ElGamal scheme [45] and the trapdoor is created in such a way that it takes as input the server's public keys and the receiver's private-key. When encrypting messages and decrypting ciphertexts the scheme does not use pairing operations and works only with group elements from $\mathbb{G}$, while Type-3 pairing operations are used to generate trapdoors and to run the testing functions only.

[1] In $\mathcal{PKEDS}$ we use the notion *word* to refer to the message being encrypted. In PEKS the message being encrypted is referred to as *keyword.* The reason that we use *word* and not *keyword,* as we will explain in Section 7.6, is that in the $\mathcal{PKEDS}$ every word of the message is searchable (i.e. not only words describing the message(a.k.a keywords)).

[2] In PEKS the Encrypt algorithm always take as input the same message, *true*, encrypted under a public-key which is generated from the keyword string. In our construction the Encrypt algorithm takes as input the keyword (or word) itself encrypted under recipients public key.

The Scheme. We are now ready to present our construction. When explaining the scheme we consider the single-user setting. The scheme consists of nine algorithms (Setup, KeyGen_S, KeyGen_R, $\mathsf{Encrypt}$, $\mathsf{Delegate}$, $\mathsf{TrapGen}$, Test_1, Test_2, $\mathsf{Decrypt}$) defined as follows:

- Setup. On input of the security parameter λ the algorithm outputs public parameters ($\mathcal{PP}$) which contain the description of groups $\langle \mathbb{G}, \Gamma \rangle$ of order p, the bilinear map $e : \mathbb{G} \times \Gamma \rightarrow \mathbb{G}_T$, the generators g and γ of the groups $\mathbb{G}$ and Γ respectively.

- $\mathsf{KeyGen}_S(\mathcal{PP})$. On input of public parameters $\mathcal{PP}$ the algorithm picks a random $x \in \mathbb{Z}_p$ and outputs the server's key pair:
$$(\mathsf{sk}_S, \mathsf{pk}_S) = (x, p_s = \gamma^x) \ .$$

- $\mathsf{KeyGen}_R(\mathcal{PP})$. On input of public parameters $\mathcal{PP}$ the algorithm picks a random $\alpha, y \in \mathbb{Z}_p$ and outputs the receiver's key pair:
$$(\mathsf{sk}_R, \mathsf{pk}_R) = ((y, \gamma^\alpha), p_r = g^y) \ .$$

- $\mathsf{Encrypt}(\mathsf{pk}_R, w)$. On input of the receiver's public key and a word $w \in \mathbb{G}$ the algorithm picks a random $k \in \mathbb{Z}_p$ and outputs the ElGamal ciphertext:
$$c_w = (c_1, c_2) = \left(w \cdot p_r^k, g^k\right) \ .$$

- $\mathsf{Delegate}(\mathsf{pk}_S, \mathsf{sk}_R)$. The algorithm creates a master trapdoor to allow the server to search the encrypted data for any word of her choice. The algorithm picks at random $r_1, r_2 \in \mathbb{Z}_p$ and outputs the master trapdoor:
$$t_* = (t_1, t_2, t_3, t_4) = (\gamma^\alpha \cdot p_s^{r_1}, \gamma^{r_1}, \gamma^{y\alpha} \cdot p_s^{r_2}, \gamma^{r_2}) \ .$$

- $\mathsf{TrapGen}(\mathsf{sk}_R, \mathsf{pk}_S, w)$. The algorithm creates a trapdoor to allow the server to search for a specific word w. The algorithm picks a random $\delta \in \mathbb{Z}_p$ and outputs the trapdoor:
$$t_w = (t_5, t_6) = \left(e(w, \gamma^\alpha) \cdot e(p_r, p_s^\delta), \gamma^\delta\right) \ .$$

- $\mathsf{Test}_1(c_w, t_*, t_w, \mathsf{sk}_S)$. The algorithm tests whether the ciphertext contains the same word as the trapdoor. The algorithm parses c_w as (c_1, c_2), t_* as (t_1, t_2, t_3, t_4), t_w as (t_5, t_6) and defines:
$$t_7 = \frac{t_1}{t_2^x} \quad , \quad t_8 = \frac{t_3}{t_4^x} \quad , \quad \tilde{a} = \frac{e(p_r, t_6^x) \cdot e(c_1, t_7)}{t_5} \quad , \quad \tilde{b} = e(c_2, t_8).$$
Finally, the algorithm checks whether $\tilde{a} \overset{?}{=} \tilde{b}$. If this equation holds, the algorithm outputs *true* indicating that the ciphertext contains the same word as the trapdoor, otherwise it outputs *false*.

- $\mathsf{Test}_2(c_w, t_*, w, \mathsf{sk}_S)$. The algorithm tests whether the ciphertext contains the word w. The algorithm parse c_w as (c_1, c_2), t_* as (t_1, t_2, t_3, t_4) and defines:

  $$t_7 = \frac{t_1}{t_2^x} \quad , \quad t_8 = \frac{t_3}{t_4^x} \quad , \quad \tilde{c} = e(c_1, t_7) \quad , \quad \tilde{d} = e(c_2, t_8) \ .$$

  Finally, the algorithm checks whether $\tilde{c}/\tilde{d} \overset{?}{=} e(w, t_7)$. If this equation holds, the algorithm outputs *true* indicating that the ciphertext contains the word w, otherwise it outputs *false*.

- $\mathsf{Decrypt}(\mathsf{sk}_R, c_w)$. On input of the ciphertext and the receiver's private key the algorithm outputs:

  $$w = \frac{c_1}{c_2^y}.$$

Correctness. We show that when a ciphertext is created as a result of running $\mathsf{Encrypt}$ on input of the word w and the receiver's public key pk_R, then the same word w is revealed when running $\mathsf{Decrypt}$ on input of the ciphertext and the receiver's private key sk_R. This is shown as follows:

$$\frac{c_1}{c_2^y} = \frac{w \cdot p_r^k}{g^{rs}} = \frac{w \cdot g^{rs}}{g^{rs}} = w \ .$$

Next, we show the correctness for Test_1 algorithm. We observe that:

$$\begin{aligned} t_7 &= \frac{\gamma^\alpha \cdot p_s^{r_1}}{\gamma^{xr_1}} = \gamma^\alpha \qquad , \qquad t_8 = \frac{\gamma^{y\alpha} \cdot p_s^{r_2}}{\gamma^{xr_2}} = \gamma^{y\alpha} \ , \\ \tilde{a} &= \frac{e(p_r, \gamma^{x\delta}) \cdot e(w \cdot p_r^k, \gamma^\alpha)}{e(w, \gamma^\alpha) \cdot e(p_r, p_s^\delta)} = e(p_r^k, \gamma^\alpha) \quad , \quad \tilde{b} = e(g^k, \gamma^{y\alpha}) = e(p_r^k, \gamma^\alpha) \ . \end{aligned}$$

Thus $\tilde{a} = \tilde{b}$ and the output is *true* indicating that the word associated with the ciphertext and the word associated with the trapdoor are the same. Finally, we show the correctness for the Test_2 algorithm. We observe that:

$$\begin{aligned} t_7 &= \frac{\gamma^\alpha \cdot p_s^{r_1}}{\gamma^{xr_1}} = \gamma^\alpha \quad , \quad t_8 = \frac{\gamma^{y\alpha} \cdot p_s^{r_2}}{\gamma^{xr_2}} = \gamma^{y\alpha} \quad , \quad \tilde{c} = e(w \cdot p_r^k, \gamma^\alpha) \ , \\ \tilde{d} &= e(g^k, \gamma^{y\alpha}) \quad , \quad \frac{\tilde{c}}{\tilde{d}} = e(w, \gamma^\alpha) \ . \end{aligned}$$

Thus $\tilde{c}/\tilde{d} = e(w, t_7)$ and the output is *true* indicating that the ciphertext is an encryption of the word w.

7.4.1 Efficiency

In Table 7.1 we count the number of calculations in KeyGen_S, KeyGen_R, $\mathsf{Encrypt}$, $\mathsf{Delegate}$, $\mathsf{TrapGen}$, Test_1, Test_2 and $\mathsf{Decrypt}$. KeyGen_S requires one exponentiation

Table 7.1: *Efficiency of $\mathcal{PKEDS}$.*

	Exp.($\mathbb{G}$)	Exp.(Γ)	Exp.($\mathbb{G}_T$)	Pairing
KeyGen$_S$	-	1	-	-
KeyGen$_R$	1	1	-	-
Encrypt	2	-	-	-
Delegate	-	5	-	-
TrapGen	-	1	1	2
Test$_1$	-	2	1	3
Test$_2$	-	2	-	3
Decrypt	1	-	-	-

in Γ and KeyGen$_R$ requires one exponentiation in $\mathbb{G}$ and one exponentiation in Γ. Encrypt and Decrypt are the same as in ElGamal. Encrypt requires two exponentiations in $\mathbb{G}$ which are independent of the word and can be computed ahead of time and Decrypt requires only one exponentiation in $\mathbb{G}$. Delegate requires five exponentiations in Γ. TrapGen requires one exponentiation in Γ, one exponentiation in $\mathbb{G}_T$ and two pairing operations. Test$_1$ requires two exponentiations in Γ, one exponentiation in $\mathbb{G}_T$ and three pairing operations. Test$_2$ requires two exponentiations in Γ and three pairing operations.

7.5 Security Proof

We now show that the scheme is ciphertext indistinguishable, trapdoor indistinguishable and that the scheme offers ciphertext one-wayness.

7.5.1 Ciphertext Indistinguishability

When proving this property we will closely follow the security proof of [16]. In the following we show that our construction is CI-ATK secure as long as the SXDH assumption holds.

Theorem 6. *Suppose that there is an adversary $\mathcal{A}$ that can break the CI-ATK of the $\mathcal{PKEDS}$ scheme with advantage ε. Then we can construct a polynomial-time reduction $\mathcal{B}$ that breaks the* SXDH *assumption with advantage $(1 - \frac{q}{n})\frac{1}{n}\frac{\varepsilon}{2}$ where q is the number of queries asked by $\mathcal{A}$, and n is the number of receivers in the system.*

Proof. The challenger selects a bilinear map $e : \mathbb{G} \times \Gamma \rightarrow \mathbb{G}_T$, and the generator g and γ of groups $\mathbb{G}$ and Γ respectively. Then, it picks at random $a, b \in \mathbb{Z}_p$, computes $T_0 = g^{ab}$ and picks at random $T_1 \in_R \mathbb{G}$. It flips a fair coin $\mu \in_R \{0, 1\}$ and gives the SXDH tuple $(g, g^a, g^b, T_\mu) \in \mathbb{G}$ to the reduction $\mathcal{B}$. The goal of $\mathcal{B}$ is to solve the SXDH assumption and acts as $\mathcal{A}$'s challenger as follows:

- Setup : $\mathcal{B}$ publishes $\mathcal{PP}$ in the same way as in the scheme.
- $\mathsf{KeyGen_S}$ Query : $\mathcal{B}$ generates the server's key pair $(\mathsf{sk}_S, \mathsf{pk}_S) = (x, p_s = \gamma^x)$, where $x \in_R \mathbb{Z}_p$ is chosen in the same way as in the scheme. $\mathcal{B}$ sends to $\mathcal{A}$ the server's public key pk_S. The distribution of the server's key pair is identical to the server's key pair of the scheme since g and γ are random generators, and x is a random exponent, all chosen in the same way as in the scheme.
- $\mathsf{KeyGen_R}$ Query : $\mathcal{B}$ answers the receiver's key generation queries by computing $(\mathsf{sk}_R, \mathsf{pk}_R) = ((y, \gamma^\alpha), p_r = g^y)$ where $\alpha, y \in_R \mathbb{Z}_p$ are chosen in the same way as in the scheme (each user has a different α and y value). If the query is for R^*, $\mathcal{B}$ sets the public key equal to $p_r = g^a$ (this parameter is extracted from the SXDH instance). Note that $\mathcal{B}$ does not know the private key of R^* ($\mathcal{B}$ does not know a). The distribution of the receiver's key pair is identical to the distribution of the receiver's key pair of the scheme since g is a random generator, α, a and y are random exponents, all chosen in the same way as in the scheme.
- Delegate Query : $\mathcal{A}$ requests a master trapdoor for the receiver R. If R is equal to R^*, $\mathcal{B}$ aborts the simulation and returns a guess μ'. Otherwise, if R is not equal to R^*, $\mathcal{B}$ computes the random elements $r_1, r_2 \in \mathbb{Z}_p$ and outputs the master trapdoor $t_* = (t_1, t_2, t_3, t_4) = (\gamma^\alpha \cdot p_s^{r_1}, \gamma^{r_1}, \gamma^{y\alpha} \cdot p_s^{r_2}, \gamma^{r_2})$. When $\mathcal{B}$ does not abort, the distribution of the master trapdoor is identical to the distribution of the master trapdoor in the scheme since r_1 and r_2 are random elements from $\mathbb{Z}_p$, the same as in the real scheme.
- TrapGen Query : $\mathcal{A}$ requests a trapdoor for the pair (R, w) [3]. If R is equal to R^*, $\mathcal{B}$ aborts the simulation and returns a guess μ'. Otherwise, if R is not equal to R^*, $\mathcal{B}$ picks random $\delta \in \mathbb{Z}_p$ and outputs the trapdoor $t_w = (t_5, t_6) = \big(e(w, \gamma^\alpha) \cdot e(p_r, p_s^\delta), \gamma^\delta\big)$.

  When $\mathcal{B}$ does not abort, the distribution of the trapdoor is identical to the distribution of the trapdoor in the scheme since δ is chosen at random from $\mathbb{Z}_p$, the same as in the real scheme.
- Challenge : $\mathcal{A}$ requests a ciphertext for one of the two words w_0 and w_1 generated under the public key of the receiver R. If R is equal to R^*, $\mathcal{B}$ flips a fair binary coin $v \in \{0, 1\}$ and outputs the ciphertext $\hat{c}_{w_v} = (c_1, c_2) = (w_v \cdot T_\mu, g^b)$, where g^b and T_μ are parameters extracted from the SXDH instance. The distribution of the ciphertext is identical to the distribution of the ciphertext in the scheme only if $T_\mu = g^{ab}$. Otherwise, if $T_\mu \in_R \mathbb{G}$, the ciphertext is a random element from $\mathbb{G}$. If R is not equal to R^*, $\mathcal{B}$ aborts the simulation and returns a guess μ'.
- Guess : At the end of the security-game, without loss of generality, we assume that $\mathcal{A}$ has ciphertexts for all keywords generated by each user, and has requested trapdoor queries to oracles $\mathcal{O}_2$ and $\mathcal{O}_3$ generated from all but one user.

[3] A trapdoor associated with the word w generated by user (receiver) R.

Therefore, we assume that at the end of the security-game, in a non-aborted simulation case, $\mathcal{A}$ should have ciphertexts generated by all users for each keyword, and have at least one challenge ciphertext, denoted as $\hat{c}_v$, which is either a valid or invalid ciphertext generated by R^* for which $\mathcal{A}$ does not have the corresponding trapdoor. Lastly, $\mathcal{A}$ outputs a guess v'. If the guess is correct $v' = v$, then $\mathcal{B}$ sets $\mu' = 0$ indicating that $T_0 = g^{ab}$, otherwise $\mathcal{B}$ sets $\mu' = 1$ indicating that $T_1 \in_R \mathbb{G}$.

Suppose $\mathcal{B}$ does not abort (denoted as $\overline{abort}$) during the simulation. If $\mu = 0$ then the ciphertext $\hat{c}_v$ is a valid ciphertext generated by user R^* and $\mathcal{A}$ sees an encryption of w_v. In this case we have: $\Pr\left[v' = v | \overline{abort} \wedge \mu = 0\right] = \frac{1}{2} + \varepsilon$. If $\mu = 1$ then the ciphertext $\hat{c}_v$ is a random ciphertext for $\mathcal{A}$ (i.e. $\mathcal{A}$ gains no information about w_v). Hence we have: $\Pr\left[v' \neq v | \overline{abort} \wedge \mu = 1\right] = \frac{1}{2}$. Note that the advantage of $\mathcal{B}$ is the same as the advantage of $\mathcal{A}$. For the first case when the guess of $\mathcal{A}$ is correct $v' = v$, $\mathcal{B}$ will output $\mu' = 0$ and we have $\Pr\left[\mu' = \mu | \overline{abort} \wedge \mu = 0\right] = \frac{1}{2} + \varepsilon$. For the second case when the guess is not correct $v' \neq v$, $\mathcal{B}$ will output $\mu' = 1$ and we have $\Pr\left[\mu' = \mu | \overline{abort} \wedge \mu = 1\right] = \frac{1}{2}$.

Now assume that $\mathcal{B}$ aborts (denoted as *abort*) the simulation when running either TrapGen Query or Challenge phase. In this case $\mathcal{B}$ outputs its guess μ' which is independent of the guess given by $\mathcal{A}$ in Guess phase. Therefore the advantage of $\mathcal{B}$ in the abort case is: $\Pr\left[\mu' = \mu | abort\right] = \frac{1}{2}$. Putting all together we define the overall advantage of the reduction $\mathcal{B}$:

$$\Pr[abort]\Pr\left[\mu' = \mu | abort\right] + \Pr[\overline{abort}](\Pr[\mu = 0]\Pr[\mu' = \mu | \overline{abort} \wedge \mu = 0] + \\ \Pr[\mu = 1]\Pr[\mu' = \mu | \overline{abort} \wedge \mu = 1]) - \frac{1}{2} = \frac{\Pr[\overline{abort}]\varepsilon}{2}.$$

Now we have to define the value of $\Pr[\overline{abort}]$ and give the exact overall advantage of $\mathcal{B}$. Let assume that $\mathcal{A}$ makes at most q queries during TrapGen Query phase and there are n users in the system. Since there is only one user for whom $\mathcal{B}$ cannot answer in TrapGen Query phase, the probability that a query causes $\mathcal{B}$ to abort is at most $\frac{1}{n}$. Since $\mathcal{A}$ can make q queries the overall probability that $\mathcal{A}$ aborts during TrapGen Query phase is $\frac{q}{n}$. Thus the probability that $\mathcal{B}$ does not abort in the TrapGen Query is $1 - \frac{q}{n}$. The probability that $\mathcal{B}$ will not abort in the Challenge phase is at least $\frac{1}{n}$. We now conclude that the reduction $\mathcal{B}$ solves the SXDH assumption with advantage at least $(1 - \frac{q}{n})\frac{1}{n}\frac{\varepsilon}{2}$, as required. □

7.5.2 Trapdoor Indistinguishability

We prove that our scheme is TI-ATK secure as long as the SXDH assumption is intractable. Unlike the ciphertext indistinguishability where the reduction had an SXDH instance from group $\mathbb{G}$, when proving this property the reduction has an SXDH instance from group Γ.

Theorem 7. *Suppose that there exists an adversary $\mathcal{A}$ that can break the trapdoor indistinguishability of the $\mathcal{PKEDS}$ scheme with advantage ε. Then we can construct a polynomial-time reduction $\mathcal{B}$ that solves the* SXDH *assumption with advantage $\frac{\varepsilon}{2}$.*

Proof. The challenger selects a bilinear map $e : \mathbb{G} \times \Gamma \rightarrow \mathbb{G}_T$, and the generator g and γ of the group $\mathbb{G}$ and Γ respectively. Next, the challenger defines $T_0 = \gamma^{ab}$ for a random $a, b \in_R \mathbb{Z}_p$ and picks at random $T_1 \in_R \Gamma$. After flipping a fair coin $\mu \in_R \{0, 1\}$, the challenger gives the SXDH tuple $(\gamma, \gamma^a, \gamma^b, T_\mu) \in \Gamma$ to $\mathcal{B}$. The reduction $\mathcal{B}$ solves the SXDH assumption by running $\mathcal{A}$ as a subroutine:

- Setup: $\mathcal{B}$ publishes $\mathcal{PP}$ in the same way as in the scheme.

- $\mathsf{KeyGen_R}$ Query : $\mathcal{B}$ sets the server's public key $\mathsf{pk}_S = (p_s = \gamma^a)$, where γ^a is extracted from the SXDH instance, and implicitly sets the server's secret-key $\mathsf{sk}_S = a$. $\mathcal{B}$ publishes the server's public key which distribution is identical to the server's public key of the scheme since g and γ are random generators, a is random exponent, all chosen in the same way as in the scheme.

- $\mathsf{KeyGen_R}$ Query : $\mathcal{B}$ answers receiver's key generation queries by computing $(\mathsf{sk}_R, \mathsf{pk}_R) = ((y, \gamma^\alpha), p_r = g^y)$ where $\alpha, y \in_R \mathbb{Z}_p$ are chosen in the same way as in the scheme (each user has different α and y value). The distribution of the receiver's key pair is identical to the distribution of the receiver's key pair of the scheme since g is a random generator, α and y are random exponents, all chosen in the same way as in the scheme.

- $\mathsf{Delegate}$ Query : $\mathcal{A}$ requests a master trapdoor for the receiver R. $\mathcal{B}$ compute random elements $r_1, r_2 \in \mathbb{Z}_p$ and outputs the master trapdoor $t_* = (t_1, t_2, t_3, t_4) = (\gamma^\alpha \cdot p_s^{r_1}, \gamma^{r_1}, \gamma^{y\alpha} \cdot p_s^{r_2}, \gamma^{r_2})$.

  The distribution of the master trapdoor is identical to the distribution of the master trapdoor in the scheme since r_1 and r_2 are random elements from $\mathbb{Z}_p$, the same as in the real scheme.

- $\mathsf{TrapGen}$ Query : $\mathcal{A}$ requests a trapdoor for the pair (R, w). $\mathcal{B}$ picks random $\delta \in \mathbb{Z}_p$ and outputs the trapdoor t_w associated with the keyword w, $t_w = (t_5, t_6) = \big(e(w, \gamma^\alpha) \cdot e(p_r, p_s^\delta), \gamma^\delta\big)$. The distribution of the trapdoor is identical to the distribution of the trapdoor in the scheme since δ is randomly chosen from $\mathbb{Z}_p$, the same as in the real scheme.

- $\mathsf{Challenge}$: $\mathcal{A}$ sends two words w_0 and w_1 to $\mathcal{B}$ and asks for a trapdoor generated by user R^*. $\mathcal{B}$ flips a fair coin $v \in_R \{0, 1\}$, picks at random $\alpha \in \mathbb{Z}_p$ and implicitly sets $\delta = b$ (where b is an exponent from the SXDH instance) and returns the trapdoor to $\mathcal{A}$: $t_{w_v} = (t_5, t_6) = \big(e(w_v, \gamma^\alpha) \cdot e(p_r, T_\mu), \gamma^b\big)$, where $(p_r = g^y,\, y)$ is R^*'s public/private key pair.

- Guess : $\mathcal{A}$ outputs a guess v'.

If $\mu = 0$ and $T_\mu = \gamma^{ab}$, then the generated challenged trapdoor t_{w_v} is a valid trapdoor generated by user R^* and the view of $\mathcal{A}$ is distributed as if it had received the trapdoor from the real scheme. In this case we have: $\Pr\left[v' = v | \mu = 0\right] = \frac{1}{2} + \varepsilon$. If $\mu = 1$ and $T_\mu \in_R \Gamma$, then the generated trapdoor t_{w_v} is an invalid trapdoor. In this case we have: $\Pr[v' \neq v|\mu = 1] = \frac{1}{2}$. Putting all together we define the overall advantage of $\mathcal{B}$:

$$(\Pr[\mu = 0]\Pr[\mu' = \mu|\mu = 0] + \Pr[\mu = 1]\Pr[\mu' = \mu|\mu = 1]) - \frac{1}{2} = \frac{\varepsilon}{2}.$$

□

7.5.3 Ciphertext One-Wayness

In this section we show that our construction is $\mathcal{CTOW}-ATK$ secure in the standard model.

Theorem 8. *The $\mathcal{PKEDS}$ scheme with the message space in $\mathbb{G}$ is $\mathcal{CTOW} - ATK$ secure in the standard model assuming* mCDH *is intractable.*

Proof. The challenger selects a bilinear map $e : \mathbb{G} \times \Gamma \to \mathbb{G}_T$, and the generator g and γ of the group $\mathbb{G}$ and Γ respectively. Then, it picks at random $a, b \in \mathbb{Z}_p$, and gives mCDH tuples $(g, g^a, g^b) \in \mathbb{G}$ and $(\gamma, \gamma^b) \in \Gamma$ to the reduction $\mathcal{B}$. The goal of $\mathcal{B}$ is to solve the mCDH assumption and acts as $\mathcal{A}$'s challenger as follows:

- Setup : $\mathcal{B}$ publishes $\mathcal{PP}$ in the same way as in the scheme.
- $\mathsf{KeyGen_S}$ Query: $\mathcal{B}$ generates the server's key pair $(\mathsf{sk}_S, \mathsf{pk}_S) = (x, p_s = \gamma^x)$, where $x \in_R \mathbb{Z}_p$ is chosen in the same way as in the scheme. $\mathcal{B}$ publishes the server's key pair. The distribution of the server's key pair is identical to the server's key pair of the scheme since g and γ are random generators, and x is a random exponent, all chosen in the same way as in the scheme.
- $\mathsf{KeyGen_R}$ Query: $\mathcal{B}$ answers receiver's key generation queries by computing $(\mathsf{sk}_R, \mathsf{pk}_R) = ((y, \gamma^\alpha), p_r = g^y)$ where $\alpha, y \in_R \mathbb{Z}_p$ are chosen in the same way as in the scheme (each user has different α and y value). If the query is for R^*, $\mathcal{B}$ sets the public key equal to $p_r = g^a$ (this parameter is extracted from the mCDH instance). Note that $\mathcal{B}$ does not know the private key of R^* (the reduction does not know a). The distribution of the receiver's key pair is identical to the distribution of the receiver's key pair of the scheme since g is a random generator, α, a and y are random exponents, all chosen in the same way as in the scheme.
- $\mathsf{Delegate\ Query}$: $\mathcal{A}$ requests a master trapdoor for the receiver R. $\mathcal{B}$ computes random elements $r_1, r_2 \in \mathbb{Z}_p$ and outputs the master trapdoor $t_* = (t_1, t_2, t_3, t_4) = (\gamma^\alpha \cdot p_s^{r_1}, \gamma^{r_1}, \gamma^{y\alpha} \cdot p_s^{r_2}, \gamma^{r_2})$. If $R = R^*$ then $y = a$. The distribution of the master trapdoor is identical to the distribution of the master trapdoor in the scheme since r_1 and r_2 are random elements from $\mathbb{Z}_p$, the same as in the real scheme.
- $\mathsf{TrapGen\ Query}$: $\mathcal{A}$ requests a trapdoor for the pair (R, w). $\mathcal{B}$ picks random $\delta \in \mathbb{Z}_p$ and outputs the trapdoor $t_w = (t_5, t_6) = (e(w, \gamma^\alpha) \cdot e(p_r, p_s^\delta), \gamma^\delta)$. The distribution of the trapdoor is identical to the distribution of the trapdoor in the scheme since δ is chosen at random from $\mathbb{Z}_p$, the same as in the real scheme.

- Challenge : The reduction picks at random $c' \in \mathbb{Z}_p$, computes $\frac{g^{c'}}{g^a} = g^{\bar{c}}$ (thus, $c' = a + \bar{c}$), implicitly sets $w^* = g^{b\bar{c}}$ and outputs the challenge ciphertext $\hat{c}_{w^*} = (c_1, c_2) = (g^{bc'}, g^b)$ and the challenge trapdoor $t_{w^*} = (t_5, t_6) = (e(g^{\bar{c}}, \gamma^{b\alpha}) \cdot e(p_r, p_s^{\delta}), \gamma^{\delta})$. The challenge ciphertext $\hat{c}_{w^*} = (c_1, c_2) = (g^{bc'}, g^b) = (g^{b\bar{c}} \cdot g^{ab}, g^b)$ is a valid encryption of the message $w^* = g^{b\bar{c}}$ under the public key of R^* and has the same distribution as the ciphertext in the real scheme. The same holds for the trapdoor t_{w^*}.

- Output : At the end of the security-game, $\mathcal{A}$ outputs the message w'.

$\mathcal{B}$ checks whether $e(w', \gamma) \overset{?}{=} e(g^a, \gamma^b)$. If so, then $\mathcal{A}$ has decrypted the challenge ciphertext and $\mathcal{B}$ uses the output of $\mathcal{A}$ to solve the mCDH assumption: $g^{ab} = \frac{c_1}{w'}$, which will contradict the assumption and proof the theorem. □

7.6 Applications

In this section we illustrate two applications of the $\mathcal{PKEDS}$ scheme: to detect encrypted malware and to forward encrypted emails.

Detecting Encrypted Malware

A polymorphic virus uses encryption to modify its form as it spreads in such a way that different infected files have different byte-strings (each file is encrypted with a different key) [73]. The polymorphic virus instance is divided into three parts: the decryption algorithm, the decryption key and the encrypted virus. The decryption algorithm uses the decryption key in order to decrypt and run the virus. The fact that polymorphic viruses store the decryption key within each virus instance, makes them detectable by a virus scanner. As pointed out in the introduction, in this chapter we consider attackers who use encryption to hide malware even in a more powerful way. Namely, in our attack scenario an attacker does not include the decryption key within the encrypted data: indeed, an attacker does not know the decryption key which belongs to the receiver. Hence, the virus instance that we consider contains only the encrypted malware and the decryption algorithm.

In the proposed $\mathcal{PKEDS}$, the server can use the master trapdoor, a ciphertext and a malware signature, to check whether the ciphertext contains the malware signature, without decrypting it. This kind of detection is known as signature-based detection and is performed by most existing antivirus software packages, which maintain a database with known malware signatures and check whether the scanned data has the same signature as the one of the signatures stored in the database. If the signatures match, then a malware is found and the antivirus takes further steps to quarantine, repair or delete the infected data. We assume that the server has a database with known malware signatures and for each signature, using the master trapdoor t_*, it creates a trapdoor and checks whether the trapdoor and the scanned ciphertext have the same signature. If so, then a malware is detected and the server takes further steps to quarantine or delete the ciphertext, otherwise the ciphertext is clean and is forwarded to the receiver. The crucial property of the scheme is that ciphertexts are both searchable and decryptable, thus the server can search every part of the ciphertext for a possible malware. Note that allowing the server to search the encrypted data does not mean that the server can perform any kind of intrusion detection. For instance, 0-day attacks cannot be detected through signature-based detection.

Roschke *et al.* [9] propose a technique which detects malicious content in the encrypted data. The solution presented in [9] is based on the IBE [27] scheme and suffers from the key escrow property where the compromise of the master secret key breaks the whole system. The conceptual difference between $\mathcal{PKEDS}$ and the approach presented in [9] is, that the scheme in [9] uses the master secret key of the IBE to decrypt the ciphertext and then uses the virus scanner to scan the plaintext, while in $\mathcal{PKEDS}$ the scanning can be done in the encrypted data, without having to decrypt it.

Forwarding Encrypted Emails

The original motivation for a PEKS scheme is to allow an email server to categorize user encrypted emails based on keywords contained in the message text. Using this property, Alice can create trapdoors t_{work} and t_{family}, and instruct the server to forward her encrypted emails tagged with the word "work" to her secretary and encrypted emails tagged with the word "family" to one of her family members. Waters *et al.* [105] showed that PEKS schemes can also be used to build a searchable audit log which is encrypted. The $\mathcal{PKEDS}$ scheme adds the decryption property to the PEKS scheme and as such can be used in every application that PEKS can be used. The $\mathcal{PKEDS}$ scheme has the following additional advantages compared to PEKS:

- $\mathcal{PKEDS}$ can be used on its own, without employing an additional PKE encryption scheme, to send encrypted messages in an open environment. This property inherently brings an additional advantage - each word of the message becomes searchable by the server. For instance, to encrypt a message m which consists of words $w_1, ..., w_k$, the sender generates the ciphertext:

  $$(\mathsf{Encrypt}_{\mathcal{PKEDS}}(w_1)||...||\mathsf{Encrypt}_{\mathcal{PKEDS}}(w_k)),$$

  where $\mathsf{Encrypt}_{\mathcal{PKEDS}}$ is the encryption algorithm for the $\mathcal{PKEDS}$ scheme. It is clear that to reveal the message m, the receiver has to decrypt each searchable ciphertext separately. From a computational point of view, using $\mathcal{PKEDS}$ alone might be expensive since it requires a number of $\mathcal{PKEDS}$ ciphertexts linear in the number of words in the document. Note that in PEKS the server can search only for keywords and this might be a problem for scenarios when the original message might contain some words that appear in the receiver's query but do not appear in the keyword list, and as a result the server will not forward these documents to the receiver. For example, the receiver might ask the server to forward all documents that contain the word "Friday". In PEKS, if "Friday" is not a keyword in a document then the server will not forward the document to the receiver. In $\mathcal{PKEDS}$ every word is searchable and the server will forward to the receiver all documents containing the word "Friday", regardless if "Friday" is keyword or not.

- $\mathcal{PKEDS}$ can be used in the same way as PEKS is used, namely, use $\mathcal{PKEDS}$ to encrypt only keywords in addition to a non-searchable PKE scheme which encrypts the original message. For instance, to encrypt a message m with keywords $w_1, ..., w_k$, the sender generates the ciphertext:

$$(\mathsf{Encrypt}_{\mathsf{PKE}}(m)||\mathsf{Encrypt}_{\mathcal{PKEDS}}(w_1)||...||\mathsf{Encrypt}_{\mathcal{PKEDS}}(w_k)),$$

where $\mathsf{Encrypt}_{\mathsf{PKE}}$ is a regular encryption function for the PKE scheme and $\mathsf{Encrypt}_{\mathcal{PKEDS}}$ is the encryption function for the $\mathcal{PKEDS}$ scheme. Unlike PEKS which does not guarantee any relation between keywords (encrypted under PEKS) and the original message (encrypted under PKE), $\mathcal{PKEDS}$ guarantees this *relation* since it allows the receiver to decrypt the searchable ciphertext and to check whether the keywords indeed describe the original message. Another benefit is that the receiver can categorize his messages according to the keywords, unlike in PEKS where the receiver cannot categorize his messages since the searchable ciphertext is not decryptable and consequently the receiver, without decrypting the ciphertext, does not know which keywords describe the message. From a computational point of view, using $\mathcal{PKEDS}$ in addition to another PKE scheme would require a number of $\mathcal{PKEDS}$ ciphertexts linear in the number of *keywords* in the message, the same as in PEKS.

7.7 Conclusion

In this chapter we propose a new scheme called Public-Key Encryption with Delegated Search ($\mathcal{PKEDS}$) with a security proof in the standard model. In the proposed scheme the private key holder creates a master trapdoor t_* and delegates to another entity (i.e. the server) the ability to search ciphertexts intended for the receiver without decrypting it. The main property of the scheme is that ciphertexts are both searchable and decryptable, thus the scheme can be used to search not only for keywords describing the document, but also search for words inside the document. The proposed scheme also allows the receiver to provide the server with a special key (a.k.a. trapdoor t_w) associated with a specific word w, such that it enables the server to test whether the word w is in the ciphertext. As an application, we show how $\mathcal{PKEDS}$ can be used for detecting encrypted malware and for forwarding encrypted email.

Chapter 8

Conclusions

In this chapter we summarize the main contributions of the thesis. We also give some directions for future work.

8.1 Conclusions and future work

In this thesis we focus on designing novel cryptographic schemes that control access to outsourced data by means of encryption. In general, encryption protects the data when it *travels* from the sender to the recipient and it protects the data when it *rests* in an *honest-but-curious* server. In this thesis, encryption ensures that only authorized users specified during the encryption phase are able to access the data in clear.

In the introductory chapter we formulated the following main research question:

"How to construct cryptographic schemes that can enforce distributed data access control efficiently in dynamic environments?"

We answer this question by focusing on three primitives: proxy re-encryption (PRE), attribute-based encryption (ABE) and traditional public-key encryption (PKE). We propose new schemes which add new properties to PRE, ABE and PKE. For PRE we add the property of expressing fine grained AC policies, for ABE we add properties of key revocation and updating AC policies and for PKE we add the property of delegated search. Unfortunately, adding new properties does not mean that we can re-use existing schemes. Instead, adding a property usually requires the construction of new schemes. In addition, the security definition (defined through a security game) of the original primitive needs to be extended, such that it addresses the new property. For example, when we add key revocation to ABE, the original security definition of the ABE should be changed such that it has to take into account new attacks that might come from revoked users. We divide the main research question into three sub-questions; one sub-question per primitive. For the PRE primitive we

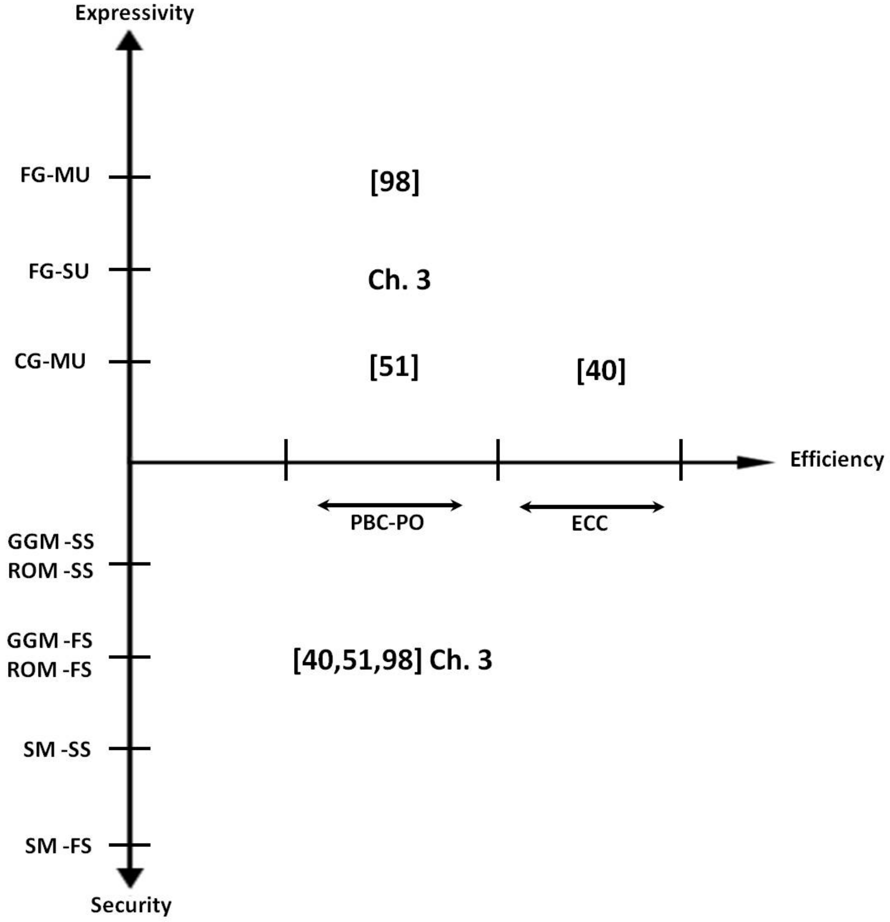

Figure 8.1: *Visualization of* TID-PRE *presented in Chapter 3 (abbreviated as Ch. 3) and its possible extensions.*

defined the following sub-question:

Q1.*How to construct a* PRE *scheme which can support fine grained AC policies, without sacrificing efficiency?*

We address this sub-question in Chapter 3 where we propose a new PRE scheme, called TID-PRE, that allows the delegator to categorize messages into different types and delegate the decryption right of each type to the delegatee through a proxy. In Figure 8.1 we visualise our contribution from Chapter 3, the related work and possible extensions. We show that from an expressivity point of view our scheme supports fine grained AC policies, whereas the most relevant related scheme presented in [51] support course grained AC policies. Our scheme belongs to the single-user (SU) setting since the proxy can re-encrypt only ciphertexts generated by the delegator. Qiang [98] extended our published work [5] and proposes a new TID-PRE scheme that belongs to the multi-user (MU) setting where the proxy can re-encrypt ciphertexts created both from the delegator and delegatee.

From an efficiency point of view, our scheme uses pairing based cryptography with prime-order (PBC-PO) groups, and the efficiency of our scheme, as shown in Section 3.3.1, is *exactly* the same as the efficiency of the scheme presented by [51]. This implies that we were able to provide a scheme that supports fine grained policies without sacrificing the efficiency. In general, the efficiency of the PBC schemes is approximately the same as the efficiency of schemes that are based on the RSA assumption [48]. However, PBC schemes are less efficient than elliptic curve cryptography (ECC) schemes. For instance, to achieve 256 bit security, for ECC we have to perform operations in a 512-bit finite field but for PBC we have to perform operations in a 15 360-bit finite field [48]. The scheme presented by [40] is more efficient than our scheme, but it does not support fine grained policies and is not identity-based (i.e. the public key of the recipient has to be certified by the party that encrypts the message).

In Figure 8.1 we also show the position of our scheme with respect to security. As explained throughout the thesis, having a full-security (FS) proof is better than having a selective-security (SS) proof, since FS better reflects the real world. In addition, having a security proof in the standard model (SM) is better than a security proof in generic group model (GGM) or the random oracle model (ROM). The reason for this, as explained in Section 2.5, is that SM bases the security proof only on complexity assumptions whereas GGM and ROM assume the existence of some idealized functions (e.g. ROM assumes that hash functions are indistinguishable from random functions). Note that there are many other properties (other than Expressivity, Efficiency and Security) based on which we can categorize PRE schemes. However, we chose these properties since in the thesis we focus on providing fine grained schemes without sacrificing efficiency, and we want our schemes to be secure in one of the three models presented in Sections 2.4 and 2.5. Based on the aforementioned observation, we define the following problem as an interested future work:

FW1*. How to construct a multi-use* TID-PRE *scheme that has a security proof in the standard model and an efficiency similar to the ECC schemes?*

In Chapters 4,5,6 we focus on the ABE primitive. For the ABE primitive we defined the following sub-question:

Q2*. How to construct* ABE *schemes which are* ***efficient****, and support* ***revoking keys*** *and* ***updating*** *AC policies*?

A review of existing CP-ABE schemes shows that they are expensive when encrypting messages and decrypting ciphertexts, and hence not suitable for resource-constrained devices. In Chapter 4 we propose two efficient CP-ABE
schemes: B-CP-ABE and E-CP-ABE. In our schemes, first the encryptor defines an AC policy over attributes, and then encrypts the data according to that policy. The

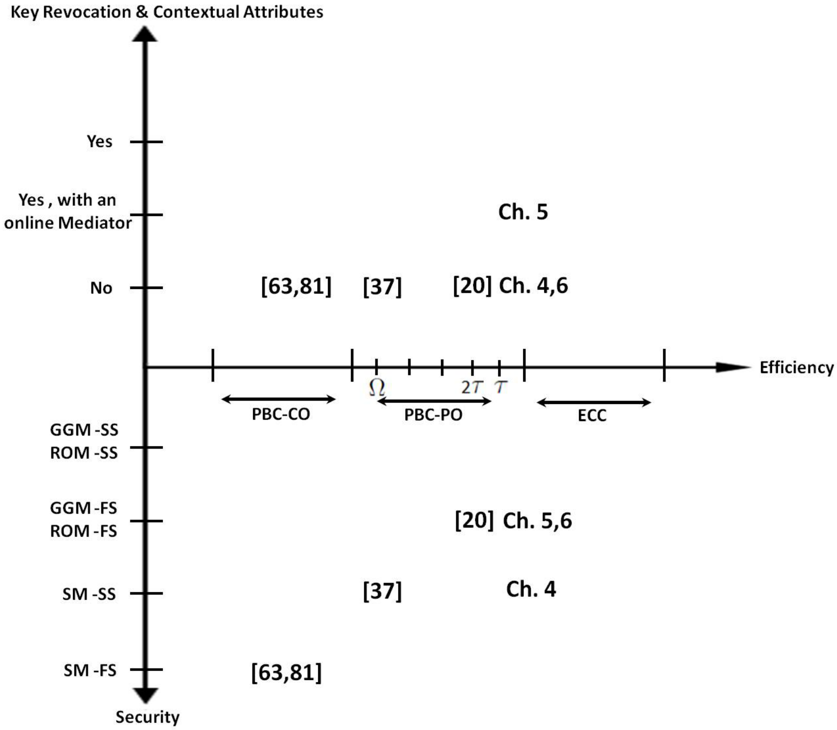

Figure 8.2: *Visualization of* ABE *schemes presented in Chapters 4,5 and 6 (abbreviated as Ch. 4,5,6) and their possible extensions.*

AC policy of the B-CP-ABE is a predicate constructed from monotone Boolean formulas while the AC policy of the E-CP-ABE is a predicate, which in addition to the monotone Boolean formula contains circuits of threshold gates. In Chapter 5 we focus on key revocation, where we propose a new scheme called mediated Ciphertext-Policy Attribute-Based Encryption (mCP-ABE), which supports different types of key revocations such as: a) revoking a subset of attributes from a user attribute set without influencing the sets of other users, b) revoking all the attribute set from some users without influencing other users or c) revoking a subset from the set of user attributes from all users. In Chapter 6 we focus on updating the AC policy of the ciphertext without decrypting it. We propose a new scheme called Ciphertext-Policy Attribute-Based Proxy Re-encryption (CP-ABPRE), which allows users who satisfy the AC policy of the ciphertext to generate a re-encryption key and forward it to the semi-trusted proxy. Once the proxy gets the re-encryption key, it updates the AC policy by re-encrypting the ciphertext. After the ciphertext gets re-encrypted, only users who satisfy the new AC policy can decrypt the data.

In Figure 8.2 we visualize our contributions from chapters 4,5,6, the related work and possible extensions. From an efficiency point of view, all schemes presented in Chapters 4,5,6 are PBC-PO. Note that we place our schemes and the Bethencourt *et al.* [20] scheme in the same setting, that is PBC-PO. However, this does not mean that they have exactly the same efficiency; this only indicates that they perform operations over the same finite field, but PBC-PO does not indicate the *number of operations*. As shown in Figure 8.2 (also shown in Tables 4.1 and 4.2), the scheme presented in Chapter 4 (schemes presented in Chapters 5 and 6 have almost the same efficiency as the scheme presented in Chapter 4) is more efficient in the number of operations compared to [37] and [20]. For example, the scheme from Chapter 4 in the Encrypt phase requires around $|\tau|$ operations in $\mathbb{G}_T$ (where τ is the AC policy), while [20] requires around $2|\tau|$ operations in $\mathbb{G}_T$ and [37] requires around $|\Omega|$ operations in $\mathbb{G}_T$ (where $|\Omega|$ is the total number of attributes in the scheme). Recently, there has been a new development in the literature in proposing pairing based ABE schemes that use composite-order groups [63, 81] (we denote this as PBC-CO). Composite-order groups are less efficient than prime-order groups, therefore, having a scheme with prime-order groups is more desirable than having a scheme in composite-order groups.

We also show the place of our ABE schemes with respect to security. The ABE schemes presented in Chapter 4 have a Selective Security (SS) proof in Standard Model (SM) whereas the ABE schemes presented in Chapters 5 and 6 have a Full Security (FS) proof in Generic Group Model (GGM). Only recently Okamoto and Takashima [81] and Lewko *et al.* [63] proposed pairing based ABE schemes with a Full Security (FS) proof in the Standard Model (SM). We define the following problem as a future work:

FW2. *How to construct a (proxy re-encryption)* ABE *scheme that has a security proof in the standard model and an efficiency similar to the ECC schemes?*

Indeed, it would be interesting to construct ABE schemes with a security proof in ROM or GGM and an efficiency similar to the ECC schemes.

In Figure 8.2 we show that the scheme presented in Chapter 5 supports key revocation. The crucial component of the proposed scheme is the mediator. The scheme splits the secret key into two parts - one part for the mediator and the other part for the decryptor. Whenever a user wants to decrypt a ciphertext, the user has to receive a decryption token from the mediator. If the user is revoked, then the mediator will not issue a decryption token to the user, thus the user part of the secret key becomes useless and the user implicitly gets revoked. A distinctive feature this scheme is that it can enforce context attributes such as time and location. However, since the mediator has to be contacted during each decryption, the scheme requires communication between the decryptor and the mediator. Hence, it would have been better to eliminate these communication costs, while still supporting both the key revocation and contextual attributes. We derive the following problem as a future work:

FW3. *How to construct an* ABE *scheme that has a security proof in the standard model, with efficiency similar to the ECC schemes, and that supports key revocation* *and contextual attributes without involving an online mediator?*

All schemes presented in Chapters 3,4,5,6 control access to the data by using encryption techniques. In Chapter 7 we focus on traditional public-key encryption (PKE) and consider a scenario when encryption is used by attackers to compromise the user. For instance, an attacker can use the user's public key to encrypt malware, which once decrypted can compromise all the user data regardless whether the data is stored in the user's local computer or whether is outsourced to another party. We define the following sub-question:

***Q3**. How to delegate the power to search in encrypted data?*

We answer this question by proposing a new scheme called Public-key Encryption with Delegated Search ($\mathcal{PKEDS}$). The crucial property of the scheme is that ciphertexts are both decryptable and searchable. This enables users to delegate to another entity the power to search the entire ciphertexts (i.e. not only the metadata part) for specific malware without decrypting it.

In Figure 8.3 we present the position of $\mathcal{PKEDS}$ with respect to Searchability, Efficiency and Security. As shown, our scheme is implemented using parings over prime order groups (i.e. PBC-PO). From the searchability point of view, the scheme allows the private key holder to generate trapdoors and delegate to another party (e.g. a mail server) the power to search for *any* word over its encrypted data (unlike in [24, 29] where the server can search only for the metadata). Probably, a more useful primitive would perhaps allow the key holder to delegate to the server the power to generate trapdoors for a *restricted class of words*. From the security point of view, our scheme achieves FS in SM. As mentioned above, having a security proof in the SM is preferable over a security proof in GGM or RM. Based on this observation, we define the following problem as a future work:

***FW4**. How to construct a $\mathcal{PKEDS}$ scheme with restricted class delegation, efficiency similar to the ECC schemes, and security proof in the standard model?*

Other Future Work

The work presented in this thesis opens a number of future research directions. In Chapters 4,5,6 we propose CP-ABE schemes where the AC policy associated with the ciphertext is in clear. The reason for this is that because the decryptor has to look at the AC policy in order to decide which attributes to use and decrypt the ciphertext. A partially-anonymous CP-ABE schemes is presented by Nishide *et al.* [79] and an inefficient CP-ABE scheme with high computational complexity is presented by Li *et al.* [64]. We see the following problem as interesting future work:

***FW5**. How to construct fully-anonymous* CP-ABE *schemes where the size of the ciphertext is linear in the size of the AC policy?*

In this thesis we focus on access control of outsourced data through encryption. However, with the recent development in cloud computing, individuals and companies might start to outsource the computation. Computation outsourcing is useful for

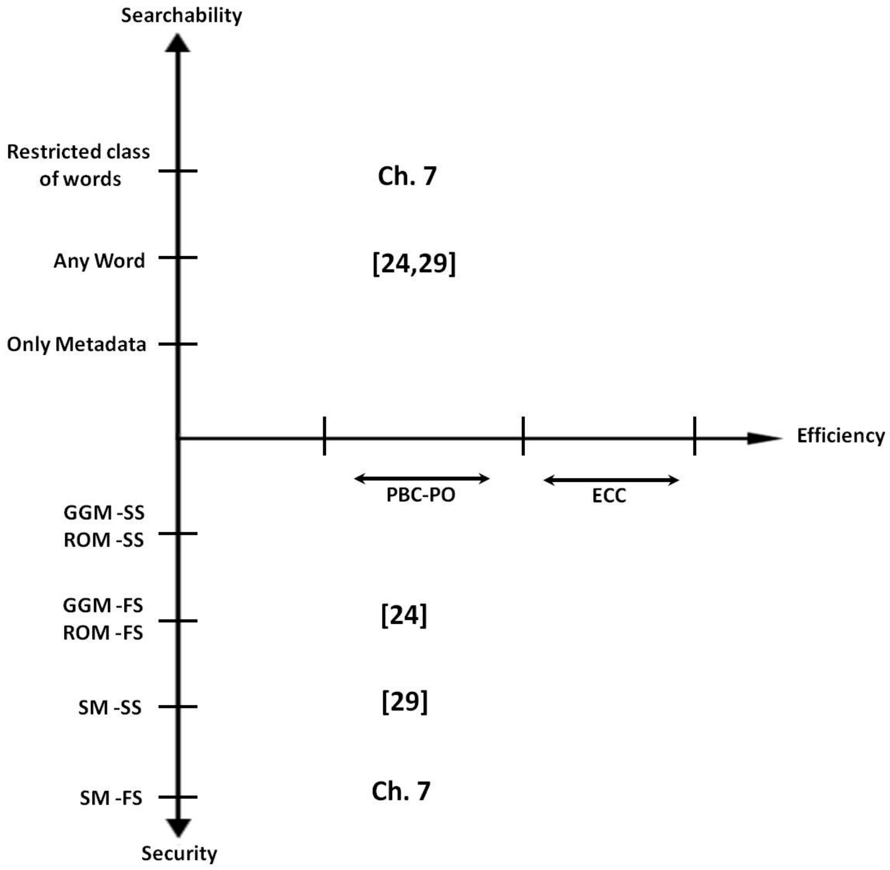

Figure 8.3: *Visualization of $\mathcal{PKEDS}$ presented in Chapter 7(abbreviated as Ch. 7) and its possible extensions.*

situations when a device needs to perform an operation but it does not have the computation power to execute it. In addition, computation outsourcing is cost-effective since individuals or organization do not have to buy a new devices to perform a particular operation.

A relevant example of outsourcing the computation is to outsource the decryption cost such that the cloud performs most of the computations, while the thin client performs only a small part of the computation. For our proposed schemes, we envision *computation outsourcing* as a possible solution that can considerably reduce decryption costs performed by the recipient. For instance, in our ABE schemes the pairing computations performed during Decrypt depend on the size of the access policy associated with the ciphertext. It would have been better for the client to delegate all pairing computations to the cloud, such that the client computes only less-expensive operations. The main challenge to build such scheme is to hide as much as possible information from the cloud, while still leveraging its computation power. We define the following problem as an interesting future work:

FW6. *How to outsource the computation of* PRE, ABE *and* PKE *to the cloud?*

While we focus on three cryptographic primitives: PRE, ABE and PKE, this list is not complete with respect to enforcing access policies. Recently introduced functional encryption schemes, including hidden vector encryption and inner product predicate encryption, can enforce expressive AC policies as well. Our extensions to ABE, such as key revocation and updating access policies, are also needed for these primitives. At the time of writing this thesis there is no scheme or security definition for these extensions in hidden vector encryption and inner product predicate encryption. Thus, we see this as an open problem and an interesting area of future work. We define the following problem as a future work:

FW7. *How to construct hidden-vector encryption and inner-product predicate encryption schemes that can support key revocation and that can allow to update the vector associated with the ciphertext without decrypting it?*

Publications by the Author

[1] L. Ibraimi, M. Asim, and M. Petković. An encryption scheme for a secure policy updating. In S. Katsikas and P Samarati, editors, *Proceedings of International Conference on Security and Cryptography – SECRYPT 2010*, pages 399–408. SciTePress, 2010.

[2] L. Ibraimi, A. Muhammad, and M. Petković. Secure management of personal health records by applying attribute-based encryption. In *Proceedings of the 6th International Workshop on Wearable Micro and Nano Technologies for Personalized Health – pHealth 2009*, pages 71 –74. IEEE, 2009.

[3] L. Ibraimi, S. Nikova, P. Hartel, and W. Jonker. Public-key encryption with delegated search. In J. Lopez and G. Tsudik, editors, *Proceedings of the 9th International Conference on Applied Cryptography and Network Security – ACNS 2011*, volume 6715 of *LNCS*, pages 532–549. Springer, 2011.

[4] L. Ibraimi, M. Petković, S. Nikova, P. Hartel, and W. Jonker. Mediated ciphertext-policy attribute-based encryption and its application. In Y. H. Youm and M. Yung, editors, *Proceedings of the 10th International Workshop on Information Security Applications – WISA 2009*, volume 5932 of *LNCS*, pages 309–323. Springer, 2009.

[5] L. Ibraimi, Q. Tang, P. Hartel, and W. Jonker. A type-and-identity-based proxy re-encryption scheme and its application in healthcare. In W. Jonker and M. Petković, editors, *Porceedings of the 5th VLDB Workshop on Secure Data Management – SDM 2008*, volume 5159 of *LNCS*, pages 185–198. Springer, 2008.

[6] L. Ibraimi, Q. Tang, P. Hartel, and W. Jonker. Efficient and provable secure ciphertext-policy attribute-based encryption schemes. In F. Bao, H Li, and G. Wang, editors, *Proceedings of the 5th International Conference on Information Security Practice and Experience – ISPEC 2009*, volume 5451 of *LNCS*, pages 1–12. Springer, 2009.

[7] L. Ibraimi, Q. Tang, P. Hartel, and W. Jonker. Exploring type-and-identity-based proxy re-encryption scheme to securely manage personal health records. *International Journal of Computational Models and Algorithms in Medicine (IJCMAM)*, 1(2):1–21, 2010.

[8] M. Petković and L. Ibraimi. *E–Health, Assistive Technologies and Applications for Assisted Living: Challenges and Solutions*, chapter Privacy and Security in e–Health Applications, pages 23–48. IGI Global, 2011.

[9] S. Roschke, L. Ibraimi, F. Cheng, and C. Meinel. Secure communication using identity-based encryption. In De B. Decker and I. S. Bichl, editors, *Proceedings of Communications and Multimedia Security – CMS 2010*, volume 6109 of *LNCS*, pages 256–267. Springer, 2010.

Other References

[10] M. Abdalla, M. Bellare, D. Catalano, E. Kiltz, T. Kohno, T. Lange, J. Malone-Lee, G. Neven, P. Paillier, and H. Shi. Searchable encryption revisited: Consistency properties, relation to anonymous IBE, and extensions. In V. Shoup, editor, *Proceedings of the 25th Annual International Cryptology Conference, Advances in Cryptology – CRYPTO 2005*, volume 3621 of *LNCS*, pages 205–222. Springer, 2005.

[11] M. Abdalla, M. Bellare, D. Catalano, E. Kiltz, T. Kohno, T. Lange, J. Malone-Lee, G. Neven, P. Paillier, and H. Shi. Searchable encryption revisited: consistency properties, relation to anonymous IBE, and extensions. *Journal of Cryptology*, 21(3):350–391, 2008.

[12] L. Adleman. A subexponential algorithm for the discrete logarithm problem with applications to cryptography. In *Proceedings of the 20th Annual Symposium on Foundations of Computer Science – FOCS 1979*. IEEE, 2008.

[13] G. Ateniese, K. Fu, M. Green, and S. Hohenberger. Improved proxy re-encryption schemes with applications to secure distributed storage. *ACM Transactions on Information and System Security (TISSEC)*, 9(1):1–30, 2006.

[14] J. Baek, R. Safavi-Naini, and W. Susilo. On the integration of public key data encryption and public key encryption with keyword search. In S. K. Katsikas, J. Lopez, M. Backes, S. Gritzalis, and B. Preneel, editors, *Proceedings of the 9th Information Security Conference – ISC 2006*, volume 4176 of *LNCS*, pages 217–232. Springer, 2006.

[15] J. Baek, R. Safavi-Naini, and W. Susilo. Public key encryption with keyword search revisited. In O. Gervasi, B. Murgante, A. Lagan, D. Taniar, Y. Mun, and L. M. L. Gavrilova, editors, *Proceedings of International Conference on Computational Science and Its Applications – ICCSA 2008*, volume 5072 of *LNCS*, pages 1249–1259. Springer, 2008.

[16] L. Ballard, M. Green, D. B. Medeiros, and F. Monrose. Correlation-resistant storage via keyword-searchable encryption. Technical report, Cryptology ePrint Archive, Report 2005/417, Available at http://eprint.iacr.org/2005/417, 2005.

[17] A. Beimel. *Secure Schemes for Secret Sharing and Key Distribution*. PhD thesis, Department of Computer Science, Israel Institute of Technology, Technion, 1996.

[18] M. Bellare and P. Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *Proceedings of the 1st ACM Conference on Computer and Communications Security – CCS 1993*, pages 62–73. ACM, 1993.

[19] J. Benaloh and J. Leichter. Generalized secret sharing and monotone functions. In Sh. Goldwasser, editor, *Proccedings of the 8th Annual International Cryptology Conference, Advances in Cryptology – CRYPTO 1988*, volume 403 of *LNCS*, pages 27–35. Springer, 1990.

[20] J. Bethencourt, A. Sahai, and B. Waters. Ciphertext-policy attribute-based encryption. In *IEEE Symposium on Security and Privacy 2007*, pages 321–334. IEEE, 2007.

[21] M. Blaze, G. Bleumer, and M. Strauss. Divertible protocols and atomic proxy cryptography. In K. Nyberg, editor, *Proceedings of International Conference on the Theory and Application of Cryptographic Techniques, Advances in Cryptology – EUROCRYPT 1998*, volume 1403 of *LNCS*, pages 127–144. Springer, 1998.

[22] D. Boneh and X. Boyen. Efficient selective-id secure identity-based encryption without random oracles. In C. Cachin and J. Camenisch, editors, *Proceedings of the 23rd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Advances in Cryptology – EUROCRYPT 2004*, volume 3027 of *LNCS*, pages 223–238. Springer, 2004.

[23] D. Boneh and X. Boyen. Efficient selective-ID secure identity based encryption without random oracles. In C. Cachin and J. Camenisch, editors, *Proceedings of the 23rd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Advances in Cryptology – EUROCRYPT 2004*, volume 3027 of *LNCS*, pages 223–238. Springer, 2004.

[24] D. Boneh, G. Di Crescenzo, R. Ostrovsky, and G. Persiano. Public key encryption with keyword search. In C. Cachin and J. Camenisch, editors, *Proceedings of the 23rd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Advances in Cryptology – EUROCRYPT 2004*, volume 3027 of *LNCS*, pages 506–522. Springer, 2004.

[25] D. Boneh, X. Ding, and G. Tsudik. Fine-grained control of security capabilities. *ACM Transactions on Internet Technology (TOIT)*, 4(1):60–82, 2004.

[26] D. Boneh, X. Ding, G. Tsudik, and C. M. Wong. A method for fast revocation of public key certificates and security capabilities. In *Proceedings of the*

10th conference on USENIX Security Symposium – SSYM 2001, pages 22–22, Berkeley, CA, USA, 2001. USENIX Association.

[27] D. Boneh and M. Franklin. Identity-based encryption from the Weil pairing. In J. Kilian, editor, *Proceedings of the 21st Annual International Cryptology Conference, Advances in Cryptology - CRYPTO 2001*, volume 2139 of *LNCS*, pages 213–229. Springer, 2001.

[28] D. Boneh, B. Lynn, and H. Shacham. Short signatures from the Weil pairing. In C. Boy, editor, *Proceedings of 7th International Conference on the Theory and Application of Cryptology and Information Security, Advances in Cryptology – ASIACRYPT 2001*, volume 2248 of *LNCS*, pages 514–532. Springer, 2001.

[29] D. Boneh and B. Waters. Conjunctive, subset, and range queries on encrypted data. In S. P. Vadhan, editor, *Proceedings of the 4th IACR Theory of Cryptography Conference – TCC 2007*, volume 4392 of *LNCS*, pages 535–554. Springer, 2007.

[30] X. Boyen and B. Waters. Anonymous hierarchical identity-based encryption (without random oracles). In C. Dwork, editor, *Proceedings of the 26th Annual International Cryptology Conference, Advances in Cryptology – CRYPTO 2006*, volume 4117 of *LNCS*, pages 290–307. Springer, 2006.

[31] R. Canetti, O. Goldreich, and S. Halevi. The random oracle methodology, revisited. *Journal of the ACM (JACM)*, 51(4):557–594, 2004.

[32] R. Canetti, S. Halevi, and J. Katz. Chosen-ciphertext security from identity-based encryption. In C. Cachin and J. Camenisch, editors, *Proceedings of the 23rd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Advances in Cryptology – EUROCRYPT 2004*, volume 3027 of *LNCS*, pages 207–222. Springer, 2004.

[33] M. Chase. Multi-authority attribute based encryption. In *Proceedings of the 4th Theory of Cryptography Conference, Theory of Cryptography – TCC 2007*, volume 4392 of *LNCS*, pages 515–534. Springer, 2007.

[34] M. Chase. Multi-authority attribute based encryption. In P. S. Vadhan, editor, *Proceedings of the 4th Theory of Cryptography Conference, Theory of Cryptography – TCC 2007*, volume 4392 of *LNCS*, pages 515–534. Springer, 2007.

[35] S. Chatterjee and A. Menezes. On cryptographic protocols employing asymmetric pairings – the role of ψ revisited. Technical report, Cryptology ePrint Archive, Report 2009/480, Available at http://eprint.iacr.org/2009/480, 2009.

[36] L. Chen. An interpretation of identity-based cryptography. In A. Aldini and R. Gorrieri, editors, *Proceedings of Foundations of Security Analysis and Design IV – FOSAD 2006/2007 Tutorial Lectures*, volume 4677 of *LNCS*, pages 183–208. Springer, 2007.

[37] L. Cheung and C. Newport. Provably secure ciphertext policy ABE. In *Proceedings of the 14th ACM conference on Computer and communications security – CCS 2007*, pages 456–465. ACM, 2007.

[38] C. Cocks. An identity based encryption scheme based on quadratic residues. In B. Honary, editor, *Proceedings of the 8th IMA International Conference Cirencester on Cryptography and Coding*, volume 2260 of *LNCS*, pages 360–363. Springer, 2001.

[39] J. Daemen and V. Rijmen. *The design of Rijndael: AES–the advanced encryption standard*. Springer, 2002.

[40] R. Deng, J. Weng, S. Liu, and K. Chen. Chosen-ciphertext secure proxy re-encryption without pairings. In M. Franklin, L. Hui, and D. Wong, editors, *Proceedings of the 7th International Conference on Cryptology and Network Security – CANS 2008*, volume 5339 of *LNCS*, pages 1–17. Springer, 2008.

[41] A. Dent. Adapting the weaknesses of the random oracle model to the generic group model. In Y. Zheng, editor, *Proceedings of 8th International Conference on the Theory and Application of Cryptology and Information Security, Advances in Cryptology – ASIACRYPT 2002*, volume 2501 of *LNCS*, pages 95–104. Springer, 2002.

[42] Y. Desmedt and Y. Frankel. Threshold cryptosystems. In G. Brassard, editor, *Proceedings of the 9th Annual International Cryptology Conference, Advances in Cryptology – CRYPTO 1989*, volume 435 of *LNCS*, pages 307–315. Springer, 1990.

[43] W. Diffie and M. Hellman. New directions in cryptography. *IEEE Transactions on information Theory*, 22(6):644–654, 1976.

[44] ECRYPT (European Network of Excellence for Cryptology). *ECRYPT II Yearly Report on Algorithms and Keysizes (2009-2010)*, 2010.

[45] T. ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. In G. Blakley and D. Chaum, editors, *Proceedings of International Cryptology Conference, Advances in Cryptology – CRYPTO 1984*, volume 196 of *LNCS*, pages 10–18. Springer, 1985.

[46] G. Frey, M. Muller, and H.G. Ruck. The Tate pairing and the discrete logarithm applied to elliptic curve cryptosystems. *IEEE Transactions on Information Theory*, 45(5):1717–1719, 1999.

[47] T. Fuhr and P. Paillier. Decryptable searchable encryption. In W. Susilo, J. K. Liu, and Y. Mu, editors, *Proceedings of the 1st International Conference on Provable Security – ProvSec 2007*, volume 4784 of *LNCS*, pages 228–236. Springer, 2007.

[48] S.D. Galbraith, K.G. Paterson, and N.P. Smart. Pairings for cryptographers. *Discrete Applied Mathematics*, 156(16):3113–3121, 2008.

[49] V. Goyal, A. Jain, O. Pandey, and A. Sahai. Bounded ciphertext policy attribute based encryption. In *Automata, Languages and Programming*, volume 5126 of *LNCS*, pages 579–591. Springer, 2010.

[50] 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 – CCS 2006*, pages 89–98. ACM Press New York, NY, USA, 2006.

[51] M. Green and G. Ateniese. Identity-based proxy re-encryption. In J. Katz and M. Yung, editors, *Proceedings of the 5th International Conference on Applied Cryptography and Network Security – ACNS 2007*, volume 4521 of *LNCS*, pages 288–306. Springer, 2007.

[52] P. Gutman. Pki: It's not dead, just resting. In *IEEE Computer*, volume 35 of *IEEE Computer*, pages 41–49, 2002.

[53] D. Hofheinz and E. Weinreb. Searchable encryption with decryption in the standard model. Technical report, Cryptology ePrint Archive, Report 2008/423, Available at http://eprint.iacr.org/2008/423, 2008.

[54] Y. Hwang and P. Lee. Public key encryption with conjunctive keyword search and its extension to a multi-user system. In T. Takagi, T. Okamoto, E. Okamoto, and T. Okamoto, editors, *Proceedings of the 1st International Conference on Pairing-based Cryptography – Pairing 2007*, volume 4575 of *LNCS*, pages 2–22. Springer, 2007.

[55] A. Ivan and Y. Dodis. Proxy cryptography revisited. In *Proceedings of the 10th Annual Network and Distributed System Security Symposium – NDSS 2003*. The Internet Society, 2003.

[56] M. Jakobsson. On quorum controlled asymmetric proxy re-encryption. In H. Imai and Y. Zheng, editors, *Proceedings of the 2nd International Workshop on Practice and Theory in Public Key Cryptography – PKC 1999*, volume 1560 of *LNCS*, pages 112–121. Springer, 1999.

[57] A. Joux. A one round protocol for tripartite Diffie–Hellman. *Journal of Cryptology*, 17(4):263–276, 2004.

[58] A. Joux and K. Nguyen. Separating decision Diffie–Hellman from computational Diffie–Hellman in cryptographic groups. *Journal of Cryptology*, 16(4):239–247, 2003.

[59] J. Katz and Y. Lindell. *Introduction to modern cryptography.* Chapman & Hall/CRC, 2008.

[60] J. Katz, A. Sahai, and B. Waters. Predicate encryption supporting disjunctions, polynomial equations, and inner products. In N. Smart, editor, *Porceedings of 27th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Advances in Cryptology – EUROCRYPT 2008*, volume 4965 of *LNCS*, pages 146–162. Springer, 2008.

[61] N. Koblitz. Elliptic curve cryptosystems. *Mathematics of computation*, 48(177):203–209, 1987.

[62] K. Kurosawa and Y. Desmedt. Optimum traitor tracing and asymmetric schemes. In K. Nyberg, editor, *Proceedings of International Conference on the Theory and Applications of Cryptographic Techniques, Advances in Cryptology – EUROCRYPT 1998*, volume 1403 of *LNCS*, pages 145–157. Springer, 1998.

[63] A. Lewko, T. Okamoto, A. Sahai, K. Takashima, and B. Waters. Fully secure functional encryption: Attribute-based encryption and (hierarchical) inner product encryption. In H. Gilbert, editor, *Proceedings of the 29th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Advances in Cryptology – EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 62–91. Springer, 2010.

[64] J. Li, K. Ren, B. Zhu, and Z. Wan. Privacy-aware attribute-based encryption with user accountability. In *Proceedings of the 12th International Conference on Information Security – ISC 2009*, volume 5735 of *LNCS*, pages 347–362. Springer, 2009.

[65] X. Liang, Z. Cao, H. Lin, and J. Shao. Attribute based proxy re-encryption with delegating capabilities. In *Proceedings of the 4th International Symposium on Information, Computer, and Communications Security – ASIACCS 2009*, pages 276–286. ACM, 2009.

[66] B. Libert and J.J. Quisquater. Efficient revocation and threshold pairing based cryptosystems. In *Proceedings of the 22nd annual symposium on Principles of distributed computing – PODC 2003*, pages 163–171. ACM, 2003.

[67] D. Lubicz and T. Sirvent. Attribute-based broadcast encryption scheme made efficient. In S. Vaudenay, editor, *Proceedings of the 1st International Conference on Cryptology in Africa – AFRICACRYPT 2008*, volume 5023 of *LNCS*. Springer, 2008.

[68] M. Mambo and E. Okamoto. Proxy Cryptosystems: Delegation of the Power to Decrypt Ciphertexts. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, 80(1):54–63, 1997.

[69] T. Matsuo. Proxy re-encryption systems for identity-based encryption. In T. Takagi, T. Okamoto, E. Okamoto, and T. Okamoto, editors, *Proceedings of the 1st International Conference on Pairing-based Cryptography – Pairing 2007*, volume 4575 of *LNCS*, pages 247–267. Springer, 2007.

[70] A.J. Menezes, P.C. Van Oorschot, and S.A. Vanstone. *Handbook of applied cryptography.* CRC, 1997.

[71] V. Miller. Use of elliptic curves in cryptography. In H. Williams, editor, *Proceedings of International Cryptology Conference, Advances in Cryptology – CRYPTO 1985*, volume 218 of *LNCS*, pages 417–426. Springer, 1986.

[72] V.S. Miller. The Weil pairing, and its efficient calculation. *Journal of Cryptology*, 17(4):235–261, 2004.

[73] J.F. Morar and D.M. Chess. Can cryptography prevent computer viruses? *VIRUS*, 127, 2000.

[74] D. Nali, C. Adams, and A. Miri. Using mediated identity-based cryptography to support role-based access control. In K. Zhang and Y. Zheng, editors, *Proceedings of the 7th International Conference, Information Security – ISC 2004*, volume 3225 of *LNCS*, pages 245–256. Springer, 2004.

[75] D. Nali, A. Miri, and C. Adams. Efficient revocation of dynamic security privileges in hierarchically structured communities. In *Proceedings of the 2nd Annual Conference on Privacy, Security and Trust – PST 2004*, pages 219–223. Citeseer, 2004.

[76] D. Naor, M. Naor, and J. Lotspiech. Revocation and tracing schemes for stateless receivers. In J. Kilian, editor, *Proceedings of the 21st Annual International Cryptology Conference, Advances in Cryptology – CRYPTO 2001*, volume 2139 of *LNCS*, pages 41–62. Springer, 2001.

[77] M. Naor and B. Pinkas. Efficient trace and revoke schemes. In Y. Frankel, editor, *Proceedings of the 4th International Conference on Financial Cryptography – FC 2000*, volume 1962 of *LNCS*, pages 1–20. Springer, 2000.

[78] M. Naor and M. Yung. Public-key cryptosystems provably secure against chosen ciphertext attacks. In *Proceedings of the 22nd STOC*, pages 427–437. ACM, 1995.

[79] T. Nishide, K. Yoneyama, and K. Ohta. Attribute-based encryption with partially hidden encryptor-specified access structures. In S. Bellovin, R. Gennaro, A. Keromytis, and M. Yung, editors, *Proceedings of the 6th International Conference on Applied Cryptography and Network Security – ACNS 2008*, volume 5037 of *LNCS*, pages 111–129. Springer, 2008.

[80] T. Okamoto. Cryptography based on bilinear maps. In M. Fossorier, H. Imai, S. Lin, and A. Poli, editors, *Proceedings of Applied Algebra, Algebraic Algorithms and Error-Correcting Codes – AAECC 2006*, volume 3857 of *LNCS*, pages 35–50. Springer, 2006.

[81] T. Okamoto and K. Takashima. Fully secure functional encryption with general relations from the decisional linear assumption. In H. Gilbert, editor, *Proceedings of the 29th Annual International Conference on the Theory and Applications*

of Cryptographic Techniques, Advances in Cryptology – EUROCRYPT 2010, volume 6110 of *LNCS*, pages 191–208. Springer, 2010.

[82] A. Ostrovksy, 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 – CCS 2007*, pages 195–203. ACM, 2007.

[83] M. Pirretti, P. Traynor, P. McDaniel, and B. Waters. Secure attribute-based systems. In *Proceedings of the 13th ACM conference on Computer and Communications Security – CCS 2006*, pages 99–112. ACM, 2006.

[84] F. PUB. Data Encryption Standard (DES). *National Bureau of Standards, US Department of Commerce*, 1977.

[85] R.L. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978.

[86] P. Rogaway and T. Shrimpton. Cryptographic hash-function basics: Definitions, implications, and separations for preimage resistance, second-preimage resistance, and collision resistance. In B. Roy and W. Meier, editors, *Proceedings of Fast Software Encryption – FSE 2004*, LNCS, pages 371–388. Springer, 2004.

[87] R. Sakai, K. Ohgishi, and M. Kasahara. Cryptosystems based on pairing over elliptic curves. In *Proceedings of the 2001 Symposium on Cryptography and Information Security*, 2001.

[88] A. Shamir. How to share a secret. *Communications of the ACM*, 22(11):612–613, 1979.

[89] A. Shamir. Identity-based cryptosystems and signature schemes. In G. Blakley and D. Chaum, editors, *Proceedings of International Cryptology Conference, Advances in Cryptology – CRYPTO 1984*, volume 196 of *LNCS*, pages 47–53. Springer, 1985.

[90] C.E. Shannon. Communication theory of secrecy systems. *Bell System Technical Journal*, 28(4):656–715, 1949.

[91] G. Shanqing, Z. Yingpei, W. Juan, and X. Qiuliang. Attribute-based re-encryption scheme in the standard model. *Journal of Natural Sciences*, 13(005):621–625, 2008.

[92] E. Shen, E. Shi, and B. Waters. Predicate privacy in encryption systems. In O. Reingold, editor, *Proceedings of the 6th Theory of Cryptography Conference – TCC 2009*, volume 5444 of *LNCS*, pages 457–473. Springer, 2009.

[93] V. Shoup. Lower bounds for discrete logarithms and related problems. In W. Fumy, editor, *Proceedings of International Conference on the Theory and Application of Cryptographic Techniques, Advances in Cryptology – EUROCRYPT 1997*, volume 1233 of *LNCS*, pages 256–266. Springer, 1997.

[94] V. Shoup. Sequences of games: a tool for taming complexity in security proofs. http://shoup.net/papers/, 2006.

[95] N.P. Smart. Access control using pairing based cryptography. In M. Joye, editor, *Proceedings of Topics in Cryptology, The Cryptographer's Track at the RSA Conference – CT-RSA 2003*, volume 2612 of *LNCS*, pages 111–121. Springer, 2003.

[96] D.X. Song, D. Wagner, and A. Perrig. Practical techniques for searches on encrypted data. In *Proceedings of the 2000 IEEE Symposium on Security and Privacy*, pages 44–55. IEEE Computer Society, 2000.

[97] P.C. Tang, J.S. Ash, D.W. Bates, J.M. Overhage, and D.Z. Sands. Personal health records: definitions, benefits, and strategies for overcoming barriers to adoption. *Journal of the American Medical Informatics Association*, 13(2):121, 2006.

[98] Q. Tang. Type-based proxy re-encryption and its construction. In D. Chowdhury, V. Rijmen, and A. Das, editors, *Proceedings of the 9th International Conference on Cryptology – INDOCRYPT 2008*, LNCS, pages 130–144. Springer, 2008.

[99] The US Department of Health and Human Services. Summary of the HIPAA Privacy Rule, 2003.

[100] G. Vanrenen and S. Smith. Distributing security-mediated PKI. In K. S. Katsikas, S. Gritzalis, and J. Lpez, editors, *First European PKI Workshop: Research and Applications – EuroPKI 2004*, volume 3093 of *LNCS*, pages 218–231. Springer, 2004.

[101] E. Verheul. Self-blindable credential certificates from the Weil pairing. In C Boyd, editor, *Proceedings of the 7th International Conference on the Theory and Application of Cryptology and Information Security, Advances in Cryptology – ASIACRYPT 2001*, volume 2248 of *LNCS*, pages 533–551. Springer, 2001.

[102] E.R. Verheul. Evidence that XTR is more secure than supersingular elliptic curve cryptosystems. In B. Pfitzmann, editor, *Proceedings of International Conference on the Theory and Application of Cryptographic Techniques, Advances in Cryptology – EUROCRYPT 2001*, volume 2045 of *LNCS*, pages 195–210. Springer, 2001.

[103] L. Wang, Z. Cao, T. Okamoto, Y. Miao, and E. Okamoto. Authorization-limited transformation-free proxy cryptosystems and their security analyses. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, (1):106–114, 2006.

[104] B. Waters and A. Sahai. Fuzzy identity based encryption. In R.J.F. Cramer, editor, *Proceedings of the 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Advances in Cryptology – EUROCRYPT 2005*, volume 3494 of *LNCS*, pages 457–473. Springer, 2005.

[105] B.R. Waters, D. Balfanz, G. Durfee, and D.K. Smetters. Building an encrypted and searchable audit log. In *Proceedings of ISOC Network and Distributed System Security Symposium – NDSS 2004*. Citeseer, 2004.

[106] E. Yoo, N. Jho, J. Cheon, and M. Kim. Efficient broadcast encryption using multiple interpolation methods. In Ch. Park and S. Chee, editors, *Proceedings of Information Security and Cryptology - ICISC 2004*, pages 87–103, 2004.

[107] R. Zhang and H Imai. Generic combination of public key encryption with keyword search and public key encryption. In F. Bao, S. Ling, T. Okamoto, H. Wang, and C. Xing, editors, *Proceedings of the 6th International Conference on Cryptology and Network Security – CANS 2007*, volume 4856 of *LNCS*, pages 159–174. Springer, 2007.

Abbreviations and Symbols

$\mathcal{A}$	The adversary
$\mathcal{B}$	The simulator
$\mathcal{CTOW} - ATK$	Ciphertext one-wayness attack
$\exists$	There exists
$\mathbb{F}$	Field
$\forall$	For all
$\leftarrow$	Uniformly chosen from
$\mathbb{G}$	Group
H	Hash Function
λ	Security Parameter
$\mathbb{C}$	Complex Number Set
$\mathbb{Q}$	Rational numbers set
$\mathbb{R}$	Real numbers set
$\mathcal{A}(x, y, ...,)$	Algorithm $\mathcal{A}$ has inputs $(x, y, ...,)$
$\mathcal{CI} - \mathcal{ATK}$	Ciphertext indistinguishability attack
mCDH	Modified Computational Diffie-Hellman
mpk	Master public key
Ω	Systems attribute set

ω	Users attribute set
ω'	The set of attributes that satisfy the access policy
$\Pr[X]$	Probability of event X
τ	Access policy
$\mathcal{TI} - ATK$	Trapdoor indistinguishability attack
$\mathbb{Z}$	Integer number set
e	Bilinear Map function
$s \in_R S$	s is selected uniformly at random from S
ABE	Attribute-based encryption
B-CP-ABE	Basic ciphertext-policy attribute-based encryption
CP-ABE	Ciphertext-policy attribute-based encryption
CP-ABPRE	Ciphertext-policy attribute-based proxy re-encryption
E-CP-ABE	Extended ciphertext-policy attribute-based encryption
IBE	Identity-Based Encryption
IND-ID-CCA	Indistinguishability under chosen-ciphertext attacks in IBE schemes
KP-ABE	Key-policy attribute-based encryption
mCP-ABE	Mediated ciphertext-policy attribute-based encryption
PKE	Public-key encryption
PRE	Proxy re-encryption
AC	Access control
BDH	Bilinear Diffie-Hellman
CDH	Computational Diffie-Hellman
DBDH	Decisional bilinear Diffie-Hellman
DDH	Decisional Diffie-Hellman
ECC	Elliptic curve cryptography
GGM	Generic group model

PBC-CO	Pairing-based cryptography with composite order group
PBC-PO	Pairing-based cryptography with prime order group
ROM	Random oracle model
SM	Standard model
SXDH	Symmetric External Diffie-Hellman
IND-CCA	Indistinguishable under chosen-ciphertext attacks
IND-CPA	Indistinguishable under chosen-plaintext attacks

SIKS Dissertatiereeks

====
1998
====

1998-1 Johan van den Akker (CWI)
DEGAS - An Active, Temporal Database of Autonomous Objects

1998-2 Floris Wiesman (UM)
Information Retrieval by Graphically Browsing Meta-Information

1998-3 Ans Steuten (TUD)
A Contribution to the Linguistic Analysis of Business Conversations
within the Language/Action Perspective

1998-4 Dennis Breuker (UM)
Memory versus Search in Games
1998-5 E.W.Oskamp (RUL)
Computerondersteuning bij Straftoemeting

====
1999
====

1999-1 Mark Sloof (VU)
Physiology of Quality Change Modelling;
Automated modelling of Quality Change of Agricultural Products

1999-2 Rob Potharst (EUR)
Classification using decision trees and neural nets

1999-3 Don Beal (UM)
The Nature of Minimax Search

1999-4 Jacques Penders (UM)
The practical Art of Moving Physical Objects

1999-5 Aldo de Moor (KUB)
Empowering Communities: A Method for the Legitimate User-Driven
Specification of Network Information Systems

1999-6 Niek J.E. Wijngaards (VU)
Re-design of compositional systems

1999-7 David Spelt (UT)
Verification support for object database design

1999-8 Jacques H.J. Lenting (UM)
Informed Gambling: Conception and Analysis of a Multi-Agent
Mechanism for Discrete Reallocation.

====
2000
====

2000-1 Frank Niessink (VU)
Perspectives on Improving Software Maintenance

2000-2 Koen Holtman (TUE)
Prototyping of CMS Storage Management

2000-3 Carolien M.T. Metselaar (UVA)
Sociaal-organisatorische gevolgen van kennistechnologie;
een procesbenadering en actorperspectief.

2000-4 Geert de Haan (VU)
ETAG, A Formal Model of Competence Knowledge for User Interface Design

2000-5 Ruud van der Pol (UM)
Knowledge-based Query Formulation in Information Retrieval.

2000-6 Rogier van Eijk (UU)
Programming Languages for Agent Communication

2000-7 Niels Peek (UU)
Decision-theoretic Planning of Clinical Patient Management

2000-8 Veerle Coup (EUR)
Sensitivity Analyis of Decision-Theoretic Networks

2000-9 Florian Waas (CWI)
Principles of Probabilistic Query Optimization

2000-10 Niels Nes (CWI)
Image Database Management System Design Considerations,
Algorithms and Architecture

2000-11 Jonas Karlsson (CWI)
Scalable Distributed Data Structures for Database Management

====
2001
====

2001-1 Silja Renooij (UU)
Qualitative Approaches to Quantifying Probabilistic Networks

2001-2 Koen Hindriks (UU)
Agent Programming Languages: Programming with Mental Models

2001-3 Maarten van Someren (UvA)
Learning as problem solving

2001-4 Evgueni Smirnov (UM)
Conjunctive and Disjunctive Version Spaces with
Instance-Based Boundary Sets

2001-5 Jacco van Ossenbruggen (VU)
Processing Structured Hypermedia: A Matter of Style

2001-6 Martijn van Welie (VU)
Task-based User Interface Design

2001-7 Bastiaan Schonhage (VU)
Diva: Architectural Perspectives on Information Visualization

2001-8 Pascal van Eck (VU)
A Compositional Semantic Structure for Multi-Agent Systems Dynamics.

2001-9 Pieter Jan 't Hoen (RUL)
Towards Distributed Development of Large Object-Oriented Models,
Views of Packages as Classes

2001-10 Maarten Sierhuis (UvA)
Modeling and Simulating Work Practice

BRAHMS: a multiagent modeling and simulation language
for work practice analysis and design

2001-11 Tom M. van Engers (VUA)
Knowledge Management:
The Role of Mental Models in Business Systems Design

====
2002
====

2002-01 Nico Lassing (VU)
Architecture-Level Modifiability Analysis

2002-02 Roelof van Zwol (UT)
Modelling and searching web-based document collections

2002-03 Henk Ernst Blok (UT)
Database Optimization Aspects for Information Retrieval

2002-04 Juan Roberto Castelo Valdueza (UU)
The Discrete Acyclic Digraph Markov Model in Data Mining

2002-05 Radu Serban (VU)
The Private Cyberspace Modeling Electronic Environments
inhabited by Privacy-concerned Agents

2002-06 Laurens Mommers (UL)
Applied legal epistemology;
Building a knowledge-based ontology of the legal domain

2002-07 Peter Boncz (CWI)
Monet: A Next-Generation DBMS Kernel For Query-Intensive Applications

2002-08 Jaap Gordijn (VU)
Value Based Requirements Engineering: Exploring Innovative
E-Commerce Ideas

2002-09 Willem-Jan van den Heuvel(KUB)
Integrating Modern Business Applications with Objectified Legacy Systems

2002-10 Brian Sheppard (UM)
Towards Perfect Play of Scrabble

2002-11 Wouter C.A. Wijngaards (VU)
Agent Based Modelling of Dynamics: Biological and Organisational Applications

2002-12 Albrecht Schmidt (Uva)
Processing XML in Database Systems

2002-13 Hongjing Wu (TUE)
A Reference Architecture for Adaptive Hypermedia Applications

2002-14 Wieke de Vries (UU)
Agent Interaction: Abstract Approaches to Modelling, Programming and
Verifying Multi-Agent Systems

2002-15 Rik Eshuis (UT)
Semantics and Verification of UML Activity Diagrams for Workflow Modelling

2002-16 Pieter van Langen (VU)
The Anatomy of Design: Foundations, Models and Applications

2002-17 Stefan Manegold (UVA)
Understanding, Modeling, and Improving Main-Memory Database Performance

====
2003
====

2003-01 Heiner Stuckenschmidt (VU)
Ontology-Based Information Sharing in Weakly Structured Environments

2003-02 Jan Broersen (VU)
Modal Action Logics for Reasoning About Reactive Systems

2003-03 Martijn Schuemie (TUD)
Human-Computer Interaction and Presence in Virtual Reality Exposure
Therapy

2003-04 Milan Petkovic (UT)
Content-Based Video Retrieval Supported by Database Technology

2003-05 Jos Lehmann (UVA)
Causation in Artificial Intelligence and Law - A modelling approach

2003-06 Boris van Schooten (UT)
Development and specification of virtual environments

2003-07 Machiel Jansen (UvA)
Formal Explorations of Knowledge Intensive Tasks

2003-08 Yongping Ran (UM)
Repair Based Scheduling

2003-09 Rens Kortmann (UM)
The resolution of visually guided behaviour

2003-10 Andreas Lincke (UvT)
Electronic Business Negotiation: Some experimental studies on the interaction
between medium, innovation context and culture

2003-11 Simon Keizer (UT)
Reasoning under Uncertainty in Natural Language Dialogue using Bayesian Networks

2003-12 Roeland Ordelman (UT)
Dutch speech recognition in multimedia information retrieval

2003-13 Jeroen Donkers (UM)
Nosce Hostem - Searching with Opponent Models

2003-14 Stijn Hoppenbrouwers (KUN)
Freezing Language: Conceptualisation Processes across ICT-Supported Organisations

2003-15 Mathijs de Weerdt (TUD)
Plan Merging in Multi-Agent Systems

2003-16 Menzo Windhouwer (CWI)
Feature Grammar Systems - Incremental Maintenance of Indexes to
Digital Media Warehouses

2003-17 David Jansen (UT)
Extensions of Statecharts with Probability, Time, and Stochastic Timing

2003-18 Levente Kocsis (UM)
Learning Search Decisions

====
2004

====

2004-01 Virginia Dignum (UU)
A Model for Organizational Interaction: Based on Agents, Founded in Logic

2004-02 Lai Xu (UvT)
Monitoring Multi-party Contracts for E-business

2004-03 Perry Groot (VU)
A Theoretical and Empirical Analysis of Approximation in Symbolic Problem Solving

2004-04 Chris van Aart (UVA)
Organizational Principles for Multi-Agent Architectures

2004-05 Viara Popova (EUR)
Knowledge discovery and monotonicity

2004-06 Bart-Jan Hommes (TUD)
The Evaluation of Business Process Modeling Techniques

2004-07 Elise Boltjes (UM)
Voorbeeldig onderwijs; voorbeeldgestuurd onderwijs, een opstap naar
abstract denken, vooral voor meisjes

2004-08 Joop Verbeek(UM)
Politie en de Nieuwe Internationale Informatiemarkt, Grensregionale
politiële gegevensuitwisseling en digitale expertise

2004-09 Martin Caminada (VU)
For the Sake of the Argument; explorations into argument-based reasoning

2004-10 Suzanne Kabel (UVA)
Knowledge-rich indexing of learning-objects

2004-11 Michel Klein (VU)
Change Management for Distributed Ontologies

2004-12 The Duy Bui (UT)
Creating emotions and facial expressions for embodied agents

2004-13 Wojciech Jamroga (UT)
Using Multiple Models of Reality: On Agents who Know how to Play

2004-14 Paul Harrenstein (UU)
Logic in Conflict. Logical Explorations in Strategic Equilibrium

2004-15 Arno Knobbe (UU)
Multi-Relational Data Mining

2004-16 Federico Divina (VU)
Hybrid Genetic Relational Search for Inductive Learning

2004-17 Mark Winands (UM)
Informed Search in Complex Games

2004-18 Vania Bessa Machado (UvA)
Supporting the Construction of Qualitative Knowledge Models

2004-19 Thijs Westerveld (UT)
Using generative probabilistic models for multimedia retrieval

2004-20 Madelon Evers (Nyenrode)
Learning from Design: facilitating multidisciplinary design teams

====

2005
====

2005-01 Floor Verdenius (UVA)
Methodological Aspects of Designing Induction-Based Applications

2005-02 Erik van der Werf (UM))
AI techniques for the game of Go

2005-03 Franc Grootjen (RUN)
A Pragmatic Approach to the Conceptualisation of Language

2005-04 Nirvana Meratnia (UT)
Towards Database Support for Moving Object data

2005-05 Gabriel Infante-Lopez (UVA)
Two-Level Probabilistic Grammars for Natural Language Parsing

2005-06 Pieter Spronck (UM)
Adaptive Game AI

2005-07 Flavius Frasincar (TUE)
Hypermedia Presentation Generation for Semantic Web Information Systems

2005-08 Richard Vdovjak (TUE)
A Model-driven Approach for Building Distributed Ontology-based Web Applications

2005-09 Jeen Broekstra (VU)
Storage, Querying and Inferencing for Semantic Web Languages

2005-10 Anders Bouwer (UVA)
Explaining Behaviour: Using Qualitative Simulation in Interactive Learning Environments

2005-11 Elth Ogston (VU)
Agent Based Matchmaking and Clustering - A Decentralized Approach to Search

2005-12 Csaba Boer (EUR)
Distributed Simulation in Industry

2005-13 Fred Hamburg (UL)
Een Computermodel voor het Ondersteunen van Euthanasiebeslissingen

2005-14 Borys Omelayenko (VU)
Web-Service configuration on the Semantic Web; Exploring how semantics meets pragmatics

2005-15 Tibor Bosse (VU)
Analysis of the Dynamics of Cognitive Processes

2005-16 Joris Graaumans (UU)
Usability of XML Query Languages

2005-17 Boris Shishkov (TUD)
Software Specification Based on Re-usable Business Components

2005-18 Danielle Sent (UU)
Test-selection strategies for probabilistic networks

2005-19 Michel van Dartel (UM)
Situated Representation

2005-20 Cristina Coteanu (UL)
Cyber Consumer Law, State of the Art and Perspectives

2005-21 Wijnand Derks (UT)
Improving Concurrency and Recovery in Database Systems by Exploiting Application Semantics

====
2006
====

2006-01 Samuil Angelov (TUE)
Foundations of B2B Electronic Contracting

2006-02 Cristina Chisalita (VU)
Contextual issues in the design and use of information technology in organizations

2006-03 Noor Christoph (UVA)
The role of metacognitive skills in learning to solve problems

2006-04 Marta Sabou (VU)
Building Web Service Ontologies

2006-05 Cees Pierik (UU)
Validation Techniques for Object-Oriented Proof Outlines

2006-06 Ziv Baida (VU)
Software-aided Service Bundling - Intelligent Methods & Tools for Graphical Service Modeling

2006-07 Marko Smiljanic (UT)
XML schema matching – balancing efficiency and effectiveness by means of clustering

2006-08 Eelco Herder (UT)
Forward, Back and Home Again - Analyzing User Behavior on the Web

2006-09 Mohamed Wahdan (UM)
Automatic Formulation of the Auditor's Opinion

2006-10 Ronny Siebes (VU)
Semantic Routing in Peer-to-Peer Systems

2006-11 Joeri van Ruth (UT)
Flattening Queries over Nested Data Types

2006-12 Bert Bongers (VU)
Interactivation - Towards an e-cology of people, our technological environment, and the arts

2006-13 Henk-Jan Lebbink (UU)
Dialogue and Decision Games for Information Exchanging Agents

2006-14 Johan Hoorn (VU)
Software Requirements: Update, Upgrade, Redesign - towards a Theory of Requirements Change

2006-15 Rainer Malik (UU)
CONAN: Text Mining in the Biomedical Domain

2006-16 Carsten Riggelsen (UU)
Approximation Methods for Efficient Learning of Bayesian Networks

2006-17 Stacey Nagata (UU)
User Assistance for Multitasking with Interruptions on a Mobile Device

2006-18 Valentin Zhizhkun (UVA)

Graph transformation for Natural Language Processing

2006-19 Birna van Riemsdijk (UU)
Cognitive Agent Programming: A Semantic Approach

2006-20 Marina Velikova (UvT)
Monotone models for prediction in data mining

2006-21 Bas van Gils (RUN)
Aptness on the Web

2006-22 Paul de Vrieze (RUN)
Fundaments of Adaptive Personalisation

2006-23 Ion Juvina (UU)
Development of Cognitive Model for Navigating on the Web

2006-24 Laura Hollink (VU)
Semantic Annotation for Retrieval of Visual Resources

2006-25 Madalina Drugan (UU)
Conditional log-likelihood MDL and Evolutionary MCMC

2006-26 Vojkan Mihajlovic (UT)
Score Region Algebra: A Flexible Framework for Structured Information Retrieval

2006-27 Stefano Bocconi (CWI)
Vox Populi: generating video documentaries from semantically annotated media repositories

2006-28 Borkur Sigurbjornsson (UVA)
Focused Information Access using XML Element Retrieval

====
2007
====

2007-01 Kees Leune (UvT)
Access Control and Service-Oriented Architectures

2007-02 Wouter Teepe (RUG)
Reconciling Information Exchange and Confidentiality: A Formal Approach

2007-03 Peter Mika (VU)
Social Networks and the Semantic Web

2007-04 Jurriaan van Diggelen (UU)
Achieving Semantic Interoperability in Multi-agent Systems: a dialogue-based approach

2007-05 Bart Schermer (UL)
Software Agents, Surveillance, and the Right to Privacy: a Legislative Framework for Agent-enabled Surveillance

2007-06 Gilad Mishne (UVA)
Applied Text Analytics for Blogs

2007-07 Natasa Jovanovic' (UT)
To Whom It May Concern - Addressee Identification in Face-to-Face Meetings

2007-08 Mark Hoogendoorn (VU)
Modeling of Change in Multi-Agent Organizations

2007-09 David Mobach (VU)
Agent-Based Mediated Service Negotiation

2007-10 Huib Aldewereld (UU)
Autonomy vs. Conformity: an Institutional Perspective on Norms and Protocols

2007-11 Natalia Stash (TUE)
Incorporating Cognitive/Learning Styles in a General-Purpose Adaptive Hypermedia System

2007-12 Marcel van Gerven (RUN)
Bayesian Networks for Clinical Decision Support: A Rational Approach to Dynamic Decision-Making under Uncertainty

2007-13 Rutger Rienks (UT)
Meetings in Smart Environments; Implications of Progressing Technology

2007-14 Niek Bergboer (UM)
Context-Based Image Analysis

2007-15 Joyca Lacroix (UM)
NIM: a Situated Computational Memory Model

2007-16 Davide Grossi (UU)
Designing Invisible Handcuffs. Formal investigations in Institutions and Organizations for Multi-agent Systems

2007-17 Theodore Charitos (UU)
Reasoning with Dynamic Networks in Practice

2007-18 Bart Orriens (UvT)
On the development an management of adaptive business collaborations

2007-19 David Levy (UM)
Intimate relationships with artificial partners

2007-20 Slinger Jansen (UU)
Customer Configuration Updating in a Software Supply Network

2007-21 Karianne Vermaas (UU)
Fast diffusion and broadening use: A research on residential adoption and usage of broadband internet in the Netherlands between 2001 and 2005

2007-22 Zlatko Zlatev (UT)
Goal-oriented design of value and process models from patterns

2007-23 Peter Barna (TUE)
Specification of Application Logic in Web Information Systems

2007-24 Georgina Ramrez Camps (CWI)
Structural Features in XML Retrieval

2007-25 Joost Schalken (VU)
Empirical Investigations in Software Process Improvement

====
2008
====

2008-01 Katalin Boer-Sorbn (EUR)
Agent-Based Simulation of Financial Markets: A modular,continuous-time approach

2008-02 Alexei Sharpanskykh (VU)
On Computer-Aided Methods for Modeling and Analysis of Organizations

2008-03 Vera Hollink (UVA)
Optimizing hierarchical menus: a usage-based approach

2008-04 Ander de Keijzer (UT)

Management of Uncertain Data - towards unattended integration

2008-05 Bela Mutschler (UT)
Modeling and simulating causal dependencies on process-aware information systems from a cost perspective

2008-06 Arjen Hommersom (RUN)
On the Application of Formal Methods to Clinical Guidelines, an Artificial Intelligence Perspective

2008-07 Peter van Rosmalen (OU)
Supporting the tutor in the design and support of adaptive e-learning

2008-08 Janneke Bolt (UU)
Bayesian Networks: Aspects of Approximate Inference

2008-09 Christof van Nimwegen (UU)
The paradox of the guided user: assistance can be counter-effective

2008-10 Wauter Bosma (UT)
Discourse oriented summarization

2008-11 Vera Kartseva (VU)
Designing Controls for Network Organizations: A Value-Based Approach

2008-12 Jozsef Farkas (RUN)
A Semiotically Oriented Cognitive Model of Knowledge Representation

2008-13 Caterina Carraciolo (UVA)
Topic Driven Access to Scientific Handbooks

2008-14 Arthur van Bunningen (UT)
Context-Aware Querying; Better Answers with Less Effort

2008-15 Martijn van Otterlo (UT)
The Logic of Adaptive Behavior: Knowledge Representation and Algorithms for the Markov Decision Process Framework in First-Order Domains.

2008-16 Henriette van Vugt (VU)
Embodied agents from a user's perspective

2008-17 Martin Op 't Land (TUD)
Applying Architecture and Ontology to the Splitting and Allying of Enterprises

2008-18 Guido de Croon (UM)
Adaptive Active Vision

2008-19 Henning Rode (UT)
From Document to Entity Retrieval: Improving Precision and Performance of Focused Text Search

2008-20 Rex Arendsen (UVA)
Geen bericht, goed bericht. Een onderzoek naar de effecten van de introductie van elektronisch berichtenverkeer met de overheid op de administratieve lasten van bedrijven

2008-21 Krisztian Balog (UVA)
People Search in the Enterprise

2008-22 Henk Koning (UU)
Communication of IT-Architecture

2008-23 Stefan Visscher (UU)
Bayesian network models for the management of ventilator-associated pneumonia

2008-24 Zharko Aleksovski (VU)
Using background knowledge in ontology matching

2008-25 Geert Jonker (UU)
Efficient and Equitable Exchange in Air Traffic Management Plan Repair using Spender-signed Currency

2008-26 Marijn Huijbregts (UT)
Segmentation, Diarization and Speech Transcription: Surprise Data Unraveled

2008-27 Hubert Vogten (OU)
Design and Implementation Strategies for IMS Learning Design

2008-28 Ildiko Flesch (RUN)
On the Use of Independence Relations in Bayesian Networks

2008-29 Dennis Reidsma (UT)
Annotations and Subjective Machines - Of Annotators, Embodied Agents, Users, and Other Humans

2008-30 Wouter van Atteveldt (VU)
Semantic Network Analysis: Techniques for Extracting, Representing and Querying Media Content

2008-31 Loes Braun (UM)
Pro-Active Medical Information Retrieval

2008-32 Trung H. Bui (UT)
Toward Affective Dialogue Management using Partially Observable Markov Decision Processes

2008-33 Frank Terpstra (UVA)
Scientific Workflow Design; theoretical and practical issues

2008-34 Jeroen de Knijf (UU)
Studies in Frequent Tree Mining

2008-35 Ben Torben Nielsen (UvT)
Dendritic morphologies: function shapes structure

====
2009
====

2009-01 Rasa Jurgelenaite (RUN)
Symmetric Causal Independence Models

2009-02 Willem Robert van Hage (VU)
Evaluating Ontology-Alignment Techniques

2009-03 Hans Stol (UvT)
A Framework for Evidence-based Policy Making Using IT

2009-04 Josephine Nabukenya (RUN)
Improving the Quality of Organisational Policy Making using Collaboration Engineering

2009-05 Sietse Overbeek (RUN)
Bridging Supply and Demand for Knowledge Intensive Tasks - Based on Knowledge, Cognition, and Quality

2009-06 Muhammad Subianto (UU)
Understanding Classification

2009-07 Ronald Poppe (UT)
Discriminative Vision-Based Recovery and Recognition of Human Motion

2009-08 Volker Nannen (VU)
Evolutionary Agent-Based Policy Analysis in Dynamic Environments

2009-09 Benjamin Kanagwa (RUN)
Design, Discovery and Construction of Service-oriented Systems

2009-10 Jan Wielemaker (UVA)
Logic programming for knowledge-intensive interactive applications

2009-11 Alexander Boer (UVA)
Legal Theory, Sources of Law & the Semantic Web

2009-12 Peter Massuthe (TUE, Humboldt-Universitaet zu Berlin)
perating Guidelines for Services

2009-13 Steven de Jong (UM)
Fairness in Multi-Agent Systems

2009-14 Maksym Korotkiy (VU)
From ontology-enabled services to service-enabled ontologies (making ontologies work in e-science with ONTO-SOA)

2009-15 Rinke Hoekstra (UVA)
Ontology Representation - Design Patterns and Ontologies that Make Sense

2009-16 Fritz Reul (UvT)
New Architectures in Computer Chess

2009-17 Laurens van der Maaten (UvT)
Feature Extraction from Visual Data

2009-18 Fabian Groffen (CWI)
Armada, An Evolving Database System

2009-19 Valentin Robu (CWI)
Modeling Preferences, Strategic Reasoning and Collaboration in Agent-Mediated Electronic Markets

2009-20 Bob van der Vecht (UU)
Adjustable Autonomy: Controling Influences on Decision Making

2009-21 Stijn Vanderlooy (UM)
Ranking and Reliable Classification

2009-22 Pavel Serdyukov (UT)
Search For Expertise: Going beyond direct evidence

2009-23 Peter Hofgesang (VU)
Modelling Web Usage in a Changing Environment

2009-24 Annerieke Heuvelink (VUA)
Cognitive Models for Training Simulations

2009-25 Alex van Ballegooij (CWI)
"RAM: Array Database Management through Relational Mapping"

2009-26 Fernando Koch (UU)
An Agent-Based Model for the Development of Intelligent Mobile Services

2009-27 Christian Glahn (OU)
Contextual Support of social Engagement and Reflection on the Web

2009-28 Sander Evers (UT)
Sensor Data Management with Probabilistic Models

2009-29 Stanislav Pokraev (UT)
Model-Driven Semantic Integration of Service-Oriented Applications

2009-30 Marcin Zukowski (CWI)
Balancing vectorized query execution with bandwidth-optimized storage

2009-31 Sofiya Katrenko (UVA)
A Closer Look at Learning Relations from Text

2009-32 Rik Farenhorst (VU) and Remco de Boer (VU)
Architectural Knowledge Management: Supporting Architects and Auditors

2009-33 Khiet Truong (UT)
How Does Real Affect Affect Affect Recognition In Speech?

2009-34 Inge van de Weerd (UU)
Advancing in Software Product Management: An Incremental Method
Engineering Approach

2009-35 Wouter Koelewijn (UL)
Privacy en Politiegegevens; Over geautomatiseerde normatieve informatie-uitwisseling

2009-36 Marco Kalz (OUN)
Placement Support for Learners in Learning Networks

2009-37 Hendrik Drachsler (OUN)
Navigation Support for Learners in Informal Learning Networks

2009-38 Riina Vuorikari (OU)
Tags and self-organisation: a metadata ecology for learning resources in a
multilingual context

2009-39 Christian Stahl (TUE, Humboldt-Universitaet zu Berlin)
Service Substitution – A Behavioral Approach Based on Petri Nets

2009-40 Stephan Raaijmakers (UvT)
Multinomial Language Learning: Investigations into the Geometry of Language

2009-41 Igor Berezhnyy (UvT)
Digital Analysis of Paintings

2009-42 Toine Bogers
Recommender Systems for Social Bookmarking

2009-43 Virginia Nunes Leal Franqueira (UT)
Finding Multi-step Attacks in Computer Networks using Heuristic Search
and Mobile Ambients

2009-44 Roberto Santana Tapia (UT)
Assessing Business-IT Alignment in Networked Organizations

2009-45 Jilles Vreeken (UU)
Making Pattern Mining Useful

2009-46 Loredana Afanasiev (UvA)
Querying XML: Benchmarks and Recursion

====

2010

====

2010-01 Matthijs van Leeuwen (UU)
Patterns that Matter

2010-02 Ingo Wassink (UT)
Work flows in Life Science

2010-03 Joost Geurts (CWI)
A Document Engineering Model and Processing Framework for Multimedia documents

2010-04 Olga Kulyk (UT)
Do You Know What I Know? Situational Awareness of Co-located Teams
in Multidisplay Environments

2010-05 Claudia Hauff (UT)
Predicting the Effectiveness of Queries and Retrieval Systems

2010-06 Sander Bakkes (UvT)
Rapid Adaptation of Video Game AI

2010-07 Wim Fikkert (UT)
Gesture interaction at a Distance

2010-08 Krzysztof Siewicz (UL)
Towards an Improved Regulatory Framework of Free Software. Protecting user
freedoms in a world of software communities and eGovernments

2010-09 Hugo Kielman (UL)
A Politiele gegevensverwerking en Privacy, Naar een effectieve waarborging

2010-10 Rebecca Ong (UL)
Mobile Communication and Protection of Children 2010-11 Adriaan Ter Mors (TUD)
The world according to MARP: Multi-Agent Route Planning

2010-12 Susan van den Braak (UU)
Sensemaking software for crime analysis

2010-13 Gianluigi Folino (RUN)
High Performance Data Mining using Bio-inspired techniques

2010-14 Sander van Splunter (VU)
Automated Web Service Reconfiguration

2010-15 Lianne Bodenstaff (UT)
Managing Dependency Relations in Inter-Organizational Models

2010-16 Sicco Verwer (TUD)
Efficient Identification of Timed Automata, theory and practice

2010-17 Spyros Kotoulas (VU)
Scalable Discovery of Networked Resources: Algorithms, Infrastructure, Applications

2010-18 Charlotte Gerritsen (VU)
Caught in the Act: Investigating Crime by Agent-Based Simulation

2010-19 Henriette Cramer (UvA)
People's Responses to Autonomous and Adaptive Systems

2010-20 Ivo Swartjes (UT)
Whose Story Is It Anyway? How Improv Informs Agency and Authorship of Emergent Narrative

2010-21 Harold van Heerde (UT)
Privacy-aware data management by means of data degradation

2010-22 Michiel Hildebrand (CWI)
End-user Support for Access to
Heterogeneous Linked Data

2010-23 Bas Steunebrink (UU)
The Logical Structure of Emotions
2010-24 Dmytro Tykhonov
Designing Generic and Efficient Negotiation Strategies

2010-25 Zulfiqar Ali Memon (VU)
Modelling Human-Awareness for Ambient Agents: A Human Mindreading Perspective

2010-26 Ying Zhang (CWI)
XRPC: Efficient Distributed Query Processing on Heterogeneous XQuery Engines

2010-27 Marten Voulon (UL)
Automatisch contracteren

2010-28 Arne Koopman (UU)
Characteristic Relational Patterns

2010-29 Stratos Idreos(CWI)
Database Cracking: Towards Auto-tuning Database Kernels

2010-30 Marieke van Erp (UvT)
Accessing Natural History - Discoveries in data cleaning, structuring, and retrieval

2010-31 Victor de Boer (UVA)
Ontology Enrichment from Heterogeneous Sources on the Web

2010-32 Marcel Hiel (UvT)
An Adaptive Service Oriented Architecture: Automatically solving Interoperability Problems

2010-33 Robin Aly (UT)
Modeling Representation Uncertainty in Concept-Based Multimedia Retrieval

2010-34 Teduh Dirgahayu (UT)
Interaction Design in Service Compositions

2010-35 Dolf Trieschnigg (UT)
Proof of Concept: Concept-based Biomedical Information Retrieval

2010-36 Jose Janssen (OU)
Paving the Way for Lifelong Learning; Facilitating competence development
through a learning path specification

2010-37 Niels Lohmann (TUE)
Correctness of services and their composition

2010-38 Dirk Fahland (TUE)
From Scenarios to components

2010-39 Ghazanfar Farooq Siddiqui (VU)
Integrative modeling of emotions in virtual agents

2010-40 Mark van Assem (VU)
Converting and Integrating Vocabularies for the Semantic Web

2010-41 Guillaume Chaslot (UM)
Monte-Carlo Tree Search

2010-42 Sybren de Kinderen (VU)
Needs-driven service bundling in a multi-supplier setting - the computational
e3-service approach

2010-43 Peter van Kranenburg (UU)
A Computational Approach to Content-Based Retrieval of Folk Song Melodies

2010-44 Pieter Bellekens (TUE)
An Approach towards Context-sensitive and User-adapted Access to Heterogeneous
Data Sources, Illustrated in the Television Domain

2010-45 Vasilios Andrikopoulos (UvT)
A theory and model for the evolution of software services

2010-46 Vincent Pijpers (VU)
e3alignment: Exploring Inter-Organizational Business-ICT Alignment

2010-47 Chen Li (UT)
Mining Process Model Variants: Challenges, Techniques, Examples

2010-48 Milan Lovric (EUR)
Behavioral Finance and Agent-Based Artificial Markets

2010-49 Jahn-Takeshi Saito (UM)
Solving difficult game positions

2010-50 Bouke Huurnink (UVA)
Search in Audiovisual Broadcast Archives

2010-51 Alia Khairia Amin (CWI)
Understanding and supporting information seeking tasks in multiple sources

2010-52 Peter-Paul van Maanen (VU)
Adaptive Support for Human-Computer Teams: Exploring the Use of Cognitive Models of Trust and Attention

2010-53 Edgar Meij (UVA)
Combining Concepts and Language Models for Information Access

====
2010
====

2011-01 Botond Cseke (RUN)
Variational Algorithms for Bayesian Inference in Latent Gaussian Models

2011-02 Nick Tinnemeier(UU)
Work flows in Life Science

2011-03
Jan Martijn van der Werf (TUE)
Compositional Design and Verification of Component-Based Information Systems

2011-04 Hado van Hasselt (UU)
Insights in Reinforcement Learning; Formal analysis and empirical evaluation of temporal-difference learning algorithms

2011-05 Base van der Raadt (VU)
Enterprise Architecture Coming of Age - Increasing the Performance of an Emerging Discipline.

2011-06 Yiwen Wang (TUE)
Semantically-Enhanced Recommendations in Cultural Heritage

2011-07 Yujia Cao (UT)
Multimodal Information Presentation for High Load Human Computer Interaction

2011-08 Nieske Vergunst (UU)
BDI-based Generation of Robust Task-Oriented Dialogues 2011-09 Tim de Jong (OU)
Contextualised Mobile Media for Learning

2011-10 Bart Bogaert (UvT)
Cloud Content Contention

2011-11 Dhaval Vyas (UT)
Designing for Awareness: An Experience-focused HCI Perspective

2011-12 Carmen Bratosin (TUE)
Grid Architecture for Distributed Process Mining

2011-13 Xiaoyu Mao (UvT)
Airport under Control. Multiagent Scheduling for Airport Ground Handling

2011-14 Milan Lovric (EUR)
Behavioral Finance and Agent-Based Artificial Markets

2011-15 Marijn Koolen (UvA)
The Meaning of Structure: the Value of Link Evidence for Information Retrieval

2011-16 Maarten Schadd (UM)
Selective Search in Games of Different Complexity

2011-17 Jiyin He (UVA)
Exploring Topic Structure: Coherence, Diversity and Relatedness

2011-18 Mark Ponsen (UM)
Strategic Decision-Making in complex games

2011-19 Ellen Rusman (OU)
The Mind ' s Eye on Personal Profiles

2011-20 Qing Gu (VU)
Guiding service-oriented software engineering - A view-based approach

2011-21 Linda Terlouw (TUD)
Modularization and Specification of Service-Oriented Systems

2011-22 Junte Zhang (UVA)
System Evaluation of Archival Description and Access

2011-23 Wouter Weerkamp (UVA)
Finding People and their Utterances in Social Media

2011-24 Herwin van Welbergen (UT)
Behavior Generation for Interpersonal Coordination with Virtual Humans On Specifying, Scheduling and Realizing Multimodal Virtual Human Behavior

2011-25 Syed Waqar ul Qounain Jaffry (VU)
Analysis and Validation of Models for Trust Dynamics

2011-26 Matthijs Aart Pontier (VU)
Virtual Agents for Human Communication - Emotion Regulation and Involvement-Distance Trade-Offs in Embodied Conversational Agents and Robots

2011-27 Aniel Bhulai (VU)
Dynamic website optimization through autonomous management of design patterns

2011-28 Rianne Kaptein(UVA)
Effective Focused Retrieval by Exploiting Query Context and Document Structure

2011-29 Faisal Kamiran (TUE)
Discrimination-aware Classification

2011-30 Egon van den Broek (UT)
Affective Signal Processing (ASP): Unraveling the mystery of emotions