# Attribute Based Cryptographic Enforcements for Security and Privacy in E-health Environments

**Harsha Sandaruwan Gardiyawasam Pussewalage**

# Attribute Based Cryptographic Enforcements for Security and Privacy in E-health Environments

Doctoral Dissertation for the Degree *Philosophiae Doctor (PhD)* at
the Faculty of Engineering and Science, Specialization in
Information and Communication Technology

University of Agder
Faculty of Engineering and Science
2019

Doctoral Dissertations at the University of Agder 224
ISBN: 978-82-7117-923-6
ISSN: 1504-9272



Printed by 07 Media
Oslo

*Dedicated to my parents*

***G. P. Sumanadasa and Nalini Samarakoon***

*my wife and son*

***Sasanka Ranasinghe and Sandira Sahasvin***

*and my sister*

***Anusha Lakmini***

# Abstract

In the last few decades, there have been significant efforts in integrating information and communication technologies into healthcare practices. This new paradigm commonly known as electronic healthcare (e-health) allows provisioning of healthcare services at an affordable price to its consumers while enabling a platform for efficient inter-domain health information exchange. Although such benefits exist, given that health information of patients contain a lot of sensitive information, secure sharing of patient records is of utmost importance to ensure the privacy of the patients. In addition, the linkability of different user access sessions over patient health information could also lead to the violation of patient privacy as well as the privacy of the accessing user. Furthermore, to strengthen the access flexibility in collaborative e-health environments, access delegation plays a vital role. However, access delegation has to be enforced in a controlled manner, and it is a research area that has not received significant attention.

In this dissertation, we considered two application scenarios that resemble a collaborative e-health environment. In the first scenario, the health information of patients are stored under the control of a local healthcare provider (LHP), and we require the health information to be shared with the healthcare professionals of LHP as well as users from other domains in a flexible and a privacy preserving manner. In the second scenario, we considered the case where health information of patients are stored in a third-party cloud platform which brings the challenge of enforcing flexible and privacy preserving access over the encrypted data. In relation to the above stated scenarios, our objective is to propose efficient attribute based cryptographic constructions that enable access anonymization and controlled access delegatability. To achieve this objective, in this dissertation, we propose seven attribute based cryptographic constructions which not only enable the aforementioned characteristics but also ensure secure, privacy preserving and flexible access to the stored health information of patients.

# Preface

This dissertation is the result of the research work carried out at the Department of Information and Communication Technology (ICT) of Faculty of Engineering and Science, University of Agder (UiA), in Grimstad, Norway, from August 2015 to November 2018. Professor Vladimir A. Oleshchuk has been the main supervisor of the research work, and Professor Geir M. Køien has been the co-supervisor. The funding for the research work presented in this dissertation has been granted by UiA.

Production note: LaTeX has been adopted as the tool for writing this dissertation, and the simulation results presented in this dissertation are obtained through JAVA programs with the help of Java Pairing Based Cryptography (JPBC) cryptographic library.

# Acknowledgments

After an enthralling journey of three years, my PhD carrier has finally come to a conclusion. Hereby, I would like to take this as an opportunity to show my gratitude to all who supported me throughout my PhD carrier.

First and foremost, I am deeply indebted to my supervisor, Professor Vladimir A. Oleshchuk whose constant support, motivation and guidance have been instrumental factors to my success towards the PhD. It was an honor for me to continue my PhD studies under his supervision after receiving guidance from him for my master's thesis during the two-year master's degree at UiA. I am also very much thankful for him to give me enough space and time to discover where my true research interests lie and listen to me patiently and giving me valuable suggestions to sharpen my ideas. Not only that, I have benefited tremendously from his mentoring, encouraging thoughts and enthusiasm towards research which undoubtedly helped me immensely to grow as a researcher. It has been a great pleasure for me to go on this journey with him and without his broad knowledge and constructive suggestions I would not have come this far as a researcher. I would also like to express my sincere gratitude to my co-supervisor, Professor Geir M. Køien who has his own way of inspiring PhD students with asking a lot of questions and it definitely had a positive impact on my critical thinking capabilities.

A special thank should also go to Professor Frank Reichert, the Rector of UiA and Dr. Chandana Perera, Dr. Keerthi Gunawickrama, Dr. Nisabha Jayasundere, Dr. Kandasamy Pirapaharan of the Faculty of Engineering, University of Ruhuna, Sri Lanka for making it a possibility for me to come to Norway through the quota scheme as a master's student in the first place. I also wish to thank the Head of the Department of ICT at UiA, Professor Folke Haugland for providing me with all the necessary administrative support. I am also very much grateful to PhD program coordinators at UiA, Mrs. Emma Elisabeth Horneman, Mrs. Tonje Sti and Mrs. Kristine Evensen Reinfjord for all the help and kindness I received, both professionally and personally as well as making the working environment a quite pleasant one.

I would also like to thank Dr. Indika Anuradha Mendis for his encouragements and guidance during different phases of this eventful journey, in addition to being more than a friend for both me and my family. My gratitude also goes to my colleagues and friends I met in Grimstad including Waseem Al-Hawani, Debasish Ghose, Darshana Abeyrathna, Madhava Jayathilake, Kalpanie Mendis, Ranga Wijesekara, Samangi Perera, Thilina Weerasinghe and Jagath Senanayaka for providing a sense of community and having a nice time together. Many thanks should also go to Milindanath Samarasinghe and Viveka Edirisinghe for being true friends and welcoming us always, although you both run a busy life.

It is with immense love and affection that I show my sincere gratitude for my parents, Gardiyawasam Pussewalage Sumanadasa and Nalini Samarakoon and my sister, Anusha Lakmini for staying behind and supporting me through the whole time. Last but not least, I am heartily thankful to my wife, Sasanka Ranasinghe for your encouragements, understanding and your love made this journey an enjoyable one.

G. P. Harsha Sandaruwan
November 2018
Grimstad, Norway

# List of Publications

All the papers listed below are an outcome of the research work carried out by the author of this dissertation, including two submitted and seven published papers. In the first part of the list, we have mentioned the papers that contribute towards the content of different chapters of this dissertation and they are sorted according to the topics of this dissertation. Paper 1 contributes to Chapter 1, while Papers 2 - 3, 4 - 6, 7, 8 contribute to Chapters 3, 4, 5, and Chapter 6 respectively. Paper 9 is within the framework of this doctoral thesis, but it does not contribute towards this thesis.

## Papers Contributing to the Dissertation

**Paper 1:** H. S. G. Pussewalage and V. A. Oleshchuk, "Privacy Preserving Mechanisms for Enforcing Security and Privacy Requirements in E-health Solutions", in *International Journal of Information Management*, vol. 36, no. 6, pp. 1161-1173, 2016.

**Paper 2:** H. S. G. Pussewalage and V. A. Oleshchuk, "An Attribute Based Access Control Scheme for Secure Sharing of Electronic Health Records", in *Proceedings of the 18th IEEE International Conference on E-health, Networking, Application and Services (IEEE HEALTHCOM'16)*. IEEE, Sep. 2016, pp. 526-531.

**Paper 3:** H. S. G. Pussewalage and V. A. Oleshchuk, "An Efficient Multi-Show Unlinkable Attribute Based Credential Scheme for a Collaborative E-health Environment", in *Proceedings of the 3rd IEEE International Conference on Collaboration and Internet Computing (IEEE CIC'17)*. IEEE, Oct. 2017, pp. 421-428.

**Paper 4:** H. S. G. Pussewalage and V. A. Oleshchuk, "An Anonymous Delegatable Attribute Based Credential Scheme for a Collaborative E-health Environment", Submitted to SS: *Advances in Internet-Based Collaborative Technologies of ACM Transactions on Internet Technology (ACM TOIT)*, Jan. 2018 (first round); Aug. 2018 (second round); Dec. 2018 (third round).

**Paper 5:** H. S. G. Pussewalage and V. A. Oleshchuk, "Attribute Based Access Control Scheme With Controlled Access Delegation for Collaborative E-health Environments", in *Journal of Information Security and Applications*, vol. 37, pp. 50-64, 2017.

**Paper 6:** H. S. G. Pussewalage and V. A. Oleshchuk, "Blockchain Based Delegatable Access Control Scheme for a Collaborative E-health Environment", in *Proceedings of the 1st IEEE International Conference on Blockchain (IEEE Blockchain'18)*. IEEE, Jul. 2018, pp. 1204-1211.

**Paper 7:** H. S. G. Pussewalage and V. A. Oleshchuk, "A Distributed Multi-Authority Attribute Based Encryption Scheme for Secure Sharing of Personal Health Records", in *Proceedings of the 22nd ACM Symposium on Access Control Models and Technologies (ACM SACMAT'17)*. ACM, Jun. 2017, pp. 255-262.

**Paper 8:** H. S. G. Pussewalage and V. A. Oleshchuk, "A Delegatable Attribute Based Encryption Scheme for a Collaborative E-health Cloud", Submitted to SS: *Cloud Computing, Edge Computing, Internet of Things and Big Data Analytics Applications for Healthcare Industry 4.0 of IEEE Transactions on Industrial Informatics (IEEE TII)*, Nov. 2017 (first round); Aug. 2018 (second round).

# Other Publications Not Included in the Dissertation

**Paper 9:** H. S. G. Pussewalage and V. A. Oleshchuk, "A Patient-Centric Attribute Based Access Control Scheme for Secure Sharing of Personal Health Records Using Cloud Computing", in *Proceedings of the 2nd IEEE International Conference on Collaboration and Internet Computing (IEEE CIC'16)*. IEEE, Nov. 2016, pp. 46-53.

# Contents

# List of Figures

# List of Tables

# List of Abbreviations

| | |
|---|---|
| AA | Attribute Authority |
| ABAC | Attribute Based Access Control |
| ABE | Attribute Based Encryption |
| AC | Access Control |
| ACK | Acknowledgment |
| ACL | Access Control List |
| AGT | Aggregated Token |
| AS 1 | Application Scenario 1 |
| AS 2 | Application Scenario 2 |
| BCC | Blockchain Cloud |
| CL | Camenisch-Lysyanskaya |
| CP-ABE | Ciphertext-Policy Attribute Based Encryption |
| CSP | Cloud Service Provider |
| DBDH | Decisional Bilinear Diffie-Hellman |
| DC | Delegation Count |
| DL | Discrete Logarithm |
| DK | Decryption Key |
| DP | Delegation Permission |
| DT | Delegation Token |
| E-health | Electronic Healthcare |
| EHR | Electronic Health Record |
| EDK | Encrypted Decryption Key |
| ERK | Encrypted Re-encryption Key |
| EXP | Expiration Information |
| FHP | Foreign Healthcare Provider |
| GCD | Greatest Common Divisor |
| HC | Healthcare Cloud |
| HIPAA | Health Insurance Portability and Accountability Act |
| IND-CPA | Indistinguishable Under Chosen Plaintext Attacks |

| | |
|---|---|
| Idemix | Identity Mixer |
| JPBC | Java Pairing Based Cryptography |
| KP-ABE | Key-Policy Attribute Based Encryption |
| LHP | Local Healthcare Provider |
| MA-ABE | Multi-Authority Attribute Based Encryption |
| MA-CP-ABE | Multi-Authority Ciphertext-Policy Attribute Based Encryption |
| NIST | National Institute of Standards and Technology |
| PDP | Policy Decision Point |
| PEP | Policy Enforcement Point |
| PHR | Personal Health Record |
| PKI | Public Key Infrastructure |
| POD | Proof of Delegatability |
| PRE | Proxy Re-encryption |
| PR | Policy Repository |
| RBAC | Role Based Access Control |
| RC | Re-encrypted Cipher |
| ST | Signed Token |
| TA | Trusted Authority |
| TS | Timestamp |
| US | United States |

# Chapter 1

# Introduction

*Modern electronic healthcare (e-health) systems constitute collaborative environments in which patients' health information are shared across multiple domains. Given the sensitiveness of information handled, such systems require sophisticated access control mechanisms to not only cater to the associated access demands but also to enforce security and privacy. This dissertation proposes several privacy preserving, attribute based cryptographic constructions which enable us to realize secure, efficient as well as flexible health information sharing environments. In this chapter, we put forward our motivations, identified research questions, contributions together with the dissertation outline.*

The recent advances in information and communication technologies allowed the development of e-health solutions providing a means for efficient sharing of health resources and making it a part of day to day life of the general public. Such systems intend to provide healthcare services with high efficiency and flexibility while establishing a platform for secure sharing of health related data across different healthcare settings and work-flows among different healthcare providers [1]. The discussion on progressing towards e-health received a significant boost especially in United States (US), with the US Institute of Medicine issuing a major report in 1991, indicating the necessity of adopting computer based patient records [2]. With this transition, paper based records advanced to their respective digitized electronic versions. In practice, such records are named with different terminology based on their usage. Electronic health records (EHRs) are generally used to denote health records of patients that are generated and handled mostly by the healthcare professionals. On the other hand, personal health records (PHRs) are used to distin-

guish the health records that are maintained by the patients themselves or perhaps relatives of the patients to monitor the health status regularly. Evidence for the deployment of e-health systems worldwide can be found in [3–7].

From the users' perspective, it is evident that many benefits exist for the users for being part of e-health systems. It is practical to consider that any person will probably seek the help of different care providers during his or her lifetime. Given the fact that a patient's health record is available electronically and can be shared between different care providers, it would definitely help the patients to convey information regarding previous consultations and diagnosis while also helping the healthcare professionals to make better judgments with the help of comprehensive nature of supportive electronic documents [8]. In addition to that, there have been instances reported where medical information of patients were used for various research agendas such as disease tracking by generating national health data repositories from the information gathered from general practitioners and other care deliverers [9, 10].

## 1.1 Motivations

Along with such benefits, there have been questions raised about the security and privacy of health data which had hindered the widespread deployment of such systems. One of the main reasons is that the health information accumulates a lot of private information over time which may affect one's life, social status as well as social stability [11]. Furthermore, given the severity and the sensitiveness of the information handled in an e-health environment, it is evident that the security and privacy of health information actually resemble the privacy of the associated person. As a result, many government organizations have come up with sets of guidelines or acts to enforce legal requirements for the handling of healthcare data as a way of achieving the intended level of security and privacy [12–18]. However, in reality, many incidents can be found where e-health systems become susceptible due to the miss-management and privacy violations of sensitive health data. The findings in [19] provide significant evidence for the above fact, where they have reported more than 22 million healthcare related privacy violations even with the existence of regulations for security and privacy of health information. Moreover, in [20] further statistics reveal the privacy violations by providing examples where personal health information are stolen or acquired without the authorization of legally obliged parties. This emphasizes the need for adopting sophisticated security and privacy measures in order to experience the benefits of e-health systems while assuring users regarding the safety of their private data.

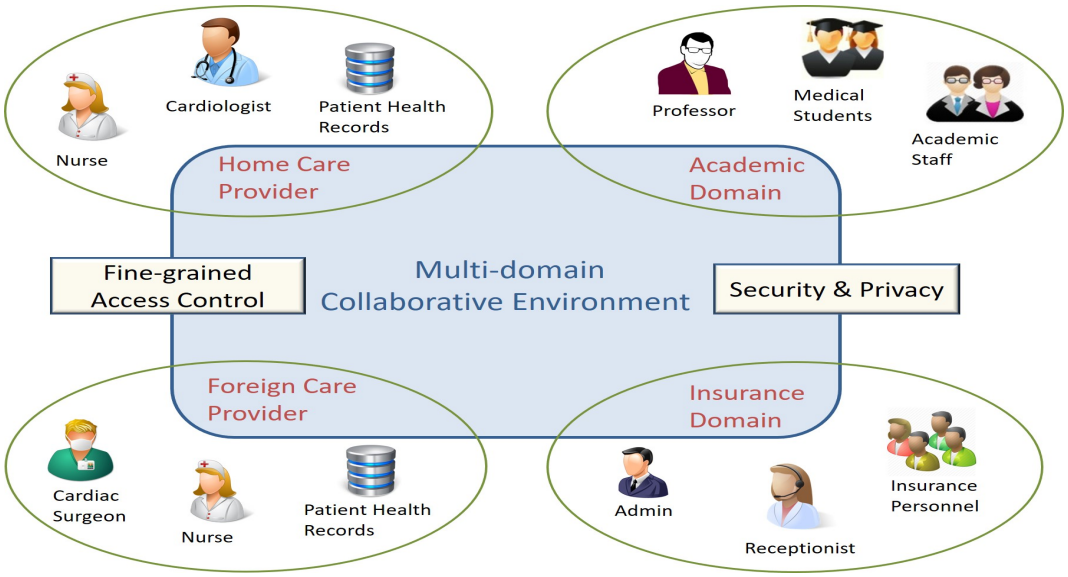



Figure 1.1: Collaborative E-health Environment

### 1.1.1 Access Anonymization

Enforcing privacy of the patients as well as the users who are accessing health information of patients is also quite crucial in constructing a secure and privacy preserving health information sharing environment [21]. Although patient privacy is primarily affected by the illegitimate disclosure of patient health information, it may also be possible to violate privacy by other means. For instance, if the authentication and authorization of access sessions are enforced using user identifiable information, it is quite possible to make premises on the associated patient's conditions, given that access information were traced by an intruder. Also, the linkability of a patient health record to the accessing user's identity could also impact the privacy of the health information accessing user, if this information is exposed to interested third parties [22]. Hence, provisioning anonymous access in a secure manner is also vital for an e-health system.

### 1.1.2 Access Delegatability

As depicted in Figure 1.1, an e-health environment is a collaborative environment meaning that a treatment process of a patient may require collaborative efforts of multiple parties including general practitioners, emergency staff members, specialists in the patient's home care provider as well as professionals from foreign healthcare providers (FHPs) [22, 23]. In such environments, a user's ability to be able to delegate access is quite vital to achieving timely sharing of health information of patients among intended parties to realize flexible health information sharing envi-

ronments [24]. However, it is necessary that access delegatability must be enforced in a controlled manner to ensure that it will not jeopardize the security of the whole system.

### 1.1.3 Cloud Based Deployments

Cloud computing significantly contributes to the development of e-health solutions, since it is capable of providing necessary infrastructure facilities to overcome the issue of managing health data. Thus, care providers have the luxury of outsourcing health data to third-party cloud service providers (CSPs) which not only relieves them from management and maintenance tasks but also helps in maintaining the availability of the data in an efficient, cost-effective manner [25]. However, considering the fact that cloud infrastructures are managed by third parties who may be curious about the stored data, confidentiality, integrity and privacy concerns have been raised on the stored data [26, 27]. Furthermore, given that the outsourcing of data lifts the control of data from its owners may lead to collusion attacks and data modification [28]. A promising approach would be to encrypt the health data before being outsourced to a cloud platform so that the confidentiality of private health data is kept preserved. However, the conundrum that we have to answer is that how we can provision fine-grained access (with and without delegatability) for the intended recipients flexibly and effectively.

### 1.1.4 Access Control (AC) Mechanisms

In order to realize secure and privacy preserving data sharing environments with the aforementioned characteristics, the adoption of suitable AC mechanisms plays a crucial role. An AC mechanism is the process that evaluates whether an access request received from a user or the subject satisfies the access requirements associated with the resource or the object and thereby makes the access decision [29, 30]. Access requirements associated with each object are stored in an authorization database in the form of AC policies, and if a particular access request complies with the AC policies, the subject is granted with the permission to carry out the access stipulated in the access request. If not the subject's request is denied and discarded.

If we consider a health information sharing environment, the set of eligible users who have the right to access a particular health record of a patient might not be registered under the same care provider which stores the health record. For instance, consider the following scenario. Suppose, Alice is a registered cardiac patient in

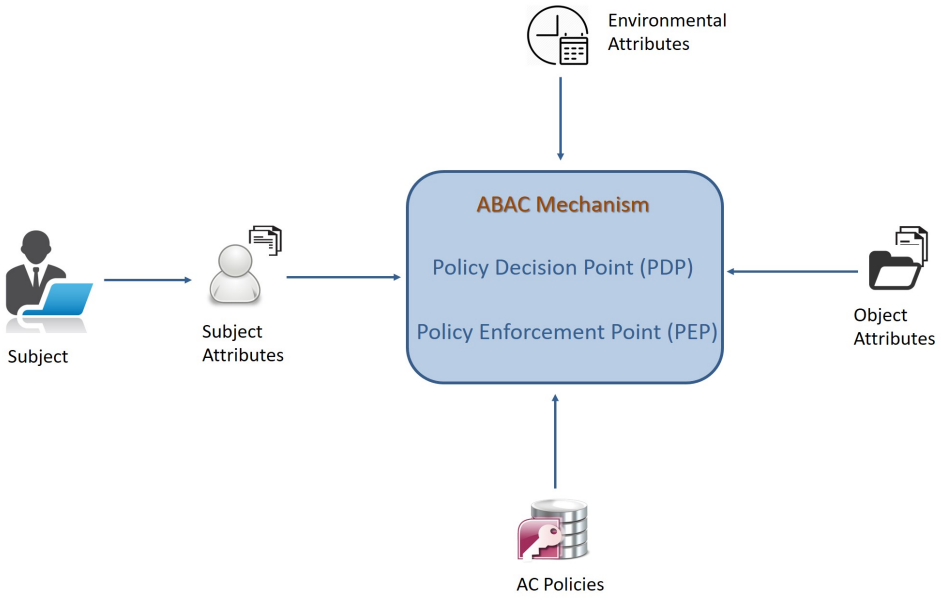



Figure 1.2: Attribute based access control mechanism

hospital A and her health record is stored under the control of hospital A. When Alice is visiting a foreign domain, she feels some discomfort and wants to have a check-up from the foreign domain care provider, hospital B. In such a scenario, it would be ideal if the healthcare professionals associated with the hospital B, are allowed to retrieve the essential parts of Alice's record stored under the control of the hospital A, to provide efficient service to the patient. Furthermore, we emphasized the need for the access delegatability in realizing flexible information sharing environments in Section 1.1.2. In this case, the users who receive access via delegation might also be registered in different entities other than the local care provider, under which the health records are stored.

A variety of AC mechanisms such as access control lists (ACLs) [31, 32], role based access control (RBAC) [33–38] and attribute based access control (ABAC) [39–42] have been widely utilized for ensuring secure and restricted access for computer resources, especially in multi-user data sharing settings. Among them, ABAC is more conducive for establishing fine-grained access in collaborative e-health systems since ABAC inherently supports inter-domain data access which is not the case with RBAC models [43]. However, ABAC is not yet formally standardized while a general guideline for ABAC implementations is published by the National Institute of Standards and Technology (NIST) recently [44]. In ABAC, access decisions are made based on the attributes of the subjects, attributes of the objects as well as environmental attributes and a set of AC policies which defines the allowable operations determined by the relationships between subject-object combinations. We

can illustrate the general principle of operation of an ABAC system as shown in Figure 1.2. When a subject requires to access a specific object, it initiates an access request indicating the object information as well as the subject attributes. Then, the policy decision point (PDP) decides whether to permit or deny the access based on the combination of attributes (subject, object and environmental attributes) and their satisfiability of the stored AC policies. Finally, the decision is enforced through the policy enforcement point (PEP) of the system.

When the resources (for which the access restrictions are imposed) are stored remotely, the aforementioned ABAC model cannot be adopted as it is, since the confidentiality of stored data is also a concern. Furthermore, the integration of conventional encryption schemes is also not an efficient solution due to the data owner's impotence of being able to enable selective, fine-grained access to the stored resources. As a solution attribute based encryption (ABE) [45–47] is introduced, which is capable of granting decryption privileges not to a particular entity but to entities with a defined set of attributes [48]. In the ABE model, similar to the earlier presented ABAC model, each resource is associated with an attribute based access policy (generally called as an access structure in ABE terms) which is either bound to the ciphertext or the secret keys of a user, permitting the users who satisfy the associated access structure to decrypt the ciphertext with the relevant secret keys. Hence, in the broader sense, ABE can be regarded as a part of the scope of ABAC.

In this dissertation, we focus on the issues put forward in Section 1.1.1 - Section 1.1.3, and we propose efficient and flexible ABAC schemes compatible with the needs of collaborative e-health environments.

## 1.2 Research Questions and Contributions

Based on the above mentioned motivations, several research questions are raised. These research questions are addressed with our proposed cryptographic enforcements throughout this dissertation. In this section, we summarize those research questions as well as our contributions.

### 1.2.1 Research Questions

This dissertation intends to answer the following research questions.

- **Question 1 (Q1):** How can we represent the collaborative health information sharing environments subjected to the following application scenarios:

  - *Application Scenario 1 (AS 1):* health information of patients are stored locally under the control of the care provider
  - *Application Scenario 2 (AS 2):* health information of patients are outsourced to a third-party cloud platform

  using appropriate attribute based system models?

- **Question 2 (Q2):** How can we construct efficient ABAC schemes to provide anonymous and fine-grained access for the users (who require access to stored health records of patients) with respect to AS 1?
- **Question 3 (Q3):** How can we enforce access delegatability in a controlled and a secure manner for the users with respect to AS 1?
- **Question 4 (Q4):** How can we construct efficient ABAC schemes to provide anonymous and fine-grained access for the users with respect to AS 2?
- **Question 5 (Q5):** What should be done, to provision delegatable access in a controlled and a secure manner in relation to AS 2?
- **Question 6 (Q6):** In any attribute based system, there may be instances where users need to be revoked from their attributes. How can we achieve this requirement cryptographically?

### 1.2.2 Contributions

In Table 1.1, we have tabulated the chapters along with the papers where the solutions for the identified research questions are proposed. As a means for answering the first research question, we present the two health information sharing scenarios considered for this dissertation in Chapter 2: health records of patients are stored locally under the control of the care provider and the case where health records are stored in a third-party cloud platform. In addition, we have put forward the system models and other background information associated with the two scenarios which provide the base for the cryptographic schemes proposed in the following chapters.

We have answered Q2 in Chapter 3 via proposing two ABAC schemes. Scheme 1 uses a zero-knowledge proof as the basis for provisioning the necessary anonymous access whereas the Scheme 2 utilizes a novel attribute based credential scheme which is superior in performance with respect to existing anonymous, multi-show unlinkable credential schemes. Both schemes allow the users to be anonymous as well as unlinkable across multiple sessions while the second construction achieves

Table 1.1: Tabulation of chapters in the dissertation and the relevant papers where the cryptographic constructions are proposed for the identified research questions. The check marks (✓) indicate the places where each research question is addressed.

| Research Question | Chapters | | | | | Papers | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 2 | 3 | 4 | 5 | 6 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| Q1 | ✓ | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| Q2 | ... | ✓ | ... | ... | ... | ✓ | ✓ | ✓ | ... | ... | ... | ... |
| Q3 | ... | ... | ✓ | ... | ... | ... | ... | ✓ | ✓ | ✓ | ... | ... |
| Q4 | ... | ... | ... | ✓ | ... | ... | ... | ... | ... | ... | ✓ | ✓ |
| Q5 | ... | ... | ... | ... | ✓ | ... | ... | ... | ... | ... | ... | ✓ |
| Q6 | ... | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |

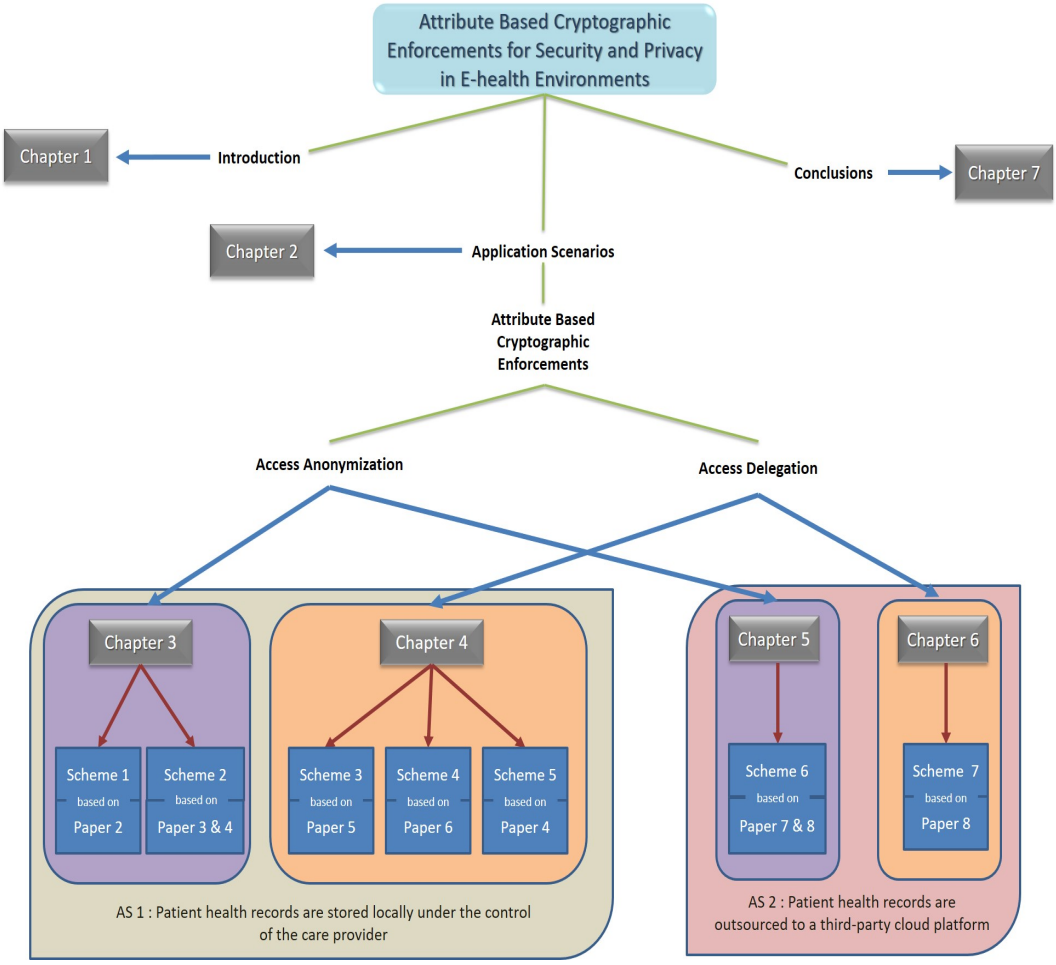



Figure 1.3: Structure of the dissertation

this property with a comparatively lower end-user key management overhead. In addition, the second scheme also equipped with a trap-door mechanism to revoke the anonymity of users.

To answer Q3, we proposed three cryptographic constructions which facilitate controlled delegatability of access to the users who intend to access the health records of patients. The first construction (denoted as Scheme 3) is based on an attribute based signature scheme whereas the second construction (Scheme 4) adopts the blockchain technology. The third construction (Scheme 5) is an extension of the attribute based credential scheme proposed in relation to answering Q2. Among them, Scheme 4 provides pseudo-anonymity guarantees whereas Scheme 5 offers full anonymity for the users.

We have answered Q4 by proposing an efficient ABE scheme in Chapter 5 (Scheme 6) and it is extended to enforce controlled access delegatability to answer Q5 in Chapter 6 (Scheme 7). All of our cryptographic schemes embed mechanisms to revoke attributes from users. Hence, Q6 is answered in Chapter 3 - Chapter 6, as a part of each proposed scheme.

## 1.3 Dissertation Outline

In general terms, this dissertation focuses on constructing attribute based cryptographic schemes to realize flexible and secure health information sharing environments. More precisely, our cryptographic constructions focus on enabling anonymous access and controlled access delegatability for the users who require accessing health records of patients. The rest of the dissertation is organized as follows and the dissertation structure is also illustrated in Figure 1.3.

- In Chapter 2, we present the two health information sharing scenarios (AS 1 and AS 2 introduced in Section 1.2.1) for which the cryptographic schemes are proposed. In addition, the corresponding system models as well as other background information necessary for presenting the proposed schemes in detail in the following chapters, are also specified.

- Chapter 3 describes the proposed ABAC schemes which facilitate anonymous and fine-grained access for the users in the system with respect to AS 1. The content in this chapter is based on the work in Paper 2, Paper 3 and Paper 4.

- In Chapter 4, we present the proposed cryptographic schemes that enforce the capability of access delegatability for the users in the system with respect

to AS 1. This chapter is based on the work in Paper 4, Paper 5 and Paper 6.

- Chapter 5 and Chapter 6 deal with AS 2. In Chapter 5, we propose an AC mechanism influenced by ABE to provide anonymous and fine-grained access to outsourced health records of patients whereas in Chapter 6, the scheme in Chapter 5 is extended to facilitate controlled delegatability of access. Chapter 5 and Chapter 6 are based on our work in Paper 7 and Paper 8.
- Chapter 7 concludes the dissertation while pointing out a few potential extensions to the proposed cryptographic schemes.

# Chapter 2

# Application Scenarios and Notations

*The primary intention of this chapter is to provide the necessary background information to illustrate the proposed cryptographic schemes in the chapters to follow. Therefore, we present the two health information sharing scenarios in detail for which the attribute based cryptographic schemes are proposed. Also, we describe how a health record of a patient is represented along with how the access restrictions are imposed on patient records with respect to the presented application scenarios. The chapter is concluded with a listing of notations and their respective definitions used in the proposed AC schemes.*

## 2.1 Application Scenarios

In this dissertation, we consider two health information sharing scenarios where health data of patients are stored under the control of the care provider and the scenario where patient health data are outsourced to a third-party cloud platform by the care provider. In Chapter 1, we introduced the terms EHR and PHR as the health records of patients handled by the care providers and patients themselves respectively. Therefore, from here on, we call a health record of a patient as the patient's EHR, because we consider that the care provider manages the health records of patients in our application scenarios. In this section, first, we present the structure of a patient's EHR followed by the considered two application scenarios including how access policies are utilized to impose access restrictions on patient EHRs as well as system model assumptions in connection with the considered application scenarios.

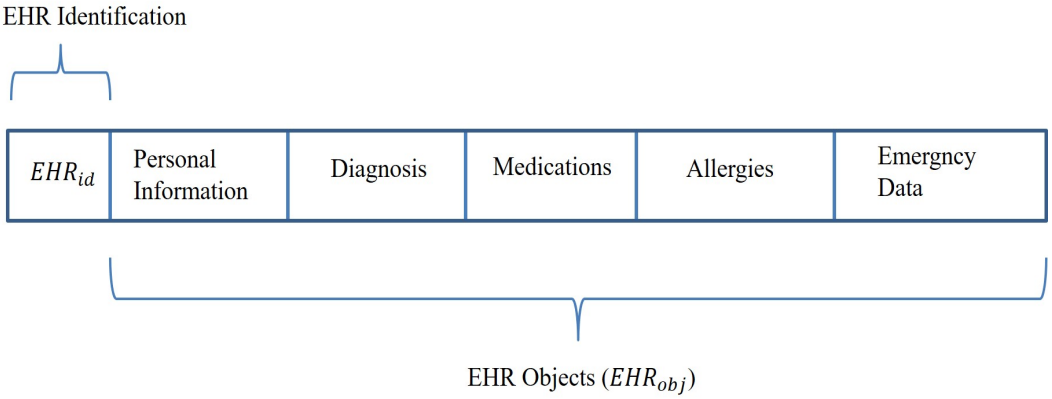



Figure 2.1: Structure of a patient's EHR

### 2.1.1 Structure of a Patient's EHR

We assume that an EHR of a patient maintained by a specific care provider is denoted with a unique identification $EHR_{id}$. An EHR can have many different information categories as personal information, diagnosis, medications, allergies, emergency data, etc. and we define them as EHR objects ($EHR_{obj}$) of a patient's EHR. Hence, a patient's EHR can have many different EHR objects as shown in Figure 2.1.

### 2.1.2 Application Scenario 1 (AS 1)

We consider a local healthcare provider (LHP) which is responsible for providing healthcare services for people who reside in a specific geographical area. Every patient registered in the LHP has an associated EHR, stored locally under the control of the LHP. We require these EHRs to be shared securely among the subjects who are eligible to access them according to the access restrictions imposed on the records. Figure 2.2 represents the system model, and its main components are defined below.

- *Local Healthcare Provider (LHP):* LHP provides healthcare services to its registered patients residing in a specific geographical area. It is composed of an EHR repository, policy repository (PR), and a policy enforcement point (PEP). PR stores the access policies relevant for each EHR stored in the EHR repository while the PEP operates as the point in which the access decisions are made and enforced. Although the general ABAC mechanism presented in Section 1.1.4 uses two entities PDP and PEP to make the decision and enforce the decision, we consider that PEP is responsible for both decision making as well as decision enforcement in our system model.

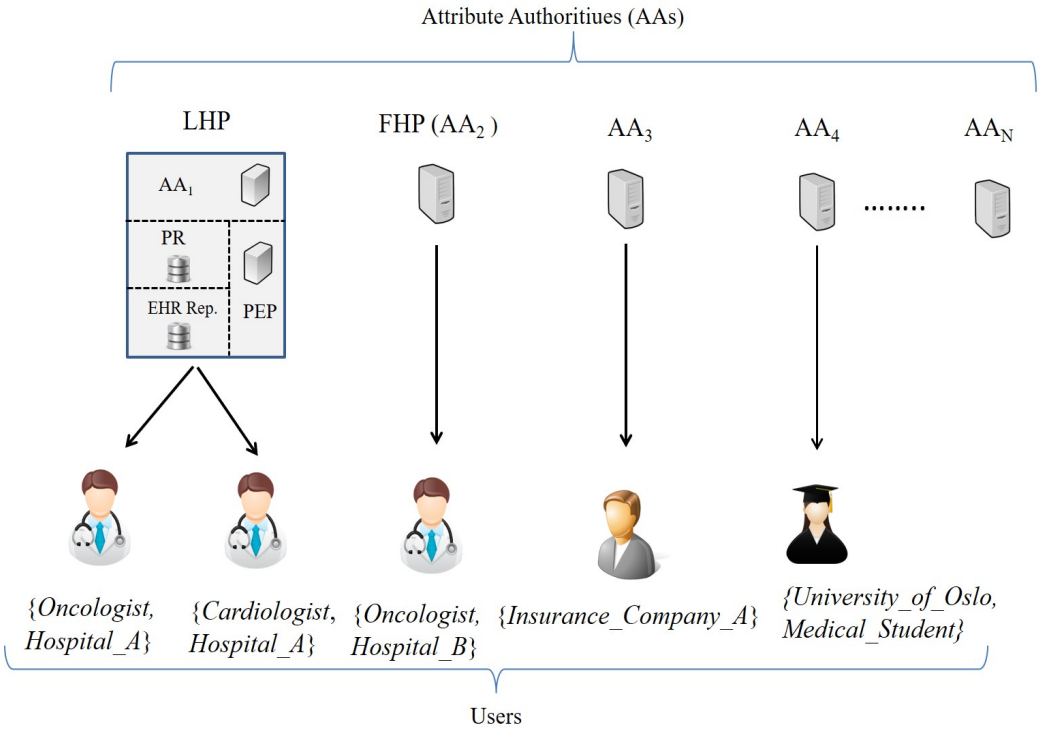



Figure 2.2: System model corresponding to AS 1

- *Attribute Authorities (AAs):* We call attribute issuing authorities as AAs. Each AA is responsible for managing a set of attributes and issuing relevant attributes to users upon validating their eligibility. Further, we assume that all AAs are trusted and LHP also acts as an AA to issue attributes specific to the LHP.

- *Users:* Users are the subjects who are eligible to access the stored EHRs based on their attributes. Subjects include healthcare professionals affiliated with the LHP as well as members of other organizations such as insurance companies, FHPs, etc.

**EHR access policies:** Each $EHR_{obj}$ of a patient's EHR can have one or more access policies which specify the access requirement for accessing the associated $EHR_{obj}$ along with the permissions. Thus, an access policy $\mathcal{P}$ is in the form of,

$$\mathcal{P} : \mathcal{T} \mapsto A$$

where $\mathcal{T}$ is a Boolean statement with logical conjunctions ($\wedge$) and logical disjunctions ($\vee$) combining required subject attributes and "$\mapsto$" has the meaning of "permitted to". $A$ represents the permissions (read, write) that a user will obtain, after

satisfying the access requirement defined by $\mathcal{T}$. For example, consider the following two access policies associated with the $EHR_{obj} = O$, corresponding to the EHR, $EHR_{id} = I$.

$$\mathcal{P}_{I,O} : (Physician \wedge Hospital\_A) \mapsto read \,\&\, write \tag{2.1}$$

$$\mathcal{P}_{I,O} : (Nurse \wedge Hospital\_A) \mapsto read \tag{2.2}$$

The statement (2.1) stipulates that a physician who is employed in hospital A is authorized to carry out read and write operations on the $EHR_{obj}$, $O$ whereas the statement (2.2) states that a nurse who works in hospital A can only read the information.

**System model assumptions:** In AS 1, we assume that the LHP is honest and trusted meaning that it will follow the protocols specified accordingly. It is also assumed that the users may be curious about the stored data and potentially interested in extracting more information than what they are allowed through the access privileges. For instance, a pharmacist would be interested in accessing patient prescriptions and learn prescription patterns of different doctors which could be useful for marketing purposes and boosting profits. To do so, users may use forged attributes, replay attacks as well as collude with other users to gain access to data which cannot be accessed individually.

### 2.1.3 Application Scenario 2 (AS 2)

Similar to AS 1, we consider that LHP provides healthcare services for people who reside in a specific geographical area and every patient who is registered in the LHP has an associated EHR. In contrast to AS 1, LHP stores all the EHRs of patients (which should be shared among the users) in an encrypted format on a third-party cloud platform, which we denote as the healthcare cloud (HC). Hence, a user who satisfies the access restrictions imposed on a specific $EHR_{obj}$ should be able to decrypt the $EHR_{obj}$ upon receiving it from the HC. Figure 2.3 represents the system model corresponding to AS 2.

**EHR access structures and access sub-structures:** In AS 2, we consider that each $EHR_{obj}$ is associated with an attribute based access structure $\mathcal{T}$ which governs the attribute requirement for accessing or decrypting the specific $EHR_{obj}$ stored in HC. An access structure is defined as a Boolean statement with the disjunction ($\vee$) and conjunction ($\wedge$) operations combining the subject attributes. An example access

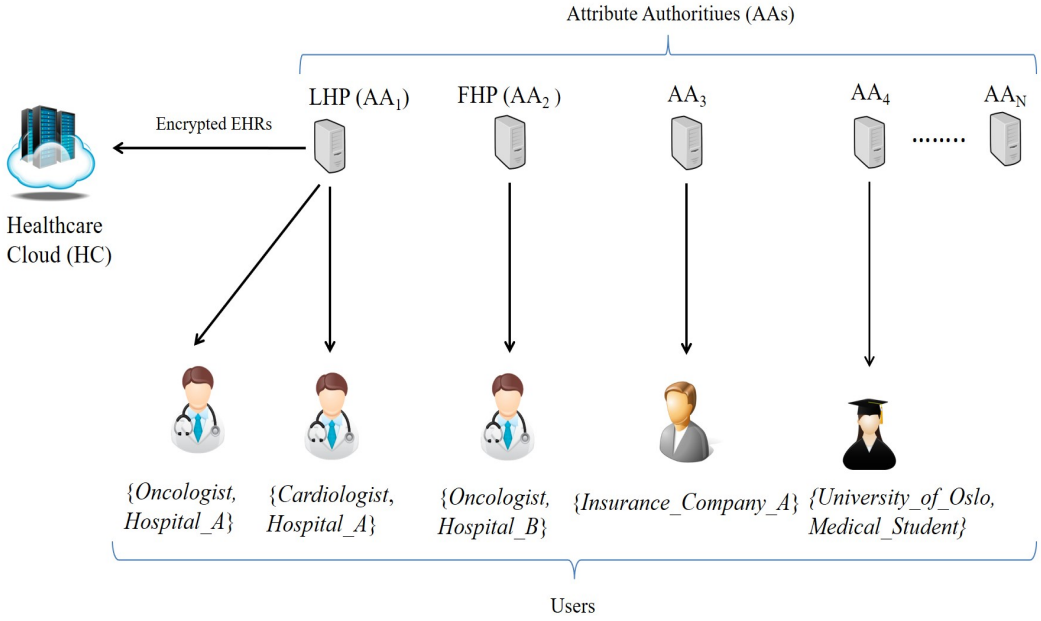



Figure 2.3: System model corresponding to AS 2

structure $\mathcal{T}_{I,O}$ for the $EHR_{obj} = O$ corresponding to $EHR_{id} = I$ is shown below.

$$\mathcal{T}_{I,O} : (Physician \vee Cardiologist) \wedge (Hospital\_A)$$

This statement states that any user who is employed as a cardiologist or a physician at hospital A is authorized to decrypt the $EHR_{obj} = O$ associated with the EHR, where $EHR_{id} = I$. Moreover, we represent an access structure $\mathcal{T}$ as the disjunction of a set of access sub-structures $\{\mathcal{T}_i\}_{i=1,2,...,q}$ such that,

$$\mathcal{T} = \mathcal{T}_1 \vee \mathcal{T}_2 \vee ... \vee \mathcal{T}_q$$

where each $\mathcal{T}_i$ is a conjunction of some subject attributes. We call each $\mathcal{T}_i$ as an access sub-structure of $\mathcal{T}$. For instance, we can represent $\mathcal{T}_{I,O}$ as the disjunction of two sub-structures $\mathcal{T}_1$ and $\mathcal{T}_2$ such that,

$$\mathcal{T}_{I,O} : (Physician \wedge Hospital\_A) \vee (Cardiologist \wedge Hospital\_A)$$

$$\mathcal{T}_1 : (Physician \wedge Hospital\_A)$$
$$\mathcal{T}_2 : (Cardiologist \wedge Hospital\_A).$$

**System model assumptions:** In AS 2, we assume that the LHP is honest and trusted whereas HC is semi-trusted. This means that HC will follow the specified operational protocol while being curious about the stored data. Furthermore, we also

assumed that the users might also be curious about the stored data and potentially interested in extracting more information than what they are allowed via colluding with other users.

## 2.2 Definitions of Notations

We use the notations given in Table 2.1 when illustrating the proposed AC schemes in the following chapters.

Table 2.1: List of notations used in the proposed schemes

| Notation | Description |
|---|---|
| | Scheme 1 |
| $\mathbb{Z}_p^*$ | The set of integers modulo $p$, where $p$ is a large prime |
| $\mathbb{G}_i$ | Multiplicative cyclic group of prime order $p$, where $i = 0, 1$ |
| $e$ | $e : \mathbb{G}_0 \times \mathbb{G}_0 \rightarrow \mathbb{G}_1$ is a bilinear map |
| $g$ | Generator of $\mathbb{G}_0$ |
| $H$ | $H : \{0,1\}^* \rightarrow \mathbb{Z}_p^*$ is a cryptographic hash function |
| $AA_k$ | $k^{th}$ AA in the system |
| $U_m$ | $m^{th}$ user in the system |
| $A^k$ | The attribute set managed by $AA_k$ |
| $A_m$ | The attribute set owned by $U_m$ |
| $A_m^k$ | The subset of attributes owned by $U_m$ acquired from $AA_k$ |
| $\xi$ | Shared secret among AAs |
| $\alpha_k,\ \beta_k$ | Master secrets of $AA_k$ |
| $t_{k,i}$ | Secret attribute exponent corresponding to the $i^{th}$ element in $A^k$ |
| $T_{k,i}$ | Public attribute key corresponding to the $i^{th}$ element in $A^k$ |
| $X_k,\ Y_k$ | Public keys of $AA_k$ |
| $MK_k$ | Master secret set of $AA_k$, where $MK_k = \{\xi, \alpha_k, \beta_k, t_{k,i}\}_{i=1,2,...,\lvert A^k\rvert}$ |
| $PK_k$ | Public tuple of $AA_k$, where $PK_k = \{X_k, Y_k, T_{k,i}\}_{i=1,2,...,\lvert A^k\rvert}$ |
| $SK_{LHP}$ | Private key associated with the PKI certificate of LHP |
| $PK_{LHP}$ | Public key associated with the PKI certificate of LHP |
| $\bar{r}_m$ | $\bar{r}_m = H(ID_m)$, where $ID_m$ denotes the identity of $U_m$ |
| $r_m$ | $r_m \in \mathbb{Z}_p^*$ such that $r_m = \xi + \bar{r}_m$ |
| $sk_0^k$ | Secret key that relates the identities of $AA_k$ and $U_m$ |
| $\{sk_i^k\}_{i=1,...,\lvert A_m^k\rvert}$ | Attribute keys of $U_m$ for the set of attributes $A_m^k$ |
| $SK_m^k$ | Secret key set of $U_m$ received from $AA_k$, where $SK_m^k = \{sk_0^k, sk_i^k\}_{i=1,...,\lvert A_m^k\rvert}$ |
| $TS$ | Timestamp |
| $ACK$ | Acknowledgment message |

| | |
|---|---|
| $K$ | Shared session key established when authenticating LHP by $U_m$ |
| $\{m\}_{PK_{LHP}}$ | Public key encryption of the message $m$ using $PK_{LHP}$ |
| $E(m, K)$ | Symmetric key encryption of the message $m$ using the session key $K$ |
| $A'_m$ | Disclosing attribute set of $U_m$ |
| $d$ | $d = \|A'_m\|$, the number of attributes in $A'_m$ |
| $\{AC_{m,i}\}_{i=1,\ldots,d}$ | Attribute commitments of $U_m$ |
| $KC_m$ | Private key commitment of $U_m$ |
| Scheme 2 | |
| $U_m$ | User $m$ in the system |
| $I$ | Credential issuer |
| $V$ | Credential verifier |
| $\mathbb{Z}_p^*$ | The set of integers modulo $p$, where $p$ is a large prime |
| $\mathbb{G}_i$ | Multiplicative cyclic group of prime order $p$, where $i = 0, 1, T$ |
| $e$ | $e : \mathbb{G}_0 \times \mathbb{G}_1 \rightarrow \mathbb{G}_T$ is a bilinear map |
| $H$ | $H : \{0,1\}^* \rightarrow \mathbb{Z}_p^*$ is a cryptographic hash function |
| $q$ | Generator of $\mathbb{G}_1$ |
| $g_0, \{g_i\}_{i \in A_I}$ | Generators of $\mathbb{G}_0$ |
| $A_I$ | The set of attribute indices of the attributes managed by $I$ |
| $A_m$ | The set of attribute indices of the attributes requested by $U_m$ |
| $A_m^R$ | The set of attribute indices of the disclosing attribute set of $U_m$ |
| $\{a_{1j}\}_{j=1,2,\ldots,l_1}$ | The set of credential expiration values |
| $\{a_i\}_{i=2,\ldots,\|A_I\|}$ | The set of attribute exponents corresponding to attributes in $A_I$ |
| $SK_I = \alpha$ | Master secret of $I$ |
| $PK_I$ | Public key of $I$ |
| $M$ | Aggregated attribute component |
| $C_m$ | Credential public key of the credential issued to $U_m$ |
| $\beta_m$ | Credential private key of the credential issued to $U_m$ |
| $S_{m,1},\ S_{m,2}$ | Signature components of $C_m$ |
| $K$ | Credential randomization exponent |
| $C'_m$ | Randomized $C_m$ |
| $S'_{m,1},\ S'_{m,2}$ | Randomized signature components of $S_{m,1}$ and $S_{m,2}$ |
| Scheme 3 | |
| $\mathbb{Z}^{\geq 0}$ | $\mathbb{Z}^+ \cup \{0\}$ |
| $\mathbb{Z}_p^*$ | The set of integers modulo $p$, where $p$ is a large prime |
| $\mathbb{G}_i$ | Multiplicative cyclic group of prime order $p$, where $i = 0, 1$ |
| $e$ | $e : \mathbb{G}_0 \times \mathbb{G}_0 \rightarrow \mathbb{G}_1$ is a bilinear map |
| $g$ | Generator of $\mathbb{G}_0$ |
| $H$ | $H : \{0,1\}^* \rightarrow \mathbb{Z}_p^*$ is a cryptographic hash function |

| | |
|---|---|
| $AA_k$ | $k^{th}$ AA in the system |
| $U_m$ | $m^{th}$ user in the system |
| $A^k$ | The attribute set managed by $AA_k$ |
| $A_m$ | The attribute set owned by $U_m$ |
| $A'_m$ | The attribute subset owned by $U_m$ that satisfies $\mathcal{T}$ |
| $\alpha_k$ | Private key of $AA_k$ |
| $\beta_m$ | Private key of $U_m$ |
| $t_{k,i}$ | Secret attribute exponent corresponding to the $i^{th}$ element in $A^k$ |
| $t'_{m,i}$ | Secret attribute exponent corresponding to the $i^{th}$ element in $A_m$ |
| $T_{k,i}$ | Public attribute key corresponding to the $i^{th}$ element in $A^k$ |
| $T'_{m,i}$ | Public attribute key corresponding to the $i^{th}$ element in $A_m$ |
| $X_k$ | Public key of $AA_k$ |
| $X'_m$ | Public key of $U_m$ |
| $MK_k$ | Master secret set of $AA_k$, where $MK_k = \{\alpha_k, t_{k,i}\}_{i=1,2,...,\|A^k\|}$ |
| $MK'_m$ | Master secret set of $U_m$, where $MK'_m = \{\beta_m, t'_{m,i}\}_{i=1,2,...,\|A_m\|}$ |
| $PK_k$ | Public tuple of $AA_k$, where $PK_k = \{X_k, T_{k,i}\}_{i=1,2,...,\|A^k\|}$ |
| $PK'_m$ | Public tuple of $U_m$, where $PK'_m = \{X'_m, T'_{m,i}\}_{i=1,2,...,\|A_m\|}$ |
| $PKI_k$ | PKI certificate of $AA_k$ |
| $PKI'_m$ | PKI certificate of $U_m$ |
| $AT^k_m$ | Assignment token issued by $AA_k$ to $U_m$ |
| $SIG^k_m$ | Signature of $AT^k_m$ |
| $AT_{m,n}$ | Delegation token which represents the delegation from $U_m$ to $U_n$ |
| $AGT_{m,n}$ | Aggregated token issued by $U_m$ to $U_n$ |
| Scheme 4 | |
| $\mathbb{Z}^+$ | The set of positive integers |
| $\mathbb{Z}^{\geq 0}$ | $\mathbb{Z}^+ \cup \{0\}$ |
| $\mathbb{Z}^*_{\bar{n}}$ | The set of integers modulo $\bar{n}$, where $\bar{n}$ is a large prime |
| $H$ | Cryptographic hash function |
| $\phi$ | Euler's phi-function |
| $GCD(A, B)$ | Greatest common divisor of $A$ and $B$ |
| $AA_k$ | $k^{th}$ AA in the system |
| $U_m$ | $m^{th}$ user in the system |
| $BC_k$ | Blockchain that records transactions associated with attributes of $AA_k$ |
| $B_{k,i}$ | $i^{th}$ block of the blockchian $BC_k$ |
| $C_{k,i}$ | Block contents of the block $B_{k,i}$ |
| $B_{k,i}(1)$ | $1^{st}$ output of the $i^{th}$ block of the blockchian $BC_k$ |
| $PKI_k$ | PKI certificate of $AA_k$ |
| $PK_k$ | RSA public key associated with the PKI certificate of $AA_k$ |

| | |
|---|---|
| $SK_k$ | RSA secret key associated with the PKI certificate of $AA_k$ |
| $PKI_{BC}$ | PKI certificate of the blockchain cloud |
| $PK_{BC}$ | RSA public key associated with the PKI certificate of the blockchain cloud |
| $SK_{BC}$ | RSA secret key associated with the PKI certificate of the blockchain cloud |
| $PI_m$ | Pseudo-identity of $U_m$ |
| $PK'_m$ | RSA public key of $U_m$ |
| $SK'_m$ | RSA secret key of $U_m$ |
| $S_m^k$ | A random seed used for the attribute assignment from $AA_k$ to $U_m$ |
| $TS_m^k$ | Expiration timestamp corresponding to the attribute assignment from $AA_k$ to $U_m$ |
| $M_m^k$ | Hash of the block representing the attribute assignment from $AA_k$ to $U_m$ |
| $\sigma_m^k$ | RSA signature of $M_m^k$ generated by $AA_k$ |
| $S_{m,n}$ | A random seed used for the attribute delegation from $U_m$ to $U_n$ |
| $TS_{m,n}$ | Expiration timestamp corresponding to the attribute delegation from $U_m$ to $U_n$ |
| $M_{m,n}$ | Hash of the block representing the attribute delegation from $U_m$ to $U_n$ |
| $\sigma_{m,n}$ | RSA signature of $M_{m,n}$ generated by $U_m$ |
| $S'_{m,n}$ | A random seed used for revoking an attribute from $U_n$ by $U_m$ |
| $M'_{m,n}$ | Hash of the block representing the revocation of an attribute from $U_n$ by $U_m$ |
| $\sigma'_{m,n}$ | RSA signature of $M'_{m,n}$ generated by $U_m$ |
| | Scheme 5 |
| $U_m$ | User $m$ in the system |
| $I$ | Credential issuer |
| $V$ | Credential verifier |
| $\mathbb{Z}_p^*$ | The set of integers modulo $p$, where $p$ is a large prime |
| $\mathbb{G}_i$ | Multiplicative cyclic group of prime order $p$, where $i = 0, 1, T$ |
| $e$ | $e : \mathbb{G}_0 \times \mathbb{G}_1 \rightarrow \mathbb{G}_T$ is a bilinear map |
| $H$ | $H : \{0,1\}^* \rightarrow \mathbb{Z}_p^*$ is a cryptographic hash function |
| $q$ | Generator of $\mathbb{G}_1$ |
| $g_0, \{g_i\}_{i \in A_I}$ | Generators of $\mathbb{G}_0$ |
| $A_I$ | The set of attribute indices of the attributes managed by $I$ |
| $A_m$ | The set of attribute indices of the attributes requested by $U_m$ |
| $A_m^R$ | The set of attribute indices of the disclosing attribute set of $U_m$ |
| $\{a_{1j}\}_{j=1,2,...,l_1}$ | The set of credential expiration values |
| $\{a_{2j}\}_{j=1,2,...,l_2}$ | The set of values defining the length of a delegation chain |
| $\{a_i\}_{i=3,...,\lvert A_I \rvert}$ | The set of attribute exponents corresponding to attributes in $A_I$ |
| $SK_I = \alpha$ | Master secret of $I$ |
| $PK_I$ | Public key of $I$ |
| $PK_V$ | Public key of $V$ |
| $M$ | Aggregated attribute component |

| | |
|---|---|
| $C_m$ | Credential public key of the credential issued to $U_m$ |
| $\beta_m$ | Credential private key of the credential issued to $U_m$ |
| $S_{m,1},\ S_{m,2}$ | Signature components of $C_m$ |
| $C'_m$ | Randomized $C_m$ |
| $S'_{m,1},\ S'_{m,2}$ | Randomized signature components of $S_{m,1}$ and $S_{m,2}$ |
| $POD(m,n)$ | POD issued from $U_m$ to $U_n$ |
| $T(m,n)$ | Tag issued from $U_m$ to $U_n$ |
| $C_{mn}$ | Credential public key of the delegated credential issued by $U_m$ to $U_n$ |
| $S_{mn,1},\ S_{mn,2}$ | Signature components of $C_{mn}$ |
| $C'_{mn}$ | Randomized $C_{mn}$ |
| $S'_{mn,1},\ S'_{mn,2}$ | Randomized signature components of $S_{mn,1}$ and $S_{mn,2}$ |
| | Scheme 6 |
| $\mathbb{Z}_p^*$ | The set of integers modulo $p$, where $p$ is a large prime |
| $\mathbb{G}_i$ | Multiplicative cyclic group of prime order $p$, where $i = 0, 1$ |
| $e$ | $e : \mathbb{G}_0 \times \mathbb{G}_0 \rightarrow \mathbb{G}_1$ is a bilinear map |
| $g$ | Generator of $\mathbb{G}_0$ |
| $H$ | $H : \{0,1\}^* \rightarrow \mathbb{Z}_p^*$ is a cryptographic hash function |
| $AA_k$ | $k^{th}$ AA in the system |
| $U_m$ | $m^{th}$ user in the system |
| $A^k$ | The attribute set managed by $AA_k$ |
| $A_m$ | The attribute set owned by $U_m$ |
| $A_m^k$ | The subset of attributes owned by $U_m$ acquired from $AA_k$ |
| $A_m^{'}$ | The subset of attributes owned by $U_m$ that satisfy the sub-structure $\mathcal{T}'$ |
| $\xi$ | Shared secret among AAs |
| $\alpha_k,\ \beta_k$ | Master secrets of $AA_k$ |
| $t_{k,i}$ | Secret attribute exponent corresponding to the $i^{th}$ element in $A^k$ |
| $T_{k,i}$ | Public attribute key corresponding to the $i^{th}$ element in $A^k$ |
| $X_k,\ Y_k$ | Public keys of $AA_k$ |
| $MK_k$ | Master secret set of $AA_k$, where $MK_k = \{\xi, \alpha_k, \beta_k, t_{k,i}\}_{i=1,2,...,\lvert A^k\rvert}$ |
| $PK_k$ | Public tuple of $AA_k$, where $PK_k = \{X_k, Y_k, T_{k,i}\}_{i=1,2,...,\lvert A^k\rvert}$ |
| $\bar{r}_m$ | $\bar{r}_m = H(ID_m)$, where $ID_m$ denotes the identity of $U_m$ |
| $r_m$ | $r_m \in \mathbb{Z}_p^*$ such that $r_m = \xi + \bar{r}_m$ |
| $sk_0^k$ | Secret key that relates the identities of $AA_k$ and $U_m$ |
| $\{sk_i^k\}_{i=1,...,\lvert A_m^k\rvert}$ | Attribute keys of $U_m$ for the set of attributes $A_m^k$ |
| $SK_m^k$ | Secret key set of $U_m$ received from $AA_k$, where $SK_m^k = \{sk_0^k, sk_i^k\}_{i=1,...,\lvert A_m^k\rvert}$ |
| $M$ | Plaintext |
| $E(M)$ | Ciphertext of $M$ generated with the access structure $\mathcal{T}$ |
| $E_k$ | Ciphertext of $M$ corresponding to the access sub-structure $\mathcal{T}_k$ |

| Scheme 7 | |
|---|---|
| $\mathbb{Z}_p^*$ | The set of integers modulo $p$, where $p$ is a large prime |
| $\mathbb{G}_i$ | Multiplicative cyclic group of prime order $p$, where $i = 0, 1$ |
| $e$ | $e : \mathbb{G}_0 \times \mathbb{G}_0 \rightarrow \mathbb{G}_1$ is a bilinear map |
| $g_0, g_1$ | Generators of $\mathbb{G}_0$ |
| $H_1$ | $H_1 : \{0,1\}^* \rightarrow \mathbb{Z}_p^*$ is a cryptographic hash function |
| $H_2$ | $H_2 : \mathbb{G}_1 \rightarrow \mathbb{Z}_p^*$ is a cryptographic hash function |
| $AA_k$ | $k^{th}$ AA in the system |
| $U_m$ | $m^{th}$ user in the system |
| $A^k$ | The attribute set managed by $AA_k$ |
| $A_m$ | The attribute set owned by $U_m$ |
| $A_m^k$ | The subset of attributes owned by $U_m$ acquired from $AA_k$ |
| $A_m^{'}$ | The subset of attributes owned by $U_m$ that satisfy the sub-structure $\mathcal{T}'$ |
| $\xi$ | Shared secret among AAs |
| $\alpha_k,\ \beta_k$ | Master secrets of $AA_k$ |
| $t_{k,i}$ | Secret attribute exponent corresponding to the $i^{th}$ element in $A^k$ |
| $T_{k,i}, D_{k,i}$ | Public keys corresponding to the $i^{th}$ element in $A^k$ |
| $X_k,\ Y_k,\ Z_k$ | Public keys of $AA_k$ |
| $MK_k$ | Master secret set of $AA_k$, where $MK_k = \{\xi, \alpha_k, \beta_k, t_{k,i}\}_{i=1,2,...,\|A^k\|}$ |
| $PK_k$ | Public tuple of $AA_k$, where $PK_k = \{X_k, Y_k, Z_k, T_{k,i}, D_{k,i}\}_{i=1,2,...,\|A^k\|}$ |
| $\bar{r}_m$ | $\bar{r}_m = H(ID_m)$, where $ID_m$ denotes the identity of $U_m$ |
| $r_m$ | $r_m \in \mathbb{Z}_p^*$ such that $r_m = \xi + \bar{r}_m$ |
| $sk_0^k$ | Secret key that relates the identities of $AA_k$ and $U_m$ |
| $\{sk_i^k\}_{i=1,...,\|A_m^k\|}$ | Attribute keys of $U_m$ for the set of attributes $A_m^k$ |
| $SK_m^k$ | Secret key set of $U_m$ received from $AA_k$, where $SK_m^k = \{sk_0^k, sk_i^k\}_{i=1,...,\|A_m^k\|}$ |
| $RK$ | Re-encryption key used to re-encrypt the ciphertext |
| $ERK$ | Encrypted re-encryption key |
| $DK$ | Decryption key |
| $EDK$ | Encrypted decryption key |
| $M$ | Plaintext |
| $E(M)$ | Ciphertext of $M$ generated with the access structure $\mathcal{T}$ |
| $E_k$ | Ciphertext of $M$ corresponding to the access sub-structure $\mathcal{T}_k$ |
| $RC$ | Re-encrypted ciphertext |
| $DT$ | Delegation token |
| $(\sigma_1, \sigma_2, \sigma_3)$ | Signature of $DT$ |
| $ST$ | Signed delegation token |

# Chapter 3

# Anonymous ABAC Schemes for AS 1

*This chapter presents the proposed anonymous ABAC cryptographic schemes for secure sharing of EHRs concerning AS 1. We start the chapter by summarizing the most prominent research efforts which propose AC mechanisms for similar health information sharing scenarios. Then, the proposed Scheme 1 is presented in detail followed by the motivations for Scheme 2 and its construction. Scheme 1 is based on our work in Paper 2 whereas Scheme 2 is based upon the work in Paper 3 and Paper 4. The chapter is concluded with some concluding remarks.*

## 3.1 Related Work

The results published in [31] initially paved the way to signify the importance of establishing policy driven AC models to secure patients private data. However, the access policies of this solution were primarily based on ACLs which is highly inefficient, considering the fact that each EHR needs to be accompanied by a separate ACL.

Therefore, the focus was shifted towards using RBAC models, in which authorization decisions were made based on the roles associated with users. A role generally defines a job function for which the users are assigned. Furthermore, access privileges are assigned to each user role based on the need-to-know principle. Given that the RBAC reference model put forward by NIST [35] in 2001 is considered to be compatible with the requirements of Health Insurance Portability and Accountability Act (HIPAA) guidelines for accessing patient health records [49]; a variety of privacy aware RBAC solutions to provide AC in shared EHR repositories

are found in [50–57]. The main drawback associated with all those solutions is the fact that every user needs to be registered in the system. If we consider a situation where a person is met with an emergency in a foreign domain, it would be difficult to provide seamless access to the relevant EHR held in the local domain for a doctor registered in the foreign domain due to the fact that the doctor is not registered in the local healthcare facility in which the patient's EHR is stored. This limits the effectiveness of the healthcare delivery process considering its collaborative nature as pointed out in Section 1.1. In comparison, ABAC uses the attributes to make access decisions instead of organizational roles which makes ABAC more conducive in adapting to inter-domain access requirements associated with EHR management systems. However, we could not find any notable research efforts in which anonymous ABAC schemes have been utilized in this context. To address this research gap, we propose two anonymous ABAC schemes which are presented in detail in the following sections.

## 3.2 Security and Privacy Requirements

Before we present the proposed schemes, first we introduce the security and privacy requirements to be satisfied in the proposed anonymous ABAC schemes with respect to AS 1.

- *Selective disclosure of attributes:* Users should be able to gain access to the required EHR data by disclosing only the minimum set of attributes that satisfy the associated access policy.

- *Resistance against attribute forgery:* Users should not be able to generate valid attributes without engaging in the attribute issuance protocol with the AA that manages the attributes.

- *Resistance against attribute collusion:* Users should not be able to gain access to any stored EHR by colluding attributes with other users.

- *Resistance against replay attacks:* Schemes should exhibit resistance to attacks mounted via replaying of intercepted attribute proofs to gain access to EHR data of patients.

- *Attribute revocation:* The process of revoking an issued attribute from a user is termed as attribute revocation. When an attribute is revoked from a user, the user should not be able to gain access with the help of the revoked attribute.

- *Patient privacy:* It is important to prevent any sort of unauthorized disclosure of patients' health information. Therefore, any user who does not possess enough attributes to satisfy the access policy must be prevented from granting the access to the stored EHR data of patients. Prevention of attacks mounted via attribute forgery, attribute collusion, replay attacks and the ability to revoke attributes when necessary, contributes to strengthening the patient privacy.

- *User privacy:* The linkability of a patient's EHR to the accessing user's identity could impact the privacy of the EHR accessing user when such information is being exposed to interested parties. Thus, users must be allowed to access EHR data in an anonymous manner. In addition, the property of selective disclosure of attributes and the unlinkability of a user's access sessions further strengthens the privacy of users.

## 3.3 Proposed Scheme 1

We present this scheme, starting with an overview, some preliminary information followed by the four phases in the proposed scheme: system initialization, attribute key distribution, AC mechanism and the attribute revocation mechanism. This section is concluded with the security analysis of Scheme 1.

### 3.3.1 Overview of Scheme 1

We refer to the system model illustrated in Figure 2.2. The system composed of multiple AAs, each responsible for a different set of attributes. Furthermore, every AA publishes a set of public parameters related to the attributes governed by the corresponding AA.

Users can obtain attributes in the form of attribute keys from relevant AAs, by providing evidence that they are eligible for the requesting attributes. When a user needs to access a particular $EHR_{obj}$, the access requesting user initiates an EHR access request indicating the $EHR_{id}$, $EHR_{obj}$ information of the desired EHR as well as the actions intend to perform on the requested object. Then, the LHP will build up an access structure $\mathcal{T}$ using the stored policies relevant for the requested $EHR_{obj}$ and sends it to the access requester. The access requester will then identify a subset of attributes which he owns, that satisfies the received $\mathcal{T}$ and computes a set of user key commitments according to the selected subset of attributes. These commitments allow the PEP of LHP to construct a zero-knowledge proof with the

help of public parameters of the committed attributes advertised by the respective AAs. A successful computation of a zero-knowledge proof provides evidence that the access requester owns a set of attributes that satisfy the access policy associated with the requested $EHR_{obj}$.

### 3.3.2 Preliminaries

In our method, we have used the properties of bilinear pairings [58–60] for the procedure of user commitment generation as well as for the construction of the adopted zero-knowledge proof which is used by the PEP to verify access legitimacy of users.

**Definition 1 (Bilinear pairings):** Consider two multiplicative cyclic groups $\mathbb{G}_0$, $\mathbb{G}_1$ of prime order $p$ and $g$ be a generator of $\mathbb{G}_0$. Thus, $e$ is considered as a bilinear map $e : \mathbb{G}_0 \times \mathbb{G}_0 \rightarrow \mathbb{G}_1$, if the following properties are held.

1. Bilinearity: $\forall u, v \in \mathbb{G}_0$ and $a, b \in \mathbb{Z}_p$, the condition $e(u^a, v^b) = e(u, v)^{ab}$ stands.
2. Non-degeneracy: $e(g, g) \neq 1$.

The group $\mathbb{G}_0$ is considered as a bilinear group if both the group operations in $\mathbb{G}_0$ and $e : \mathbb{G}_0 \times \mathbb{G}_0 \rightarrow \mathbb{G}_1$ are efficiently computable [60].

### 3.3.3 System Initialization

The system is initialized by first generating a set of global parameters which are shared among all AAs. AAs agree on two multiplicative cyclic groups $\mathbb{G}_0$, $\mathbb{G}_1$ of prime order $p$ with $g$ being a generator of $\mathbb{G}_0$ and a bilinear map $e : \mathbb{G}_0 \times \mathbb{G}_0 \rightarrow \mathbb{G}_1$ along with a secure hash function $H : \{0, 1\}^* \rightarrow \mathbb{Z}_p^*$ that maps each user identity string to a unique value in $\mathbb{Z}_p^*$. The user identity should be a unique identifier for a given user such as, for example, a public key corresponding to the public key infrastructure (PKI) certificate. In addition, AAs also agree on a shared secret $\xi \in \mathbb{Z}_p^*$. Then, each AA publishes the set of global public parameters $(\mathbb{G}_0, \mathbb{G}_1, H, e, g, p)$. Therefore, any new AA can be simply globally initialized by acquiring the set of global parameters as well as the shared secret $\xi$ which are shared by the existing AAs. After the sharing of global parameters, each AA should be locally initialized and the initialization procedure is described below by denoting the $k^{th}$ AA as $AA_k$.

- First of all, $AA_k$ selects two random exponents $\alpha_k, \beta_k \in \mathbb{Z}_p^*$ and computes $X_k = g^{\beta_k}, Y_k = g^{\alpha_k}$.

- Let us assume that $AA_k$ is responsible for the attribute set $A^k$. Then, a unique identifier $t_{k,i} \in \mathbb{Z}_p^*$ for each element $i$ in $A^k$ is also randomly selected. Furthermore, each attribute $i$ administered by $AA_k$ is also bound with a public attribute key $T_{k,i}$, where $T_{k,i} = g^{t_{k,i}}$.
- $AA_k$ will keep $\{\xi, \alpha_k, \beta_k, t_{k,i}\}_{i=1,2,...,|A^k|}$ as the master secret set $MK_k$ and publish $\{X_k, Y_k, T_{k,i}\}_{i=1,2,...,|A^k|}$ as the public tuple $PK_k$.

In addition, we assume that each entity (including AAs and users) has a PKI certificate and an associated public, private key pair. Users and AAs use their respective keys to establish a mutually authenticated channel to facilitate distribution of attribute secret keys.

### 3.3.4 Attribute Key Distribution

Users are allowed to obtain attribute keys from the relevant AAs by providing evidence for the fact that they satisfy the requirements to ascertain the requested attributes. Let us assume that the $m^{th}$ user denoted with $U_m$ wants to acquire attribute keys for the set of attributes $A_m$. In addition, assume that $A_m^k \subseteq A^k$ denotes the set of attributes that $U_m$ needs to acquire from $AA_k$. The attribute key assignment should be carried out over a mutually authenticated channel between $U_m$ and $AA_k$ and we assume that this is already established through their respective keys associated with PKI certificates. The adopted protocol for acquiring attribute keys from $AA_k$ is described below.

- $AA_k$ first uses the hash function $H$ to map the identity of $U_m$ to a unique identifier $\bar{r}_m \in \mathbb{Z}_p^*$ and computes $r_m = \bar{r}_m + \xi$.
- Then, a secret key for each requesting attribute $i$ is generated as described below. If the secret key set is denoted by $SK_m^k$,

$$SK_m^k = \{sk_0^k, sk_i^k\}_{i=1,2,...,|A_m^k|}$$

and,

$$sk_0^k = g^{\frac{\alpha_k - r_m}{\beta_k}} \tag{3.1}$$

$$sk_i^k = g^{\frac{r_m}{t_{k,i}}} \tag{3.2}$$

where $t_{k,i}$ is the master secret component of the $i^{th}$ attribute in $A_m^k$ defined by $AA_k$. Note that the secret key component $sk_0^k$ relates the user identity to the

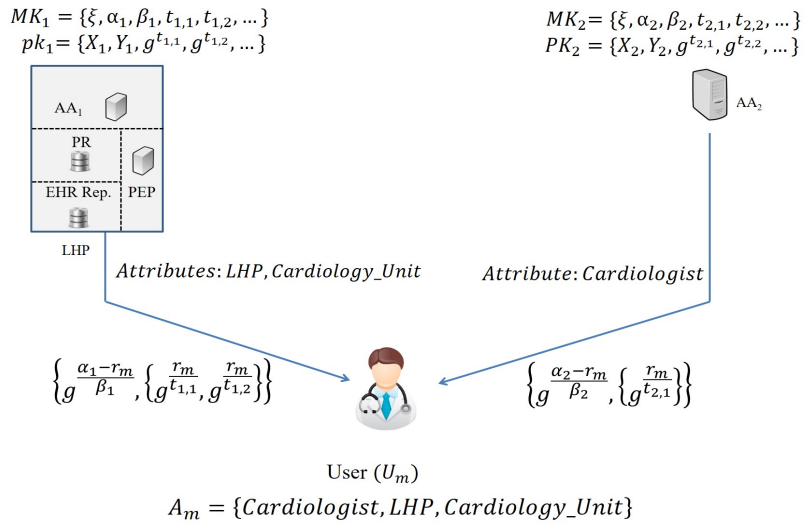



Figure 3.1: Attribute distribution scenario for Scheme 1

identity of the issuing authority $AA_k$ whereas the secret key component $sk_i^k$ relates the user identity to the $i^{th}$ attribute in $A_m^k$. The generated secret key set is sent to $U_m$.

Figure 3.1 shows an attribute distribution scenario, which is used as an example to illustrate the distribution of attributes. In this example, the user $U_m$'s attribute set would be $\{Cardiologist, LHP, Cardiology\_Unit\}$ whereas the attribute subset $\{LHP, Cardiology\_Unit\}$ is ascertained from $AA_1$ while the attribute $Cardiologist$ is obtained from $AA_2$. Let us assume that the mapped identifier corresponding to the user's identity is $r_m$ and the master secret components of attributes $LHP$, $Cardiology\_Unit$ at $AA_1$ is $t_{1,1}$ and $t_{1,2}$. Furthermore, the master secret component of the attribute $Cardiologist$ at $AA_2$ is given by $t_{2,1}$. Then, the user key set received from $AA_1$ is given by $\{g^{(\alpha_1 - r_m)/\beta_1}, \{g^{r_m/t_{1,1}}, g^{r_m/t_{1,2}}\}\}$ while the key set received from $AA_2$ is given by $\{g^{(\alpha_2 - r_m)/\beta_2}, g^{r_m/t_{2,1}}\}$.

### 3.3.5 AC Mechanism

Suppose, $U_m$ wants to access the $EHR_{obj} = O$ of the EHR, $EHR_{id} = I$. We divide the authorization mechanism into two steps. In Step 1, $U_m$ establishes a shared component $Z$ with LHP while ensuring that $U_m$ is in communication with LHP. In Step 2, we present how the attribute key commitments are generated as well as the zero-knowledge proof generation procedure which enables the LHP to learn whether $U_m$ possesses a valid set of attributes that satisfies the access requirement.

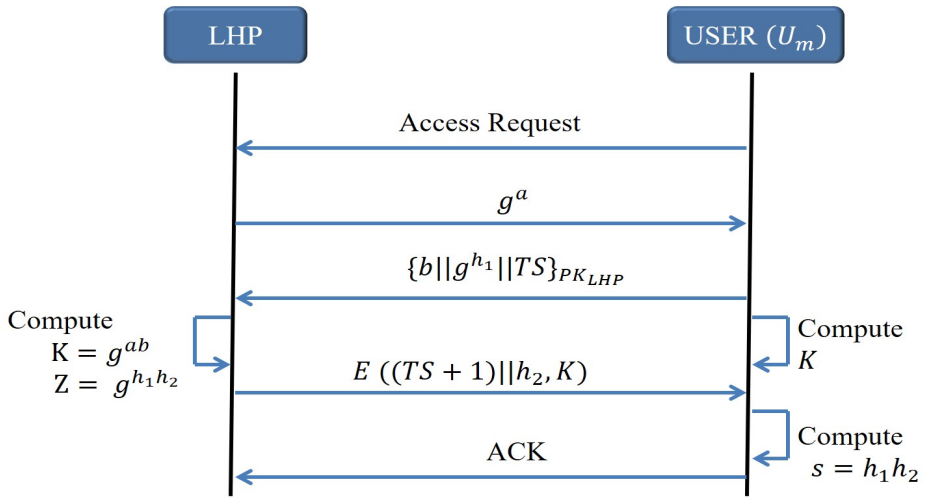



Figure 3.2: Flow diagram associated with Step 1 of the AC mechanism in Scheme 1

**Step 1:** Let us assume that the public and private key pair associated with the PKI certificate of LHP is given by $(PK_{LHP}, SK_{LHP})$. Step 1 is initiated at the user's end by sending an access request to LHP. After the reception of the access request, LHP generates the secret exponent $a \in \mathbb{Z}_p^*$ and sends the reply message of $g^a$. Then, $U_m$ selects two secret exponents $b, h_1 \in \mathbb{Z}_p^*$ and encrypts $b||g^{h_1}$ along with the current timestamp (TS) using $PK_{LHP}$ and forwards $\{b||g^{h_1}||TS\}_{PK_{LHP}}$ back to LHP. LHP uses $SK_{LHP}$ to decrypt the received cipher and computes the shared session key $K = g^{ab}$. Furthermore, LHP generates a new secret exponent $h_2 \in \mathbb{Z}_p^*$ and computes $Z = g^{h_1h_2}$. In order to make it feasible for $U_m$ to authenticate LHP, LHP sends the reply $E((TS+1)||h_2, K)$ by encrypting the message $(TS+1)||h_2$ using the established session key $K$. If the decryption reveals the message $(TS+1)$, it provides evidence for the fact that $U_m$ has successfully established a secure connection with LHP. Finally, $U_m$ computes $s = h_1h_2$ and confirms the authentication of LHP by sending an acknowledgment (ACK) message. The component $s$ will be used by $U_m$ to construct the attribute key commitments while the component $Z$ will be used by LHP in the process of generating the zero-knowledge proof in Step 2. The protocol associated with Step 1 is illustrated in Figure 3.2 in the form of a flow diagram.

**Step 2:** After authenticating with LHP, $U_m$ sends an EHR resource access request indicating the $EHR_{id} = I$, $EHR_{obj} = O$ and actions intended to be performed on the requested resource as shown in Figure 3.3. When the request is received at the PEP of the LHP, it forwards the request to the PR in which the relevant access structure $\mathcal{T}_{I,O}$ is fetched and returned to the PEP. Then, the PEP will forward $\mathcal{T}_{I,O}$

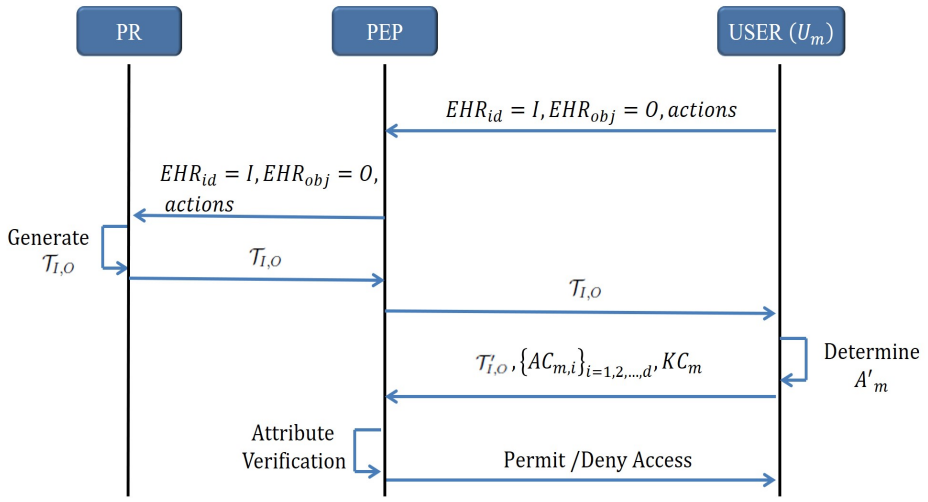



Figure 3.3: Flow diagram associated with Step 2 of the AC mechanism in Scheme 1

to $U_m$, requesting a proof that he owns a set of attributes that satisfies $\mathcal{T}_{I,O}$. Thus, based on the received $\mathcal{T}_{I,O}$, $U_m$ first determines the smallest subset of attributes $A'_m$ which satisfies $\mathcal{T}_{I,O}$, where $A'_m \subseteq A_m$. Furthermore, based on $A'_m$, $U_m$ generates the attribute commitments using the relevant attribute keys. The process of generating commitments is described below.

- Let us assume that there are $d$ attributes in $A'_m$ ascertained from $l$ AAs. Then, for each element $i$ in $A'_m$ except the last, a random value $s_i \in \mathbb{Z}_p^*$ is assigned while the last element is assigned the value of $(ls - \sum_{i=1}^{d-1} s_i)$ in which $s$ denotes the component computed by $U_m$ during Step 1.

- Afterwards, an attribute commitment for each element $i$ in $A'_m$ is computed. Suppose, $\{sk_i\}_{i=1,2...,d}$ denotes the relevant secret key set owned by $U_m$ which relates the identity of $U_m$ to the attributes in $A'_m$ whereas $\{sk_0^i\}_{i=1,2,...,l}$ refers to the set of secret key components which relates the identity of $U_m$ to the $l$ AAs which issued the $d$ attributes. According to (3.1) and (3.2),

$$sk_i = g^{r_m/t_i} \quad \text{and} \quad sk_0^i = g^{(\alpha_i - r_m)/\beta_i}. \tag{3.3}$$

  If the attribute commitments of $U_m$ are denoted by $\{AC_{m,i}\}_{i=1,2...,d}$, then according to (3.3),

$$AC_{m,i} = sk_i{}^{s_i} = g^{(r_m/t_i)s_i}. \tag{3.4}$$

- In addition to the attribute commitments, a commitment for the private keys ($\{sk_0^i\}_{i=1,2,...,l}$) received from $l$ AAs is also generated. If the private key com-

mitment is given by $KC_m$ then,

$$KC_m = \prod_{i=1}^{l} e(g^{(\alpha_i - r_m)/\beta_i}, {X_i}^s) = e(g,g)^{s\sum_{i=1}^{l}\alpha_i - l r_m s}. \tag{3.5}$$

After the commitments have computed, $U_m$ forwards a composite message to the LHP including $\mathcal{T}'_{I,O}$ which indicates the attributes that the user is intending to disclose to the LHP, attribute commitments $\{AC_{m,i}\}_{i=1,2...,d}$ and the private key commitment $KC_m$ as well as the AA information of the disclosing attributes. Then, the PEP will use the user commitments, public key components of relevant AAs to compute a zero-knowledge proof, which makes it possible to determine whether $U_m$ possesses the claiming attributes. The procedure to be followed by the PEP for the computation of the proof is given below.

- First of all, the PEP will use the attribute commitments and compute,

$$\begin{aligned}\prod_{i=1}^{d} e(g^{(r_m/t_i)s_i}, T_i) &= \prod_{i=1}^{d} e(g^{(r_m/t_i)s_i}, g^{t_i}) \\ &= \prod_{i=1}^{d} e(g,g)^{(r_m)s_i} \\ &= e(g,g)^{r_m l s}\end{aligned} \tag{3.6}$$

  where $T_i$ denotes the public attribute key of the $i^{th}$ element in $A'_m$.

- Finally, PEP will use the result in (3.6) and the private key commitment $KC_m$ of $U_m$, to check whether the condition,

$$e(g,g)^{r_m l s} \cdot KC_m \stackrel{?}{=} \prod_{i=1}^{l} e(Y_i, Z)$$

  is held or not to determine whether $U_m$ possesses an attribute set that satisfies $\mathcal{T}_{I,O}$.

### 3.3.6 Attribute Revocation Mechanism

In Scheme 1, the revocation process is handled by the AA which is responsible for the revoking attribute. The revocation process is as follows. Suppose, $AA_k$ requires to revoke the attribute $\omega$ from $U_m$. In addition, assume that the secret attribute exponent associated with the attribute $\omega$ defined by $AA_k$ is given by $t_\omega$ and the

associated public key component is given by $T_\omega = g^{t_\omega}$. First, $AA_k$ will choose a new master secret component $t'_\omega$ for the attribute $\omega$ and the corresponding public attribute key $T'_\omega = g^{t'_\omega}$ is generated and published by replacing $T_\omega$. Then, based on the new master secret exponent $t'_\omega$, new secret keys for the attribute $\omega$ are generated and sent to the users who obtained the attribute $\omega$ previously except the user to be revoked ($U_m$). Given that the public attribute key component of the attribute $\omega$ is modified, $U_m$ will not be able to use his secret key associated with the attribute $\omega$ after the execution of the revocation.

### 3.3.7 Security Analysis

In this subsection, we show that Scheme 1 is secure against attribute forgery, attribute collusion, replay attacks as well as capable of provisioning multi-session unlinkability. We use the following hardness problem and the security assumption in the analysis to follow.

**Definition 2 (Discrete Logarithm (DL) Problem):** Suppose $\mathbb{G}_0$ is a cyclic group of order $p$ with $g$ being a generator. Given $(g, g^a)$ where $a$ is selected uniformly at random from $\mathbb{Z}_p^*$, the DL problem in $\mathbb{G}_0$ is to compute $a$ [61, 62].

**Assumption 1 (DL Assumption):** Suppose $\mathbb{G}_0$ is a cyclic group of order $p$ with $g$ being a generator. Given $(g, g^a)$ there is no probabilistic polynomial-time algorithm which can compute $a \in \mathbb{Z}_p^*$ with non-negligible probability.

**Resistance against attribute forgery:** Suppose, an adversary $\mathcal{A}$ wants to forge the attribute $\omega$ managed by $AA_k$. Given the fact that the public parameter set of $AA_k$ is given by $\{X_k = g^{\beta_k}, Y_k = g^{\alpha_k}, T_\omega = g^{t_\omega}\}$ and the master secret set is given by $\{\xi, \alpha_k, \beta_k, t_\omega\}$, $\mathcal{A}$ wins the forgery if and only if $\mathcal{A}$ can generate the valid key set $\{g^{(\alpha_k - r_\mathcal{A})/\beta_k}, g^{r_\mathcal{A}/t_\omega}\}$ while only having access to the public parameter set of $AA_k$. Note that $t_\omega$ corresponds to the master secret associated with the attribute $\omega$ whereas $r_\mathcal{A}$ denotes the identifier associated with $\mathcal{A}$'s identity. For a successful construction of secret attribute keys, $\mathcal{A}$ needs to extract $t_\omega$ from $T_\omega$ which is DL hard. Hence, Scheme 1 is resistant to attribute forgery, given that the DL assumption is held.

**Resistance against attribute collusion:** We ensure the prevention of collusion attacks via infusing identity related characteristic to each attribute related secret key issued by AAs. Suppose, two users $U_1$ and $U_2$ wish to collude secret keys of two attributes $\omega_1, \omega_2$ which are owned by $U_1$ and $U_2$ respectively, to gain access to a

resource having an associated access structure $\mathcal{T} = \omega_1 \wedge \omega_2$. Further assume that $\omega_1$ is administered by $AA_1$ and $\omega_2$ is administered by $AA_2$ while $t_1, t_2$ denote the corresponding attribute secret exponents defined by the respective AA. In addition, the secret keys of $U_1$ and $U_2$ corresponding to the attributes $\omega_1$ and $\omega_2$ are given by $\{g^{r_1/t_1}, g^{(\alpha_1 - r_1)/\beta_1}\}$ and $\{g^{r_2/t_2}, g^{(\alpha_2 - r_2)/\beta_2}\}$ respectively.
According to (3.4) - (3.5), $U_1$ and $U_2$ can collaboratively generate the commitments $\{AC_i\}_{i=1,2}$ and $KC$ such that,

$$AC_1 = g^{(r_1/t_1)s_1}$$

$$AC_2 = g^{(r_2/t_2)(2s-s_1)}$$

$$KC = \prod_{i=1}^{2} e(g^{(\alpha_i - r_i)/\beta_i}, {X_i}^s) = e(g,g)^{s\sum_{i=1}^{2}\alpha_i - (r_1+r_2)s}.$$

Then, according to (3.6), the verifier computes,

$$\begin{aligned}\prod_{i=1}^{2} e(AC_i, T_i) &= \prod_{i=1}^{2} e(g^{(r_i/t_i)s_i}, g^{t_i}) \\ &= \prod_{i=1}^{2} e(g,g)^{(r_i)s_i} \\ &= e(g,g)^{r_1 s_1 + r_2(2s-s_1)}.\end{aligned}$$

In order to have a successful authorization, the computation of $KC \cdot \prod_{i=1}^{2} e(AC_i, T_i)$ must be equivalent to $e(g,g)^{s(\alpha_1+\alpha_2)}$. Therefore,

$$e(g,g)^{r_1 s_1} \cdot e(g,g)^{r_2(2s-s_1)} \cdot e(g,g)^{-(r_1+r_2)s} = 1. \tag{3.7}$$

The relation in (3.7) can only be maintained if and only if $r_1 = r_2$. Hence, it is infeasible to achieve a successful authorization via colluding attribute secret keys of more than one user. In the above analysis, we considered that the two attributes $\omega_1, \omega_2$ managed by two AAs. In the same way, it is possible to show that the case in which the collusion of attributes managed by a single AA is also will not yield a successful authorization.

**Resistance against replay attacks:** According to the AC mechanism presented in Section 3.3.5, a successful authorization requires two steps, wherein Step 1, the access requesting user authenticates the LHP while establishing a secret which is used in Step 2 of the protocol in which the user generates the attribute commitments. Suppose, $\mathcal{A}$ intercepts a set of valid attribute commitments from a user. When $\mathcal{A}$ intends to replay the intercepted commitments, first $\mathcal{A}$ needs to carry out Step 1 of the AC mechanism and establish a secret to infuse into the commitments. Due to the randomness associated with the process of establishing the secret, the secret established by $\mathcal{A}$ will not be the same as the secret associated with the intercepted commitments. Hence, the user commitments will not produce a valid zero-knowledge proof during the replay.

**Multi-session unlinkability:** If a user's two or more access sessions are unlinkable, we call the underlying AC mechanism exhibits multi-session unlinkability. In Scheme 1, we achieve this property by randomizing the attribute commitments before being disclosed to the verifier. This is evident by the expressions for attribute commitments of user $U_m$, $\{AC_{m,i}\}_{i=1,2,\ldots,d}$ in (3.4) and the private key commitment, $KC_m$ in (3.5).

$$AC_{m,i} = {sk_i}^{s_i} = g^{(r_m/t_i)s_i}$$

$$KC_m = \prod_{i=1}^{l} e(g^{(\alpha_i - r_m)/\beta_i}, {X_i}^s) = e(g,g)^{s\sum_{i=1}^{l}\alpha_i - l r_m s}$$

In each access session, $U_m$ generates the $s_i$ exponents randomly, and the secret $s$ is computed with the help of random exponents exchanged between $U_m$ and the verifier LHP, when executing Step 1 of the AC mechanism. Hence, each $AC_{m,i}$ component and $KC_m$ appear to be random elements in $\mathbb{G}_0$ and $\mathbb{G}_1$ respectively. This ensures that the verifier will not be able to link different access sessions of the same user.

## 3.4 Motivation for Scheme 2

From our analysis in Section 3.3.7, it is evident that Scheme 1 is capable of provisioning anonymous access to users with multi-session unlinkability while resisting attacks mounted via attribute forgery, attribute collusion and replay attacks. However, the complexity associated with attribute key management at the user's end is a drawback in terms of the flexibility. To be exact, if a user ascertained $d$ attributes

from an AA, the user will receive $(d + 1)$ secret keys ($d$ secret keys corresponding to the $d$ attributes and an additional key which relates the identities of the user and the attribute issuing authority). Furthermore, Scheme 1 is not equipped with a mechanism to revoke the anonymity, which might be of practical importance for resolving access disputes and making users accountable for their actions. Our main intention in Scheme 2 is to address these above stated drawbacks via proposing an ABAC scheme using attribute credentials.

A credential can be simply considered as a cryptographic container of attributes certified by a trusted issuer. Thus, a user will be able to disclose attributes embedded in the credential either fully or partially to a verifier according to the access requirements of services sought by the user [63, 64]. If a particular user's presentation of a credential over multiple sessions is unlinkable, we call such a credential as a multi-show unlinkable credential. There exist two well-known credential schemes, the Identity Mixer (Idemix) credential from IBM [65] and the Microsoft U-Prove credential [66–68]. In comparison, the U-Prove credential is more efficient compared to the Idemix, since Idemix credential uses the Camenisch-Lysyanskaya (CL) signature scheme [69] (developed based on the strong RSA assumption) which significantly increases the computational complexity [70]. However, the standard U-Prove credential does not possess the multi-show unlinkability meaning that the use of the same credential over multiple sessions can be traced and linked together. In addition to Idemix, several other multi-show unlinkable attribute credential schemes have been proposed [71–73] which are comparatively computationally more efficient than the Idemix. However, on the contrary, all these schemes lag behind the U-prove credential concerning the associated computational efficiency.

In Scheme 2, we propose a novel attribute based credential scheme influenced by the standard U-Prove credential which can enforce multi-show unlinkability. It has substantially lower end-user computational complexity in comparison to the existing credential schemes with multi-show unlinkability. Moreover, with the help of appropriate simulation results, we also show that the proposed scheme is on par with U-Prove in relation to the associated computational cost. The proposed credential scheme is also coupled with a mechanism to revoke the anonymity of credentials to counter inappropriate user behaviors via enforcing user accountability. With the help of the proposed anonymous credential scheme, we propose an efficient AC scheme conducive for a multi-domain collaborative e-health environment, which minimizes the key management overhead at the user's end in comparison to Scheme 1.

## 3.5 Proposed Scheme 2

In this section, first of all, we present the functionality of the proposed credential scheme followed by the security analysis, performance evaluation and its applicability to our considered application scenario AS 1. There are three entities associated with the credential scheme as described below.

- **Users:** Users are the entities who seek access to protected resources via providing evidence of the ownership of relevant attributes with the help of owned attribute credentials.
- **Credential Issuer ($I$):** Credential issuer issues attributes to users in the form of credentials. Issuer makes sure that the attribute requesting user is eligible for the requesting attributes before issuing the credential. Although the scheme supports multiple issuers, it is assumed that a single issuer $I$ exists for the ease of illustrations.
- **Verifier ($V$):** We define, a verifier as the entity who verifies the user credentials in order to provide access to the requested resources.

Among these entities, we require the credential issuer $I$ to be a trusted entity. Furthermore, it is necessary that the credential issuance protocol (which runs between the issuer $I$ and a user) should be carried out in a mutually authenticated channel whereas the credential disclosure and verification protocol (which runs between a user and the verifier $V$) is carried out over a one way authenticated channel meaning that the user authenticates the verifier $V$ but not vice versa.

In the following subsections (Section 3.5.1 - Section 3.5.4), we describe the functionality of the proposed credential scheme by dividing it into four phases: issuer initialization, credential issuance protocol, credential disclosure and verification protocol and enforcing user accountability.

### 3.5.1 Issuer Initialization

First, $I$ defines a cyclic group $\mathbb{G}_0$ of prime order $p$ and two cyclic groups $\mathbb{G}_1$, $\mathbb{G}_T$ of prime order $p$ with $q$ being a generator of $\mathbb{G}_1$. A bilinear map $e : \mathbb{G}_0 \times \mathbb{G}_1 \rightarrow \mathbb{G}_T$ is also generated along with a secure hash function $H : \{0,1\}^* \rightarrow \mathbb{Z}_p^*$. Furthermore, $I$ also generates a random secret exponent $\alpha \in \mathbb{Z}_p^*$. Suppose, $A_I$ denotes the set of attribute indices associated with the attributes managed by $I$. Then, $I$ generates exponents $\{\{a_{1j}\}_{j=1,2,...,l_1}, a_i\}_{i=2,3,...,|A_I|}$, in which $a_{1j}, a_i \in \mathbb{Z}_p^*$. The set of values

$\{a_{1j}\}_{j=1,2,\ldots,l_1}$ defines the credential expiration values meaning that each credential issued by $I$ includes one expiration value as the first attribute ($a_1$). Finally, $I$ generates a set of generators $\{g_0, g_i\}_{i \in A_I}$ of $\mathbb{G}_0$ and publishes its public tuple,

$$PK_I = \{q, g_0, \{g_i\}_{i \in A_I}, \{\{a_{1j}\}_{j=1,2,\ldots,l_1}, a_i\}_{i=2,3,\ldots,|A_I|}, q^{\alpha}, e, H\}$$

and keeps $SK_I = \alpha$ as its secret key.

### 3.5.2 Credential Issuance Protocol

Suppose, the user $U_m$ wants to acquire a credential corresponding to the set of attribute indices $A_m$, where $A_m \subseteq A_I$ and there exists a secure channel between $U_m$ and $I$. $U_m$ starts the credential issuance by sending a credential request to $I$ indicating the intended attributes along with evidence that $U_m$ is eligible for the requested attributes. First, $I$ will verify whether $U_m$ is eligible for the requested attributes, if validated, $I$ will issue the credential as described below. The issuance protocol is also illustrated with the help of a flow diagram in Figure 3.4.

- $I$ generates an aggregated attribute component $M$ as mentioned below and forwards it to $U_m$ along with the public tuple $PK_I$.

  $$M = g_0 \prod_{i \in A_m} {g_i}^{a_i}$$

  Note that in $M$, the first attribute exponent $a_1$ corresponds to the credential expiration information.

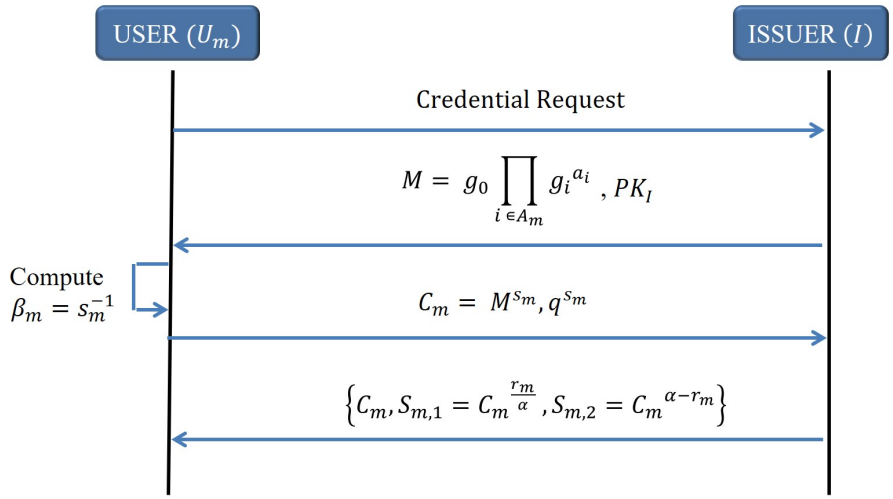



Figure 3.4: Flow diagram representing the credential issuance protocol

- $U_m$ generates a random exponent $s_m \in \mathbb{Z}_p^*$ and computes the credential secret key $\beta_m = {s_m}^{-1}$. Then, with the help of $s_m$ and $M$, $U_m$ computes the credential public key $C_m$ such that,

$$C_m = M^{s_m}$$

and forwards it to $I$ along with $q^{s_m}$.

- Then, $I$ first checks whether the condition,

$$e(C_m, q) \stackrel{?}{=} e(M, q^{s_m}) \text{ is held or not.}$$

If successful, $I$ issues the credential $\{C_m, S_{m,1}, S_{m,2}\}$ to $U_m$. $S_{m,1}, S_{m,2}$ are signature components of $C_m$, generated as explained below. $I$ generates a random identifier $r_m \in \mathbb{Z}_p^*$ and computes $S_{m,1}, S_{m,2}$ such that,

$$S_{m,1} = {C_m}^{\frac{r_m}{\alpha}}$$
$$S_{m,2} = {C_m}^{\alpha - r_m}.$$

- Finally, $U_m$ securely stores $\{C_m, S_{m,1}, S_{m,2}\}$ along with the credential secret key $\beta_m$.

### 3.5.3 Credential Disclosure and Verification Protocol

Let us assume that, $U_m$ wants to provide evidence to the verifier $V$ that he owns the attribute set $\{a_i\}_{i \in A_m^R}$, in which $A_m^R$ is the set of indices associated with the disclosing attribute set, where $A_m^R \subseteq A_m$. The protocol involves two phases. In Phase 1, the validity of the credential is examined (whether or not it is issued by $I$, via signature verification) whereas in Phase 2, a proof is generated to determine whether the user owns the disclosing attribute set.

**Phase 1:** $U_m$ selects a randomization exponent $K \in \mathbb{Z}_p^*$ and computes the randomized credential public key $C'_m$ and the randomized credential signature components $S'_{m,1}, S'_{m,2}$ such that,

$$C'_m = {C_m}^K \tag{3.8}$$
$${S'_{m,1}} = {S_{m,1}}^K = {C_m}^{\frac{r_m K}{\alpha}} \tag{3.9}$$
$${S'_{m,2}} = {S_{m,2}}^K = {C_m}^{(\alpha - r_m)K}. \tag{3.10}$$

This randomization allows $U_m$ to be unlinkable in multiple sessions (by simply selecting different randomization exponents). Then, $U_m$ sends $C'_m, S'_{m,1}, S'_{m,2}$ along with $PK_I$ to $V$ for validation. After receiving the user commitments, $V$ first computes,

$$e(S'_{m,1}, q^\alpha) = e({C_m}^{\frac{r_m K}{\alpha}}, q^\alpha) = e(C_m, q)^{r_m K} \tag{3.11}$$

$$e(S'_{m,2}, q) = e({C_m}^{(\alpha - r_m)K}, q) = e(C_m, q)^{(\alpha - r_m)K}. \tag{3.12}$$

With the computation results in (3.11) and (3.12), $V$ checks whether the condition,

$$e(C'_m, q^\alpha) \stackrel{?}{=} e(S'_{m,1}, q^\alpha) \cdot e(S'_{m,2}, q) \tag{3.13}$$

is held, and if so, $V$ determines that the committed credential is a credential issued by $I$.

**Phase 2:** If Phase 1 is successful, $V$ requests $U_m$ to disclose the required attributes as well as a proof of knowledge of the credential private key using a nonce $N_0$. Note that $U_m$ must disclose $a_1$ to $V$, since it allows $V$ to determine the disclosed credential is expired or not. First, $U_m$ generates a random exponent $\bar{s}_m \in \mathbb{Z}_p^*$ and a set of random exponents $\bar{a}_i \in \mathbb{Z}_p^*, \forall\, i \in A_m/A_m^R$. Then, $U_m$ computes,

$$L = {C'_m}^{\bar{s}_m} \cdot \prod_{i \in A_m/A_m^R} g_i^{\bar{a}_i} \tag{3.14}$$

$$D' = H(L) \quad \text{and} \quad D = H(D', N_0) \tag{3.15}$$

Thereafter, with the use of the credential private key $\beta_m$, $\bar{s}_m$ and $D$, $U_m$ computes,

$$\hat{s}_m = \bar{s}_m + D \cdot \beta_m \tag{3.16}$$

and for each attribute index $i \in A_m/A_m^R$,

$$\hat{a}_i = \bar{a}_i - D \cdot a_i. \tag{3.17}$$

Furthermore, $U_m$ computes $L'$ such that,

$$L' = L \cdot M^{K \cdot D - D} \tag{3.18}$$

where $M = g_0 \prod_{i \in A_m} {g_i}^{a_i}$ and $K$ denotes the randomization exponent. Then,

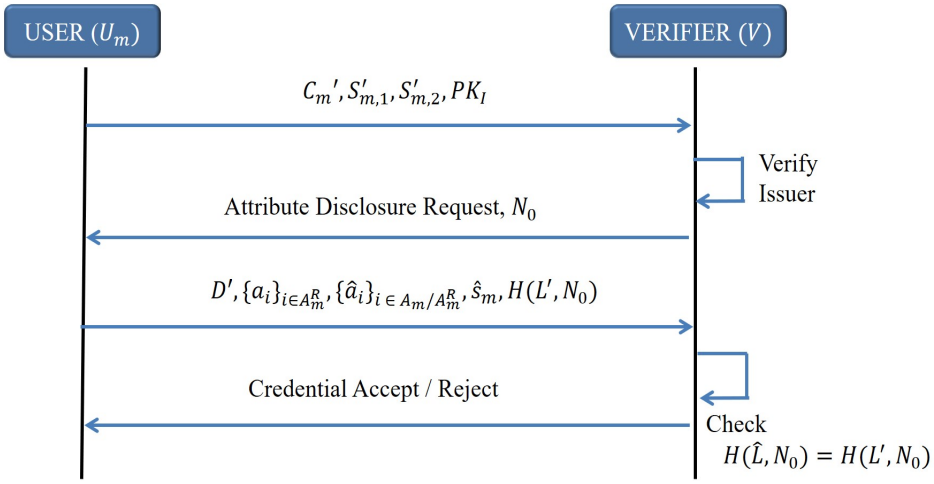



Figure 3.5: Flow diagram representing the credential disclosure and verification protocol

$U_m$ deduces $H(L', N_0)$ using the nonce $N_0$ issued by $V$. Thereafter, $U_m$ sends $D', \{a_i\}_{i \in A_m^R}, \{\hat{a}_i\}_{i \in A_m/A_m^R}, \hat{s}_m, H(L', N_0)$ to $V$ for verification. Using user commitments, $V$ first examines the attribute value $a_1$ to determine the validity of the credential and if it is not expired, $V$ computes $\hat{L}$ where,

$$\hat{L} = g_0^{-D} \cdot C_m'^{\,\hat{s}_m} \cdot \prod_{i \in A_m^R} g_i^{-D \cdot a_i} \cdot \prod_{i \in A_m/A_m^R} g_i^{\hat{a}_i}. \tag{3.19}$$

Finally, $V$ checks whether,

$$H(\hat{L}, N_0) \stackrel{?}{=} H(L', N_0)$$

to determine the validity of the $U_m$'s selective disclosure proof. The verification succeeds, if and only if $V$ can successfully reconstruct the user commitment $H(L', N_0)$ with the help of the disclosed attributes. We can demonstrate the correctness of the reconstruction as follows. From (3.19),

$$\begin{aligned}
\hat{L} &= g_0^{-D} \cdot C_m'^{\,\hat{s}_m} \cdot \prod_{i \in A_m^R} g_i^{-D \cdot a_i} \cdot \prod_{i \in A_m/A_m^R} g_i^{\hat{a}_i} \\
&= g_0^{-D} \cdot C_m'^{\,\bar{s}_m} \cdot C_m'^{\,D \cdot \beta_m} \cdot \prod_{i \in A_m^R} g_i^{-D \cdot a_i} \cdot \prod_{i \in A_m/A_m^R} g_i^{\bar{a}_i} \cdot \prod_{i \in A_m/A_m^R} g_i^{-D \cdot a_i} \\
&= g_0^{-D} \cdot C_m'^{\,\bar{s}_m} \cdot C_m'^{\,D \cdot \beta_m} \cdot \prod_{i \in A_m} g_i^{-D \cdot a_i} \cdot \prod_{i \in A_m/A_m^R} g_i^{\bar{a}_i} \\
&= C_m'^{\,\bar{s}_m} \cdot C_m'^{\,D \cdot \beta_m} \cdot M^{-D} \cdot \prod_{i \in A_m/A_m^R} g_i^{\bar{a}_i} \\
&= C_m'^{\,\bar{s}_m} \cdot (M^{K \cdot s_m})^{D \cdot \beta_m} \cdot M^{-D} \cdot \prod_{i \in A_m/A_m^R} g_i^{\bar{a}_i}
\end{aligned}$$

$$= {C'_m}^{\bar{s}_m} \cdot M^{K \cdot D - D} \cdot \prod_{i \in A_m / A_m^R} g_i^{\bar{a}_i}$$

$$= L'.$$

### 3.5.4 Enforcing User Accountability

Although anonymous credentials provide anonymous access to the users, it is important to have a mechanism to revoke the anonymity of a utilized credential and reveal the identity of the respective owner especially in the case of an access dispute and making users accountable for their actions. The proposed credential scheme allows $V$ to revoke the anonymity of a utilized credential with the help of $I$ through presenting the protocol transcript relevant for the credential disclosure and verification session. The mechanism functions as described below.

Suppose, $U_m$ obtains the credential $\{C_m, S_{m,1} = {C_m}^{\frac{r_m}{\alpha}}, S_{m,2} = {C_m}^{\alpha - r_m}\}$ by running the credential issuance protocol with $I$. As mentioned in Section 3.5.2, credential issuance protocol should run on a mutually authenticated channel and let us assume that it is simply achieved through a method using public key cryptography. To facilitate accountability, $I$ keeps track of a table which includes the PKI certificate of the credential receiver ($U_m$), public key of the issued credential ($C_m$), random identifier used to construct the signature components of the credential public key ($r_m$), credential expiration information and other attributes embedded in the issued credential.

Now, suppose $U_m$ is engaged in the credential disclosure and verification protocol with $V$ and forwarded the randomized credential $\{{C_m}^K, {S_{m,1}}^K, {S_{m,2}}^K\}$ for verification. Further assume that $U_m$ forwarded the proof $P$,

$$P = \{D', \{a_i\}_{i \in A_m^R}, \{\hat{a}_i\}_{i \in A_m / A_m^R}, \hat{s}_m, H(L', N_0)\}$$

to disclose the set of attributes $A_m^R$ during Phase 2 of the protocol. If $V$ wants to revoke the anonymity of this credential, $V$ must send a revocation request to $I$ including the randomized credential $\{{C_m}^K, {S_{m,1}}^K, {S_{m,2}}^K\}$ and the proof $P$ along with a reasoning for revoking the anonymity. If $I$ decides to revoke the anonymity, $I$ proceeds as follows. First $I$ generates the component $A$ using ${S_{m,1}}^K$ and the private key of $I$, $\alpha$ such that,

$$A = {S_{m,1}}^{K \cdot \alpha} = {C_m}^{\frac{r_m}{\alpha} K \cdot \alpha} = {C_m}^{r_m \cdot K}$$

and let $B = {C_m}^K$. Then, $I$ extracts $\{a_i\}_{i \in A_m^R}$ from $P$ and determines the expiration

information of the credential and other disclosed attributes. Afterwards, $I$ traverses through the entries in the table to figure out the possible set of rows in the table that match with the expiration and other attributes associated with the credential to be revoked. Suppose, $d$ number of entries from the table have been selected (matching the requirements) and the set of random identifiers used to generate the respective credential signatures are denoted by $\{r_i\}_{i=1,2,...,d}$. Then, for each of the selected entries, $I$ checks whether, $A \stackrel{?}{=} B^{r_i}$ and the entry which satisfies the aforementioned condition relates to the credential to be revoked. Theoretically (although rarely in practice), it may also be possible to have more than one entry satisfying the aforementioned condition due to collisions. In such a scenario, $I$ can learn the possible suspects and request them to prove their innocence (i.e. revealing the credential public key, revealing undisclosed attributes) and thereby discover the owner of the credential in question.

### 3.5.5 Security Analysis

In this subsection, our intention is to show that the proposed credential scheme exhibits resistance against attribute forgery (via credential unforgeability), resistance against replay attacks as well as multi-show unlinkability. First of all, we introduce the following hardness problem and the security assumption which we use in the analysis to follow.

**Definition 3 (Decisional Diffie-Hellman (DDH) Problem):** Suppose $\mathbb{G}_1$ is a cyclic group of order $p$ with $q$ being a generator. Given that $a, b, z \in \mathbb{Z}_p^*$ are randomly chosen, the DDH problem in $\mathbb{G}_1$ is to distinguish the tuple $(q, q^a, q^b, q^{ab})$ from the tuple $(q, q^a, q^b, q^z)$ [74].

**Assumption 2 (DDH Assumption):** Suppose $\mathbb{G}_1$ is a cyclic group of order $p$ with $q$ being a generator. Given that $a, b, z \in \mathbb{Z}_p^*$ are randomly chosen, there is no polynomial-time adversary that can distinguish the tuple $(q, q^a, q^b, q^{ab})$ from the tuple $(q, q^a, q^b, q^z)$ with non-negligible probability.

**Credential unforgeability:** According to our credential issuance protocol, a credential is composed of a credential public key (which embeds the attributes) and its signature generated using the issuer's master secret. Thus, we define credential unforgeability as the inability of an adversary to construct a valid credential signature over a credential public key that embeds a given set of attributes while only having access to the public tuple of the issuer $I$.

To prove credential unforgeability, first of all, we assume that there exist an

adversary $\mathcal{A}$ who is capable of forging a credential with an advantage of $\epsilon$ (i.e. $\epsilon = |Pr[\text{forgery}] - 1/2|$, where $Pr[\text{forgery}]$ denotes $\mathcal{A}$'s probability of a successful credential forgery). Then, we define the adversary model for credential unforgeability with the help of an indistinguishability game between a challenger $\mathcal{C}$ and the adversary $\mathcal{A}$. With this game, we claim that, if the adversary $\mathcal{A}$ has an advantage of $\epsilon$ in forging a credential, then $\mathcal{A}$ has an advantage of $\epsilon$ in winning the indistinguishability game. Finally, we show that, if such an adversary exists, it is possible to use this adversary to build a simulator that can solve the DDH hardness problem with an advantage of $\epsilon/2$.

The indistinguishability game between the challenger $\mathcal{C}$ and the adversary $\mathcal{A}$ runs as follows.

- **Setup:** The challenger $\mathcal{C}$ acts as the issuer $I$ and generates the master secret $MK_{\mathcal{C}}$ and the public tuple $PK_{\mathcal{C}}$. In addition, $\mathcal{C}$ generates two credentials $\{C_0, \sigma_0 = \{S_{0,1}, S_{0,2}\}\}$ and $\{C_1, \sigma_1 = \{S_{1,1}, S_{1,2}\}\}$ which embed the sets of attributes $\omega_0$ and $\omega_1$ respectively. Note that $\sigma_0$ and $\sigma_1$ represent the signatures of credential public keys $C_0, C_1$ generated using the master secret $MK_{\mathcal{C}}$. $\mathcal{C}$ also generates $H_0, H_1$ such that, $H_0 = H(\sigma_0)$ and $H_1 = H(\sigma_1)$ using a secure hash function $H$.

- **Phase 1:** The challenger $\mathcal{C}$ sends the two credential public keys $C_0$ and $C_1$ along with the public tuple $PK_{\mathcal{C}}$ to the adversary $\mathcal{A}$.

- **Challenge Phase:** $\mathcal{C}$ picks a random bit $v \in \{0, 1\}$ and forwards the commitment $H_v$ to the adversary $\mathcal{A}$ and challenges $\mathcal{A}$ to figure out whether the signature hidden in $H_v$ corresponds to $C_0$ or $C_1$.

- **Guess:** The adversary $\mathcal{A}$ outputs a guess $v' \in \{0, 1\}$.

Given that $H$ is a secure hash function, the only way that $\mathcal{A}$ can solve the above stated challenge is through forging the signatures of $C_0$ or / and $C_1$ and computing the respective hash values. We assumed that the adversary $\mathcal{A}$ has an advantage of $\epsilon$ in forging a credential. Hence, $\mathcal{A}$ will also have an advantage of $\epsilon$ in winning the aforementioned indistinguishability game. Furthermore, we can define the advantage of the adversary $\mathcal{A}$ in the indistinguishability game as,

$$\epsilon = |Pr[v' = v] - \frac{1}{2}|. \tag{3.20}$$

Now, we show that, if there exists an adversary $\mathcal{A}$ who can win the above mentioned game with an advantage $\epsilon$, it is possible to use this adversary $\mathcal{A}$ to build a simulator $\mathcal{S}$ that can solve the DDH hardness problem with an advantage of $\epsilon/2$. The simulation proceeds as follows.

**Setup**: The challenger $\mathcal{C}$ acts as the issuer and generates the cyclic group $\mathbb{G}_0$ of prime order $p$ and two cyclic groups $\mathbb{G}_1$, $\mathbb{G}_T$ of prime order $p$ with $q$ being a generator of $\mathbb{G}_1$. A bilinear map $e : \mathbb{G}_0 \times \mathbb{G}_1 \rightarrow \mathbb{G}_T$ is also generated along with a secure hash function $H$. Suppose, $A_{\mathcal{C}}$ denotes the set of attribute indices associated with the attributes managed by the challenger $\mathcal{C}$. Then, $\mathcal{C}$ generates the set of generators $\{g_0, g_i\}_{i \in A_{\mathcal{C}}}$ of $\mathbb{G}_0$ as well as the public attribute exponents $\{a_i\}_{i \in A_{\mathcal{C}}}$, where $a_i \in \mathbb{Z}_p^*$. In addition, $\mathcal{C}$ also generates the random secret exponents $a, b, z \in \mathbb{Z}_p^*$ and computes the master secret $MK_{\mathcal{C}}$ such that $MK_{\mathcal{C}} = ab$.

Then, $\mathcal{C}$ randomly selects two sets of attribute indices $A_0, A_1 \subseteq A_{\mathcal{C}}$ and generates the respective credentials $\{C_0, \sigma_0\}$ and $\{C_1, \sigma_1\}$ such that,

$$C_0 = (g_0 \prod_{i \in A_0} {g_i}^{a_i})^s \quad \text{and} \quad \sigma_0 = ({C_0}^{\frac{r}{ab}}, {C_0}^{ab-r})$$

$$C_1 = (g_0 \prod_{i \in A_1} {g_i}^{a_i})^s \quad \text{and} \quad \sigma_1 = ({C_1}^{\frac{r}{ab}}, {C_1}^{ab-r})$$

for some $r, s \in \mathbb{Z}_p^*$. The challenger $\mathcal{C}$ also generates $H_0, H_1$ such that $H_0 = H(\sigma_0)$ and $H_1 = H(\sigma_1)$. To complete this phase, the challenger $\mathcal{C}$ feds $\{C_0, H_0\}$, $\{C_1, H_1\}$ and the DDH instance $(q, q^a, q^b, R_\delta)$ in which $R_\delta$ is set through flipping a fair coin $\delta$ where,

$$R_\delta = \begin{cases} q^{ab} & \text{if } \delta = 0 \\ q^z & \text{otherwise} \end{cases}$$

along with the public tuple $PK_{\mathcal{C}} = \{q, g_0, \{g_i\}_{i \in A_{\mathcal{C}}}, \{a_i\}_{i \in A_{\mathcal{C}}}, R_\delta, e, H\}$ to the simulator $\mathcal{S}$.

**Phase 1:** The simulator $\mathcal{S}$ sends the two credential public keys $C_0$, $C_1$ and the public tuple $PK_{\mathcal{C}}$ to the adversary $\mathcal{A}$.

**Challenge Phase:** The simulator $\mathcal{S}$ acts as the challenger for the adversary $\mathcal{A}$ and it flips a fair binary coin $v$ and forwards $H_v$ to $\mathcal{A}$.

**Guess:** The adversary $\mathcal{A}$ submits a guess $v' \in \{0, 1\}$. If $v' = v$, the simulator $\mathcal{S}$ will guess that $\delta' = 0$ and outputs a 0 indicating that $R_\delta = q^{ab}$. Otherwise, the

simulator $\mathcal{S}$ will guess that $\delta' = 1$ and outputs a 1 indicating $R_\delta = q^z$.

In the case where $\delta = 0$ ($R_\delta = q^{ab}$), $\mathcal{A}$ has an advantage of $\epsilon$ in winning the game. Hence, according to (3.20),

$$Pr[v' = v | R_\delta = q^{ab}] = \frac{1}{2} + \epsilon.$$

Since, the simulator $\mathcal{S}$ guesses that $\delta' = 0$ when $v' = v$ we have,

$$Pr[\delta' = \delta | \delta = 0] = \frac{1}{2} + \epsilon. \tag{3.21}$$

When $\delta = 1$ ($R_\delta = q^z$), the adversary $\mathcal{A}$ will not have any advantage in the game, since $R_\delta$ embedded in $PK_\mathcal{C}$ is just a random value (i.e. $R_\delta$ does not associate with $MK_\mathcal{C} = ab$). Therefore,

$$Pr[v' \neq v | R_\delta = q^z] = \frac{1}{2}$$

Given that the simulator $\mathcal{S}$ guesses $\delta' = 1$ when $v' \neq v$ it is evident that,

$$Pr[\delta' = \delta | \delta = 1] = \frac{1}{2}. \tag{3.22}$$

Therefore, according to (3.21) and (3.22), the total advantage of the simulator $\mathcal{S}$ to solve the DDH problem is given by,

$$\frac{1}{2} Pr[\delta' = \delta | \delta = 0] + \frac{1}{2} Pr[\delta' = \delta | \delta = 1] - \frac{1}{2} = \frac{\epsilon}{2}.$$

This proves that, if there exists an adversary who can forge a credential with an advantage of $\epsilon$, it is possible to use this adversary to build a simulator that can solve the DDH problem with an advantage of $\epsilon/2$. Hence, if the DDH assumption holds, the proposed credential scheme in Scheme 2 is resistant against credential forgery.

**Resistance against replay attacks:** Let us consider the following scenario. Suppose, $U_m$ owns a credential $\{C_m, S_{m,1}, S_{m,2}\}$ and its private key is denoted with $\beta_m$. During a disclosure session with $V$, $U_m$ sends the randomized credential $\{C'_m, S'_{m,1}, S'_{m,2}\}$. If the credential is verified (according to Phase 1 of credential disclosure and verification protocol), $V$ requests to disclose the attributes using the nonce $N_0$. For a disclosing set of attributes $A_m^R$, $U_m$ generates a proof $P$,

$$P = \{D', \{a_i\}_{i \in A_m^R}, \{\hat{a}_i\}_{i \in A_m / A_m^R}, \hat{s}_m, H(L', N_0)\}$$

and forwards it to $V$ where it is evaluated according to Phase 2 validation given in Section 3.5.3. Now, assume that an adversary $\mathcal{A}$ intercepts both $\{C'_m, S'_{m,1}, S'_{m,2}\}$ and $P$ and tries to interact with $V$ at a later time to show the ownership of the set of attributes $A^R_m$. First, $\mathcal{A}$ replays the randomized credential $\{C'_m, S'_{m,1}, S'_{m,2}\}$ to $V$ and the Phase 1 validation will be successful according to Section 3.5.3. Then, $V$ challenges $\mathcal{A}$ to produce a proof using a new nonce $N_1$. According to (3.15) - (3.16), $\mathcal{A}$ will not be able to generate a valid proof since $\mathcal{A}$ has no knowledge of the credential private key $\beta_m$ and $H$ is a secure hash function. Hence, $\mathcal{A}$'s attempt at replaying of intercepted attribute disclosure proofs will not be successful.

**Multi-show unlinkability:** For a credential to be multi-show unlinkable, any entity should not be able to determine whether two executions of a credential disclosure protocol involved the same credential or two different credentials. In the proposed credential scheme, we achieve this by randomizing the credential (along with its signature) before being disclosed to a verifier. Suppose, $U_m$ owns a credential $C$ such that,

$$C = \{C_m, {C_m}^{\frac{r_m}{\alpha}}, {C_m}^{\alpha - r_m}\}$$

where $C_m = (g_0 \prod_{i \in A_m} {g_i}^{a_i})^{s_m}$. Now, consider two disclosure sessions of the credential $C$. If the randomization exponents corresponding to the two disclosure sessions are given by $k_1, k_2 \in \mathbb{Z}^*_p$, then the randomized credentials disclosed in the two sessions can be denoted as follows.

$$C_{m,1} = \{{C_m}^{k_1}, {C_m}^{\frac{r_m k_1}{\alpha}}, {C_m}^{(\alpha - r_m)k_1}\}$$
$$C_{m,2} = \{{C_m}^{k_2}, {C_m}^{\frac{r_m k_2}{\alpha}}, {C_m}^{(\alpha - r_m)k_2}\}$$

In the first disclosure session, $V$ checks whether the condition,

$$e({C_m}^{k_1}, q^\alpha) \stackrel{?}{=} e({C_m}^{\frac{r_m k_1}{\alpha}}, q^\alpha) \cdot e({C_m}^{(\alpha - r_m)k_1}, q)$$

is held, where as for the session two, $V$ checks the following relation,

$$e({C_m}^{k_2}, q^\alpha) \stackrel{?}{=} e({C_m}^{\frac{r_m k_2}{\alpha}}, q^\alpha) \cdot e({C_m}^{(\alpha - r_m)k_2}, q)$$

is maintained or not. Given the fact that the two computation results, $e({C_m}^{k_1}, q^\alpha)$ and $e({C_m}^{k_2}, q^\alpha)$ appears to be two random elements in $G_T$, $V$ will not be able to link the two disclosure sessions.

Table 3.1: Comparison of end-user computational complexity associated with an attribute disclosure session of the proposed credential scheme with existing credential schemes

| Scheme | $G_{EC}$ | $G_T$ | $e$ | $G_{RSA}$ | Unlinkable |
|---|---|---|---|---|---|
| Proposed Scheme | $h_0 + 5$ | 0 | 0 | 0 | Yes |
| [73] | $n_0 + h_0 + 9$ | 0 | 0 | 0 | Yes |
| [71] | $h_0 + 7$ | 0 | 0 | 0 | Yes |
| [72] | $2n_0 + 3$ | $h_0 + 2$ | $n_0 + 3$ | 0 | Yes |
| Idemix | 0 | 0 | 0 | $h_0 + 3$ | Yes |
| U-Prove | $h_0 + 1$ | 0 | 0 | 0 | No |

### 3.5.6 Performance Evaluation

In this subsection, we provide evidence for the performance of the proposed credential scheme in terms of the end-user computational complexity as well as the computational cost associated with the credential issuance and disclosure protocols.

To compare the end-user computational complexity of the proposed credential scheme with the existing credential schemes in literature, we have tabulated the exponentiation and pairing count (associated with end-user computations) during disclosure of a credential to a verifier in Table 3.1. The columns $G_{EC}$, $G_T$ and $G_{RSA}$ show the number of exponentiations in elliptic curves, the target group of a bilinear pairing, and RSA groups respectively, while the column labeled $e$ counts the number of pairings the user has to compute during a credential disclosure session. Furthermore, $n_0$ denotes the number of attributes in a credential whereas $h_0$ denotes the number of undisclosed attributes in a disclosure session.

According to Table 3.1, it is evident that U-Prove has the lowest end-user computational complexity. However, it does not provide multi-show unlikability. If we consider the credential schemes that exhibit multi-show unlinkability, it is evident that the proposed scheme has the least end-user computational complexity. Note that the exponentiation count in Idemix is lower than the exponentiation count in the proposed scheme. However, given that the exponentiation operations in RSA groups are significantly expensive than the exponentiation operations in elliptic curves, Idemix induces higher computational cost to the end-user compared to our solution. This fact is supported by the following computational cost comparison of the proposed scheme with Idemix and U-Prove.

To compare the computational cost of the proposed scheme with the well-known credential schemes of Idemix and U-Prove, we conducted simulations to determine the associated computational cost (in terms of the computation time) for credential issuance and credential verification. The simulations were run on a Core i5, $2.5\,GHz$ PC with $8\,GB$ of RAM. In order to generate the necessary cyclic groups

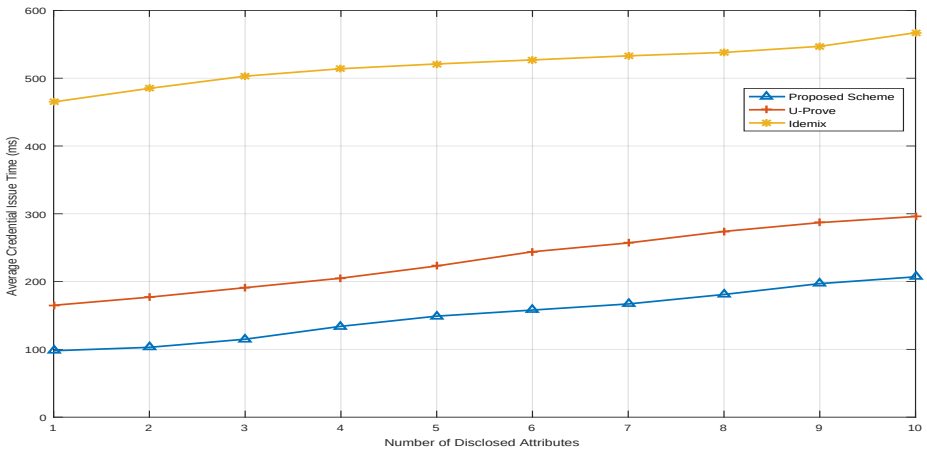
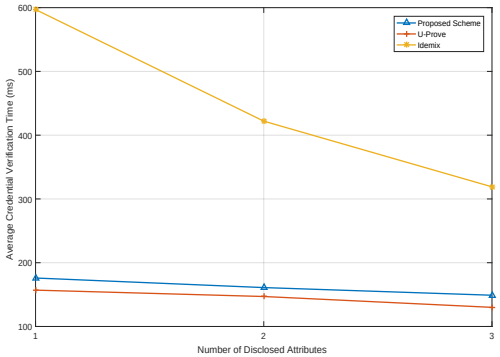
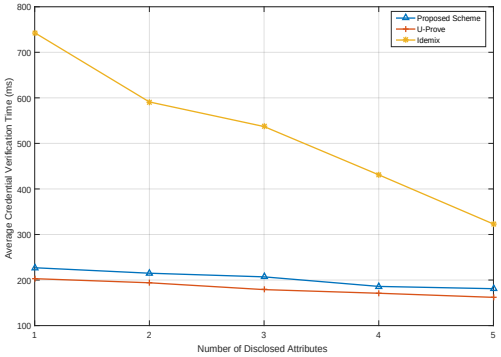


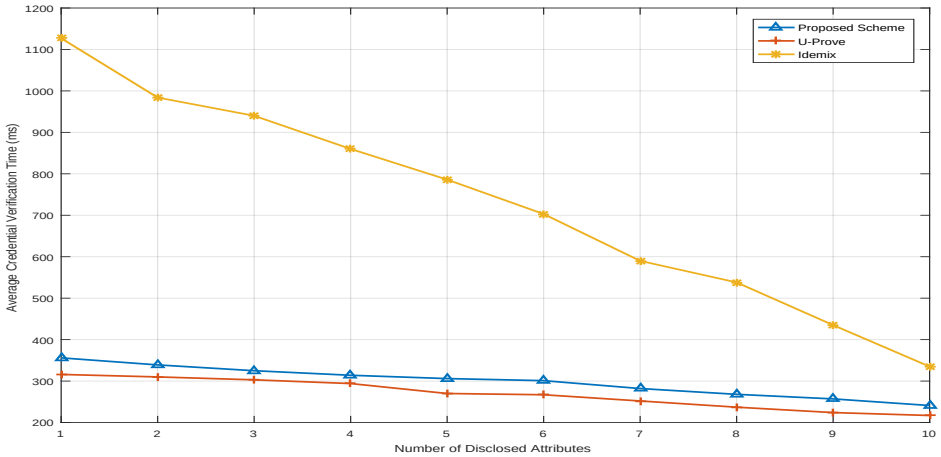

Figure 3.6: Average computational cost for credential issuance with the number of attributes embedded in the credential




(a) Case 1: 3 attributes in the credential.




(b) Case 2: 5 attributes in the credential.




(c) Case 3: 10 attributes in the credential.

Figure 3.7: Average computational cost for credential disclosure with the number of attributes disclosed

for U-Prove and the proposed scheme, the elliptic curve $y^2 = x^3 + x$ over a $512$ -bit finite field having a group order of $160$ bits was used. This parameter setting was selected by considering the fact that it can generate keys having the equivalence security of $1024$-bit RSA keys [75]. For simulating Idemix, a modulus size of $1024$ bits was used to make sure that the results are comparable with the simulation results of U-Prove and the proposed scheme.

Figure 3.6 shows the variation of average computational cost for credential issuance with the number of attributes in a credential, for the considered three credential schemes. From the results, we can see that the proposed scheme has the lowest computational cost for credential issuance whereas it is also evident that the number of attributes in the credential has a minimal effect on the associated computational cost.

In Figure 3.7, we have illustrated the variation of computational cost associated with the number of disclosed attributes from a credential during an authorization session. We considered three scenarios denoted with Case 1 - Case 3 which specify the number of attributes stored in each of the credential. The graphs show that the computational cost of the proposed scheme for disclosure of attributes is marginally higher than the U-Prove scheme, however significantly lower than the computational cost associated with Idemix. It is important to note that we did our implementations using simple Java pairing based cryptography (JPBC) library, and it would be possible to obtain better performance results by using high-speed BN-curve and pairing implementations.

### 3.5.7 Applying the Credential Scheme for AS 1

We wish to recall the system model given in Figure 2.2, which corresponds to AS 1. When adopting the proposed credential scheme to AS 1, LHP and other AAs which manage disjoint sets of attributes are functioning as trusted credential issuers while LHP functions as the verifier. Hence, respective users can obtain attribute credentials from the relevant AA, while engaging in the credential issuance protocol presented in Section 3.5.2. Then, when a user needs to access a specific $EHR_{obj}$ of a patient, the user initiates an EHR access request indicating the $EHR_{id}$, $EHR_{obj}$ and the actions intended to be performed on the requested $EHR_{obj}$ as same as in Scheme 1. Based on the request, the PEP of LHP fetches the relevant access structure $\mathcal{T}$, to be satisfied by the user (to gain access to the resource) and sends it back to the user. Then, according to the specified access requirement, the user commits one or more attribute credentials while disclosing the minimal set of attributes that is sufficient to satisfy $\mathcal{T}$. Finally, PEP verifies each committed credential using the

credential verification protocol presented in Section 3.5.3 to determine whether the user owns a valid set of attributes that enable the access to the requested resource.

## 3.6 Chapter Summary

In this chapter, we proposed two anonymous ABAC schemes compatible with the EHR sharing scenario specified in AS 1. Scheme 1 enables patient privacy via restricting EHR access to the users who satisfy the EHR access requirements along with the capabilities for withstanding attacks mounted via attribute forgery, attribute collusion and replay attacks. Furthermore, it is composed of an attribute revocation mechanism to revoke attributes from users when necessary. Scheme 1 enforces user privacy through the property of selective disclosure of attributes and allowing users to stay anonymous as well as unlinkable among multiple access sessions. The two main drawbacks associated with Scheme 1 is that it induces higher end-user key management overhead given that a user needs to store $(d + 1)$ secret keys when $d$ attributed are ascertained from an AA and the inability to revoke the anonymity of users to enforce user accountability to counter possible access disputes.

We have addressed these two issues in Scheme 2 via proposing an anonymous credential scheme which has a lower end-user computational complexity than the existing credential schemes with multi-show unlinkability. With the proposed attribute credentials based approach, the number of secret keys that a user has to manage will be reduced in comparison to Scheme 1, since a user needs to store only one secret key for any number of attributes ascertained from an AA. To counter the issue of access disputes, Scheme 2 is equipped with a trapdoor mechanism to revoke the anonymity of credentials which is achieved through allowing the verifier LHP to interact with the AA that issued the credential. Scheme 2 achieves revocation of attributes through the credential expiration attribute embedded in an issued credential. This is a simple mechanism, unlike the revocation mechanism in Scheme 1 (which requires issuing of new secret keys for all the other users who have ascertained the revoking attribute). However, given that a credential cannot be revoked before its expiry, it is of paramount importance to issue credentials with appropriate expiration and allowing users to obtain new credentials after the expiration of the current credential. One concern we have with Scheme 2 is that there is a possibility of colluding attributes embedded in two credentials owned by two users, since the verifier will not be able to recognize on its own that these two credentials are not owned by the same entity. But we can achieve this by allowing the verifier to communicate with the AAs which issued the credentials to get an acknowledgment

whether the credentials are owned by the same entity. However, this is not an efficient mechanism, since the verifier needs to communicate with AAs during every access session. We intend to address this issue in our future work.

# Chapter 4

# Delegatable ABAC Schemes for AS 1

*In this chapter, we present the proposed delegatable ABAC schemes conducive to the access demands associated with AS 1. First, we introduce the notion of controlled access delegation, and it is followed by a summary of the most prominent research efforts which propose delegatable AC mechanisms. Then, the three proposed schemes: Scheme 3, Scheme 4 and Scheme 5 are presented in detail while bridging the schemes with motivations. Scheme 3 is based on our work in Paper 5 whereas Scheme 4 and Scheme 5 are based upon the work in Paper 6 and Paper 4 respectively. The chapter is concluded with a summary.*

## 4.1 Controlled Access Delegation

The ability of a user to delegate access is quite vital to achieving timely and flexible sharing of EHRs of patients among the intended parties. For instance, consider the following scenario. Suppose, Alice is a patient of hospital A and her EHR stored in hospital A is associated with an attribute access structure $\mathcal{T} = (Cardiologist \wedge Hospital\_A)$ permitting cardiologists in hospital A to access Alice's EHR. After a recent consultation session, Dr. Bob who is a cardiologist at hospital A finds some anomalies in Alice's ECG results and wants to refer the recent findings to Dr. Charlie who is working as a cardiac surgeon at hospital B. With the presence of delegation capability, Dr. Bob is able to temporarily delegate the attributes *Cardiologist*, *Hospital_A* to Dr. Charlie which allows him to access the EHR of Alice. Suppose, Dr. Charlie wants to obtain further expertise from his colleague Dr. John who is also a cardiac surgeon at hospital B. With delegatability, it is possible to re-delegate the attributes *Cardiologist*, *Hospital_A* to Dr. John allowing him to access Alice's EHR

via satisfying $\mathcal{T}$. Incorporating such delegating capabilities induces the following challenges which need to be taken into consideration. Suppose, Dr. Bob possesses the attribute set $\omega_{Bob} = \{Director, Hospital\_A, Cardiologist\}$. During delegation, Dr. Bob may not want to delegate all his attributes to Dr. Charlie given that delegating the attributes *Cardiologist*, *Hospital_A* is sufficient to satisfy $\mathcal{T}$. Another consideration would be Dr. Bob may or may not be interested in allowing Dr. Charlie to re-delegate the attributes. We call the delegation capability adhering to the challenges mentioned above as controlled access delegation [24, 76].

## 4.2 Related Work

When we investigate the incorporation of access delegatability into AC models, it is evident that some effort has gone into infusing delegatability to RBAC models to not only provide access flexibility but also as a mechanism to decentralize the process of the user to role assignments. The schemes proposed in [77–80] are some examples which enforce delegatability via allowing a user to delegate his role to another user in a different role. Among them, [78–80] are capable of provisioning features such as revocation as well as multi-level delegation. The foundation laid by the aforementioned delegatable RBAC models have paved the way for the development of delegatable health information sharing schemes proposed in [52, 81–83].

There have only been few related works which have explored the issue of flexible access delegation in attribute based systems. We have come across several efforts, where access delegation capabilities are integrated into ABE models to facilitate delegatable access to encrypted data. We are going to discuss these schemes in Chapter 6 since it is dedicated to the topic of provisioning fine-grained, delegatable access over encrypted data. Apart from them, there exist only one solution [84], which is based on a delegatable attribute credential scheme. Although this scheme is capable of provisioning anonymous, delegatable access with the property of selective disclosure of attributes, it induces higher end-user computational overhead when disclosing attributes from delegated credentials to a verifier. Also, the scheme does not include a mechanism to provide control over delegation, meaning that there is no mechanism to control re-delegation of credentials.

In order to address this research gap, we propose three ABAC schemes (Scheme 3, Scheme 4 and Scheme 5) which are capable of provisioning multi-level, controlled access delegation. Among them, Scheme 3 does not provide anonymity whereas Scheme 4 provides pseudo-anonymous guarantees to users while Scheme 5 capable of provisioning full anonymity.

## 4.3 Security and Privacy Requirements

We require the following security and privacy requirements to be satisfied in the proposed delegatable ABAC schemes.

- The requirements: *selective disclosure of attributes*, *resistance against attribute forgery*, *resistance against attribute collusion*, *resistance against replay attacks* and *patient privacy* are as introduced in Section 3.2.
- *Controlled access delegation:* The issuer of an attribute must have the control of delegation right meaning that further delegations by the user who received the attribute are only feasible with the consent of the current issuer as well as the consent of the AA which administrates the considered attribute. Although the consent of the AA is required, it does not mean that the issuing user should communicate with the corresponding AA before each delegation.
- *Attribute revocation:* Attribute revocation refers to the process of revoking an issued attribute from a user. When revoked, the user will not be able to use the revoked attribute to gain access to the shared EHR data. In addition, the revocation process should be decentralized, meaning that both AAs and users should be able to carry out revocation when the occasion demands.
- *User privacy:* Users must be allowed to access EHR data anonymously since the linkability of a patient EHR to the accessing user's identity could affect the privacy of the EHR accessing user. Also, the property of selective disclosure of attributes further strengthens the privacy of users.

## 4.4 Proposed Scheme 3

We present this scheme, starting with preliminaries which provides the background details associated with the proposed scheme. Then, an overview is presented before the functionality of the proposed scheme is described. This section is concluded with the security analysis and the performance evaluation of Scheme 3.

### 4.4.1 Preliminaries

First of all, we define two entities as given below, in addition to the entities in the system model given in Section 2.1.2 which corresponds to AS 1. The modified system model is illustrated in Figure 4.1.

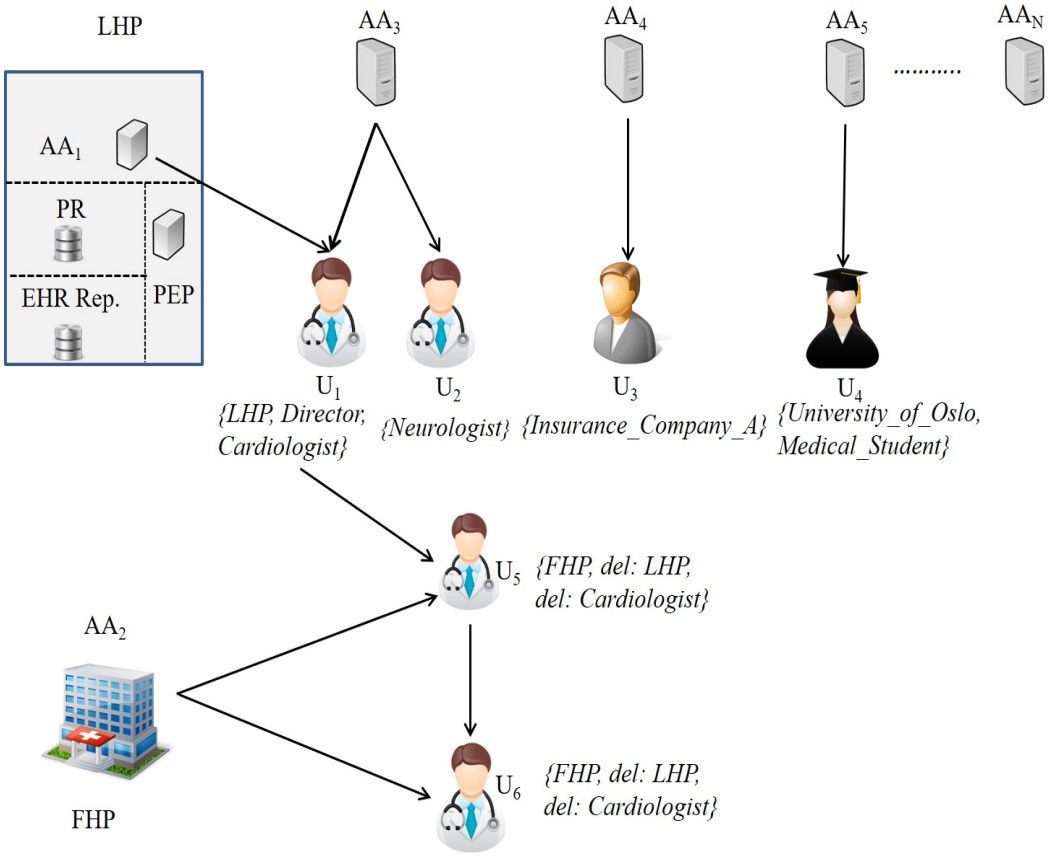



Figure 4.1: System model corresponding to Scheme 3

- **Delegator:** If a particular user is assigning an attribute (which he owns) to another user, the user who is assigning the attribute we define as the delegator.
- **Delegatee:** The user who is receiving the attribute we define as the delegatee.

**Types of attributes:** For Scheme 3, we introduce two types of attributes as defined below, based on the nature of attribute assignment.

- **Assigned attribute:** If an attribute is assigned to a user by an AA, we consider such an attribute as an assigned attribute. For instance, in Figure 4.1, the attribute *Cardiologist* of the user $U_1$ is an assigned attribute since it is assigned to $U_1$ by $AA_3$.
- **Delegated attribute:** We consider an attribute as a delegated attribute if the attribute is assigned to a particular user by another user or in other words, the issuer is not an AA. If the current delegator obtained the attribute directly from an AA, we would denote the current delegation as the first level delegation. If the delegator has obtained the considered attribute from a first level


Figure 4.2: Composition of an assignment token

delegation, then the delegator's following delegations of the same attribute (if permitted) are denoted as second level delegations. For instance, with reference to Figure 4.1, the delegation of attributes {*LHP*, *Cardiologist*} from $U_1$ to $U_5$ is a first level delegation while the delegation of the same attributes from $U_5$ to $U_6$ is a second level delegation.

Now, we wish to describe the composition of both assigned and delegated attributes.

**Composition of assigned attributes:** We represent each assigned attribute as a combination of an assignment token and a digital signature of the token.

- **Assignment token:** An assignment token contains the information about the attribute, who issued the attribute to whom, attribute expiration information as well as the maximum number of re-delegations allowed for the given attribute. Note that the issuer of an assignment token is always an AA. In order to specify who issued the attribute to whom, we use the PKI certificates of the issuer and the receiver. PKI certificates vouch for the identity of each entity; hence the issuer and the receiver of the attribute are uniquely defined. Each token also includes a delegation count ($DC$) index such that $DC \in \mathbb{Z}^{\geq 0}$. If $DC = 0$, the receiver of the attribute is not eligible to further delegate the attribute. If $DC = l, l > 0$, the attribute is allowed to be delegated at most $l$ times. Thus, with the help of $DC$, the AA which administrates the attribute can limit the maximum number of subsequent re-delegations.

- **Signed assignment token:** Each assignment token is signed by the issuing AA using a secret attribute key specific for the assigned attribute defined by the issuing AA. We denote the signature of an assignment token as the assignment token signature. We adopt an attribute based signature scheme developed based on the signature scheme in [85] and the details of signature construction will be discussed in Section 4.4.4. The combination of an assignment token and its signature is called as the signed assignment token.

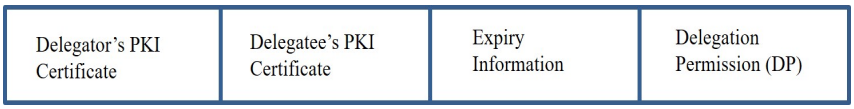



Figure 4.3: Composition of a delegation token

**Composition of delegated attributes:** We describe the composition of delegated attributes in Scheme 3, with the help of delegation tokens, delegation chain and aggregated tokens as follows.

- **Delegation token:** A delegation token contains the identity of the attribute delegator, delegatee, attribute expiration information and delegation permission ($DP$) index (either $0$ or $1$) to state the delegation permission for further delegations by the delegatee. If the $DP$ value is set to $0$, the delegatee of the current delegation is not able to delegate further. On the other hand, if $DP = 1$, the delegatee is permitted to further delegate the attribute, given that the maximum number of delegations (enforced through the $DC$ index) is not exceeded. Figure 4.2 and Figure 4.3 illustrate the composition of an assignment token and a delegation token respectively.

- **Delegation chain:** A sequence of delegations of the same attribute, we refer to as a delegation chain. Figure 4.4 shows an example of a delegation chain, where the attribute $\omega$ is initially assigned to $U_1$ by $AA_1$ followed by the delegations from $U_1$ to $U_2$ and $U_2$ to $U_3$. A delegation chain is regarded as a valid delegation chain if the following conditions are met.

  - A delegation chain should initiate with an attribute assignment.
  - The attribute assignment and each subsequent delegation in the chain must be valid.
  - The number of delegations in the delegation chain should not exceed the maximum allowable limit.

  With reference to Figure 4.1, consider the following two delegation chains for the attribute *LHP* where "$\rightarrow$" means "issued to".

$$LHP : U_1 \rightarrow U_5 \rightarrow U_6 \tag{4.1}$$

$$LHP : AA_1 \rightarrow U_1 \rightarrow U_5 \rightarrow U_6 \tag{4.2}$$

  Statement (4.1) presents a delegation chain of the attribute *LHP*, which is delegated from $U_1$ to $U_5$ and $U_5$ to $U_6$. However, this chain of delegations does not have the information on the initial assignment of the

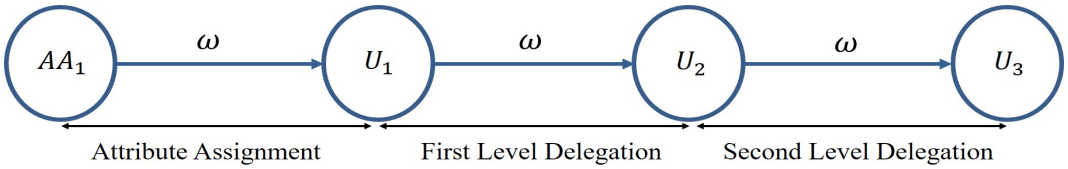



Figure 4.4: A delegation chain associated with the attribute $\omega$

attribute from $AA_1$. On the other hand, statement (4.2) shows a chain of delegations which is initiated by $AA_1$. Hence, only the delegation chain represented in the statement (4.2) is a valid delegation chain.

- **Aggregated token:** When a user acquires a delegated attribute, the user requires a token that contains verifiable evidence that the corresponding delegation chain is valid. We call such a token as an aggregated token. The aggregated token generation is explained below with the help of an example.

  Consider the assignment of the attribute $\omega =$ ***LHP*** from $AA_1$ to $U_1$ and the first level delegation of the attribute $\omega$ from $U_1$ to $U_5$. Further, assume that these tokens are denoted with $AT_1$ and $AT_2$ respectively. Also, we use the notations $t_{k,\omega}$ and $t'_{m,\omega}$ to denote the secret attribute keys associated with $AA_k$ and $U_m$ for the attribute $\omega$. More information on secret key generation and assignment is given in Section 4.4.3, when the proposed AC scheme is described in detail.

  Suppose, $U_1$ has received the signed assignment token $ST_1 = \{AT_1, SIG_1\}$ from $AA_1$, where $SIG_1$ denotes the signature of $AT_1$, which is constructed using the secret attribute key $t_{1,\omega}$ defined by $AA_1$ for the attribute $\omega = LHP$. The structure of $AT_1$ and $SIG_1$ are given below. Note that, we have only represented the issuer and the receiver in the respective tokens and omitted the rest of the details (as presented in Figure 4.2 and Figure 4.3) for the ease of illustrations.

$$AT_1 = LHP : AA_1 \rightarrow U_1$$

$$SIG_1 = [AT_1]_{t_{1,\omega}}$$

  When $U_1$ delegates the attribute ***LHP*** to $U_5$, $U_1$ first generates a delegation token $AT_2$ which includes the delegation information (delegator, delegatee, expiry information and the $DP$ value).

$$AT_2 = LHP : U_1 \rightarrow U_5$$

Then, $U_1$ concatenates $AT_1$ and $AT_2$ and signs the concatenated token $(AT_1||AT_2)$ using $t'_{1,\omega}$ which is the secret attribute key belongs to $U_1$ for the attribute $\omega = LHP$. If the signature of the concatenated token is denoted with $SIG_2$,

$$SIG_2 = [AT_1||AT_2]_{t'_{1,\omega}}.$$

Then, $U_1$ can generate the aggregated token $AGT$ using $AT_1$, $AT_2$, $SIG_1$, $SIG_2$ such that,

$$\begin{aligned} AGT &= \{AT_1, SIG_1\}||\{AT_2, SIG_2\} \\ &= \{AT_1, [AT_1]_{t_{1,\omega}}\}||\{AT_2, [AT_1||AT_2]_{t'_{1,\omega}}\}. \end{aligned}$$

The aggregated token $AGT$ allows $U_5$ to provide evidence for the presence of a valid delegation chain through recursively validating tokens in the aggregated token.

A general structure for the aggregated token can be given as follows. Suppose, the attribute $\omega$ is assigned by $AA_k$ to $U_1$ and it is delegated to $U_m$ through $(m-1)$ delegations ($U_2, U_3, ..., U_{m-1}$ are intermediate entities in the delegation chain). If the aggregated token issued by $U_{m-1}$ to $U_m$ for the attribute $\omega$ is denoted by $AGT_{m-1,m}$, then,

$$AGT_{m-1,m} = AGT_{m-2,m-1}||\{AT_m, [AT_1||...||AT_m]_{t'_{m-1,\omega}}\}$$

where,

$$\begin{aligned} AGT_{m-2,m-1} = \{AT_1, [AT_1]_{t_{k,\omega}}\}||\{AT_2, [AT_1||AT_2]_{t'_{1,\omega}}\}|| \\ ...||\{AT_{m-1}, [AT_1||AT_2||...||AT_{m-1}]_{t'_{m-2,\omega}}\}. \end{aligned} \tag{4.3}$$

Note that $AGT_{m-2,m-1}$ is the aggregated token received by $U_{m-1}$ from $U_{m-2}$ and $AT_1, AT_2, ..., AT_m$ denote the $m$ tokens (assignment token and $(m-1)$ delegation tokens).

### 4.4.2 Overview of Scheme 3

We refer to the system model illustrated in Figure 4.1. The system composed of multiple AAs, each responsible for a different set of attributes. Furthermore, every AA first defines a set of secret exponents and public key components such that each managed attribute is associated with a secret attribute exponent and a public

attribute key. To facilitate the delegation of attributes, each user must also be able to act as a virtual AA. Hence, users need to pursue the same initialization mechanism.

Users can obtain attributes from AAs (in the form of signed assignment tokens) or fellow users (in the form of aggregated tokens) providing evidence that they are qualified for ascertaining the requested attributes. We assume that each entity (users and AAs) in the system is pre-loaded with a PKI certificate vouching for the identity of the particular entity. Thus, the issuer and the receiver fields in any token (assignment token or delegation token) are occupied by the PKI certificates of the issuer and the receiver. We generate digital signatures with the help of the secret attribute exponent corresponding to the attribute (assigned or delegated), defined by the issuing entity. Hence, verification of a signature requires the public attribute key corresponding to the associated attribute, defined by the issuing entity. This construction of the signature ensures attribute revocation since the issuing entity could simply change the relevant attribute's secret exponent and thereby the public attribute key to trigger an attribute revocation.

When a user wants to access a stored $EHR_{obj}$ of a patient, the access requesting user first needs to mutually authenticate with the LHP. We use the pre-loaded PKI certificates and the associated public and private keys to establish the mutual authentication. Initial authentication serves us in two ways. Firstly, it helps the access requesting user to ensure that he is in communication with the LHP before revealing the attributes via tokens. More importantly, it helps in preventing impersonation attacks. We can prevent impersonation attacks through the authentication as follows. As we have explained, issuer's and receiver's PKI certificates are embedded in each token. Suppose, an access requesting user commits a signed assignment token for verification to the LHP. Although the verification of the signature will allow the LHP to determine the validity of the assignment token as well as who issued it to whom, LHP will not be able to learn whether the user who committed the token is actually the same user who owns the attribute token (owner of the committed token is identified with the receiver's PKI certificate in the token). However, during the initial authentication, the access requesting user must use the public and private keys associated with his PKI certificate (which is included in the token) to mutually authenticate with the LHP. Hence, LHP can realize that the access requesting user owns the committed attribute token since the ownership of the PKI certificate included in the token's receiver segment is verified via a successful authentication. Thus, using PKI certificates to identify the identity of the receiver of a token coupled with making the user authenticate with the same PKI certificate allow us to prevent impersonation attacks. It is important to note that this authentication mechanism is

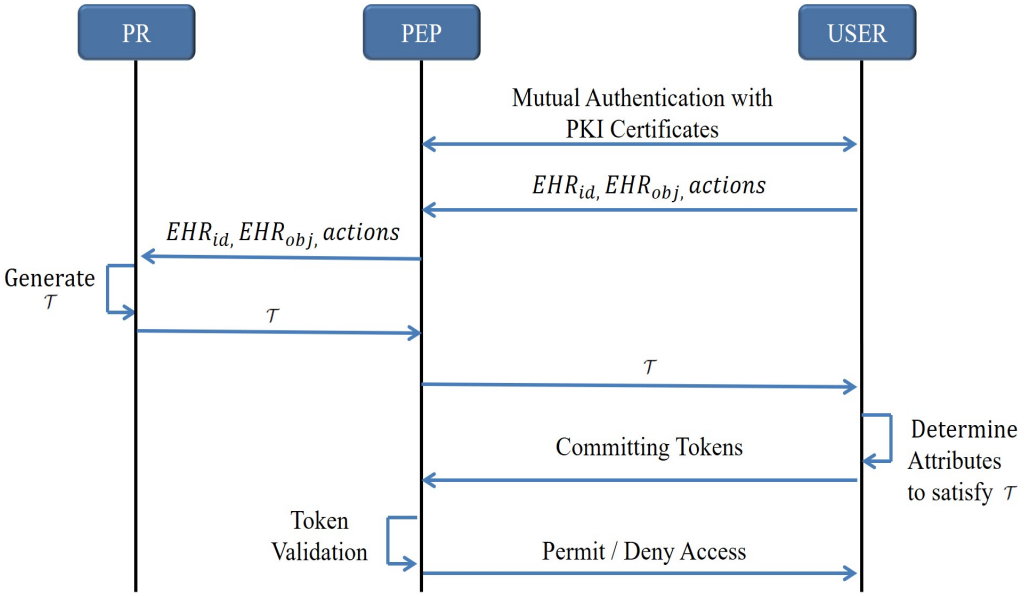



Figure 4.5: Overview of Scheme 3

just to achieve the aforementioned requirement and the authorization decisions are made completely based on the committed attributes.

After a successful authentication, the access requesting user can initiate an EHR access request indicating the $EHR_{id}$, $EHR_{obj}$ as well as the intended actions to be performed on the $EHR_{obj}$. According to the access request, the policy related to the $EHR_{obj}$ is fetched, and the corresponding Boolean statement $\mathcal{T}$ describing the access requirement is generated and forwarded back to the corresponding user by the PEP of the LHP. The access requester will then first identify a subset of attributes which he owns, that satisfies the received $\mathcal{T}$. Then, according to the selected attributes, relevant tokens (signed assignment tokens or aggregated tokens) are sent to the PEP of LHP. PEP will first rule out the possibility of an impersonation attack using the mechanism which is described above. The PEP will then verify the tokens using appropriate signature verifications. If the committed attributes are delegated attributes, the PEP will also check the validity of the corresponding delegation chains. A successful validation of tokens ensures that the user not only possesses the necessary attributes as well as the fact that any of the claiming attributes are not being revoked by the relevant AA or by any intermediate entity in the delegation chain. Given that the validations adhere, the access requesting user is given the permission to access the resource according to the permissions specified by the policy. A general flow diagram representing the overall functionality of the proposed AC scheme is illustrated in Figure 4.5.

In the following subsections, we present the proposed scheme in detail, by dividing its functionality into four phases: system initialization, attribute token distribution, AC mechanism and attribute revocation.

### 4.4.3 System Initialization

Initially, all AAs agree on a set of global parameters $(\mathbb{G}_0, \mathbb{G}_1, e, g, p, H)$ where $\mathbb{G}_0$, $\mathbb{G}_1$ are two multiplicative cyclic groups of order $p$ with $g$ being the generator, $e$ is a bilinear mapping such that $e : \mathbb{G}_0 \times \mathbb{G}_0 \rightarrow \mathbb{G}_1$ and $H$ is a secure hash function such that $H : \{0,1\}^* \rightarrow \mathbb{Z}_p^*$. Thus, any new AA can be seamlessly globally initialized by acquiring the set of global parameters which are already shared by the existing AAs. After the global initialization, each AA must be locally initialized by defining parameters related to the attributes being governed by them. The local initialization procedure is described below with respect to $AA_k$.

- Suppose that the attribute set administered by $AA_k$ is denoted by $A^k$.
- $AA_k$ chooses a random exponent $\alpha_k \in \mathbb{Z}_p^*$ as the private key of $AA_k$ and publishes $X_k = g^{\alpha_k}$ as the authority public key. Then, a unique identifier $t_{k,i} \in \mathbb{Z}_p^*$ for each element $i$ in $A^k$ is also randomly selected. Furthermore, each attribute administered by $AA_k$ is also bound with a public attribute key $T_{k,i}$, where $T_{k,i} = g^{t_{k,i}}$.
- $AA_k$ will keep $\{\alpha_k, t_{k,i}\}_{i=1,2,...,|A^k|}$ as the master secret set denoted by $MK_k$ and publish $\{X_k, T_{k,i}\}_{i=1,2,...,|A^k|}$ as the public tuple denoted by $PK_k$.

In addition, we assume that both AAs and users are pre-loaded with PKI certificates which vouch for their identity. We denote the PKI certificate of $AA_k$ with $PKI_k$ and the PKI certificate of user $U_m$ with $PKI'_m$.

### 4.4.4 Attribute Token Distribution

We subdivide the discussion on attribute distribution into two categories as the distribution of assigned attributes and the distribution of delegated attributes. Note that the following notations are used to denote an attribute token assigned from an AA to a user (assigned attribute) and an attribute token representing a delegation from a user to another user. If $AA_k$ assigns an attribute to $U_m$, the relevant token, we denote as $AT_m^k$, while the respective signature we denote as $SIG_m^k$. On the other hand, when $U_m$ delegates an attribute to $U_n$, we use $AT_{m,n}$ to denote the delegation token which includes the delegation information from $U_m$ to $U_n$ while the corresponding

signature is denoted with $SIG_{m,n}$. Furthermore, the aggregated token issued from $U_m$ to $U_n$ is denoted with $AGT_{m,n}$.

**Distribution of assigned attributes:** Recall that an assigned attribute relates to an attribute obtained directly from an AA. Let us assume that the user $U_m$ wants to acquire the attribute $\omega$ from $AA_k$. The associated process is described below.

- $U_m$ requests for the attribute $\omega$ from $AA_k$, by providing evidence for the fact that he is eligible for the requested attribute along with his PKI certificate $PKI'_m$.
- Given that $AA_k$ is satisfied with the eligibility evidence of $U_m$ to acquire the attribute $\omega$, $AA_k$ generates the assignment token $AT^k_m$ including the information about the issuing attribute, issuer's PKI certificate ($PKI_k$), receiver's PKI certificate ($PKI'_m$), expiration information along with a $DC$ value as mentioned in Section 4.4.1.
- The token is then signed by $AA_k$. The process associated with signing the token is as follows. $AA_k$ uses its public key $X_k$ and the secret attribute exponent defined (for the issuing attribute) by the $AA_k$, $t_{k,\omega}$ to generate the signature. If the signature of $AT^k_m$ is denoted by $SIG^k_m$, then,

$$SIG^k_m = X_k^{\left(H(AT^k_m)+t_{k,\omega}\right)^{-1}} = g^{\alpha_k\left(H(AT^k_m)+t_{k,\omega}\right)^{-1}}.$$

- Thus, the signed assignment token issued to $U_m$ can be represented as $\{AT^k_m, SIG^k_m\}$.

**Distribution of delegated attributes:** To explain the token generation procedure for delegated attributes, we will extend the previously discussed assigned attribute scenario to a first level delegation, in which $U_m$ delegates the attribute $\omega$ obtained from $AA_k$ to a fellow user $U_n$. Suppose, the attribute set owned by $U_m$ is denoted by $A_m$. To facilitate the delegation, $U_m$ virtually acts as an AA and therefore, $U_m$ needs to be initialized as a virtual AA. To initialize, $U_m$ first selects a secret exponent $\beta_m \in \mathbb{Z}^*_p$ as the private key of $U_m$, and computes his public key $X'_m = g^{\beta_m}$. In addition, a set of secret attribute exponents $\{t'_{m,i}\}_{i=1,2,...,|A_m|}$ are generated, where $t'_{m,i} \in \mathbb{Z}^*_p$ denotes the secret attribute exponent associated with the $i^{th}$ element in $A_m$. Furthermore, $U_m$ also generates the set of public attribute keys $\{T'_{m,i}\}_{i=1,2,...,|A_m|}$ where $T'_{m,i} = g^{t'_{m,i}}$. Similar to an AA, $U_m$ keeps $\{\beta_m, t'_{m,i}\}_{i=1,2,...,|A_m|}$ as his master secret

set $MK'_m$ while publishing $\{X'_m, T'_{m,i}\}_{i=1,2,...,|A_m|}$ as his public tuple $PK'_m$. After $U_m$ is initialized as a virtual AA, delegation tokens can be generated as follows.

- Suppose, the secret attribute exponent defined by $U_m$ for the delegating attribute $\omega$ is given by $t'_{m,\omega}$. $U_m$ first generates the delegation token $AT_{m,n}$ including the issuer's PKI certificate ($PKI'_m$), receiver's PKI certificate ($PKI'_n$), expiration information and the $DP$ value.
- To generate the aggregated token $AGT_{m,n}$, according to (4.3), $U_m$ computes the signature,

$$SIG_{m,n} = X'^{\left(H(AT^k_m||AT_{m,n})+t'_{m,\omega}\right)^{-1}}_m = g^{\beta_m\left(H(AT^k_m||AT_{m,n})+t'_{m,\omega}\right)^{-1}}.$$

- Then, $U_m$ generates the aggregated token $AGT_{m,n}$ such that,

$$AGT_{m,n} = \{AT^k_m, SIG^k_m\}||\{AT_{m,n}, SIG_{m,n}\}.$$

- Finally, the aggregated token $AGT_{m,n}$ is sent to $U_n$ to complete the delegation of the attribute $\omega$.

Figure 4.6 shows an attribute distribution scenario for both assigned and delegated attributes. We use this scenario as an example to illustrate the distribution

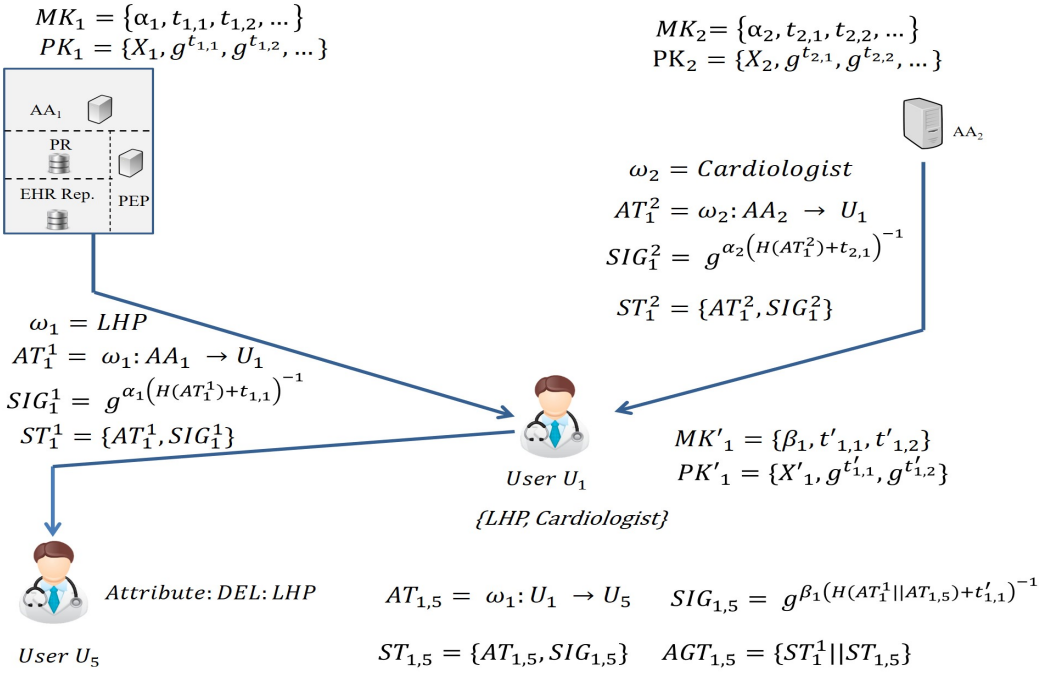



Figure 4.6: Attribute token distribution scenario

of attributes. In the given scenario, $U_1$ obtains the attribute ***LHP*** from $AA_1$ and the attribute *Cardiologist* from $AA_2$. Suppose, the secret exponent defined for the attribute ***LHP*** at $AA_1$ is $t_{1,1}$ and the public key of $AA_1$ is given by $X_1 = g^{\alpha_1}$. Then, the signed assignment token received for the attribute ***LHP*** has the form $\{AT_1^1, SIG_1^1\}$ in which $AT_1^1$ corresponds to the assignment token which includes the information of issuer's ($AA_1$) PKI certificate, receiver's ($U_1$) PKI certificate, attribute expiration information and the $DC$ value, which determines the maximum length of the delegation chain. $SIG_1^1$ is the signed form of $AT_1^1$, where $SIG_1^1 = g^{\alpha_1\left(H(AT_1^1)+t_{1,1}\right)^{-1}}$ generated with the help of the public key of $AA_1$ $g^{\alpha_1}$, and the relevant private attribute exponent $t_{1,1}$. Similarly, $U_1$'s attribute *Cardiologist* takes the form $\{AT_1^2, SIG_1^2\}$, where $SIG_1^2 = g^{\alpha_2\left(H(AT_1^2)+t_{2,1}\right)^{-1}}$. Note that both above mentioned attributes are assigned attributes since they are issued by the respective AAs.

Now, let us consider the first level delegation of the attribute ***LHP*** from $U_1$ to $U_5$. To facilitate the delegation, $U_1$ first generates a new token $AT_{1,5}$ which includes the information about the current delegation from $U_1$ to $U_5$. To be exact $AT_{1,5}$ composed of the issuer's ($U_1$) PKI certificate, receiver's ($U_5$) PKI certificate, expiration information which includes the validity period of the delegating attribute as well as the $DP$ value. Then, $U_1$ generates the signature of $AT_1^1||AT_{1,5}$, where $AT_1^1$ is the assignment token corresponding to the attribute ***LHP*** received from $AA_1$. The generated signature is given by $SIG_{1,5} = g^{\beta_1\left(H(AT_1^1||AT_{1,5})+t'_{1,1}\right)^{-1}}$, where $t'_{1,1}$ is the secret attribute exponent defined by the user $U_1$ for the attribute ***LHP***. Then, $U_1$ generates the aggregated token, $AGT_{1,5} = \{AT_1^1, SIG_1^1\}||\{AT_{1,5}, SIG_{1,5}\}$ and forwards it to $U_5$ to complete the first level delegation of the attribute ***LHP***.

### 4.4.5 AC Mechanism

In this subsection, we explain how access decisions are made at the LHP by verifying committed attribute tokens owned by the access requester, which enables the LHP to determine whether the user possesses a set of attributes that satisfy the governing access policy. Suppose, the access requesting user $U_m$ is already mutually authenticated with the LHP with $U_m$'s PKI certificate $PKI'_m$. After the authentication process, $U_m$ sends an EHR resource access request indicating the $EHR_{id}$, $EHR_{obj}$ and actions intended to be performed on the requested $EHR_{obj}$. When the request is received at the PEP of the LHP, it forwards the request to the PR in which the relevant Boolean statement $\mathcal{T}$ (corresponding to the associated access policy) is fetched and returned to the PEP. Then, the PEP will forward $\mathcal{T}$ to $U_m$, requesting at-

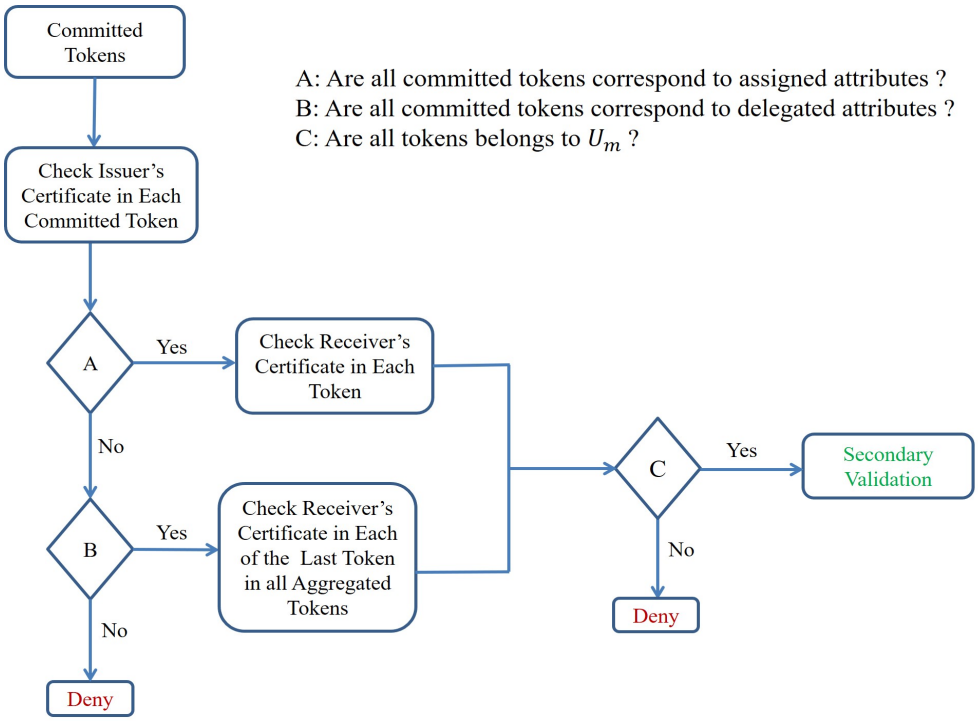



Figure 4.7: Primary validation criterion

tribute tokens for a set of attributes he owns that satisfy $\mathcal{T}$. Based on the received $\mathcal{T}$, $U_m$ first determines the smallest subset of attributes $A'_m$ which satisfies the received $\mathcal{T}$, where $A'_m \subseteq A_m$. Then, based on $A'_m$, $U_m$ sends the relevant tokens (signed assignment tokens or aggregated tokens) to the LHP for verifications. When tokens are received at the PEP, it will first carry out a primary validation, and a successful primary validation will trigger the secondary validation which involves signature verification.

**Primary validation:** Rules relevant to the primary validation are described below, and Figure 4.7 represents the associated rules in the form of a flow diagram.

- In the committed token set, if there exist both signed assignment tokens (corresponding to assigned attributes) as well as aggregated tokens (corresponding to delegated attributes) the access request is rejected. This is to prevent the possibility of colluding of attributes for gaining access to the resources. Note that a signed assignment token represents an attribute directly received from an AA while an aggregated token corresponds to an attribute received from another user. Hence, a commitment of both assigned attributes and delegated attributes constitutes an attribute collusion.

- If all the committed tokens are assigned attribute tokens, PEP further checks the receiver's PKI certificate embedded in each of the tokens to identify the owner of each token. All committed tokens should have the same receiver $U_m$, hence all tokens must have the PKI certificate of $U_m$ as the receiver. Note that PEP already validated the PKI certificate of $U_m$ since it was utilized when authenticating with LHP. Hence, if any of the committed token's receiver certificate is not $U_m$'s PKI certificate, the access request is terminated. This helps in preventing impersonation attacks as well as attribute collusion.

- If all the committed tokens are aggregated tokens, PEP checks the receiver's PKI certificate embedded in each of the last token in the respective aggregated tokens to determine whether all aggregated tokens belong to the same user $U_m$.

- If all the above mentioned conditions are satisfied, PEP will go ahead with the secondary validation.

**Secondary validation:** From Figure 4.7, it is evident that the secondary validation will trigger from two possible scenarios. They are,

- Scenario 1: All committed tokens correspond to assigned attributes (all attributes are issued to $U_m$ by one or more AAs).

- Scenario 2: All committed tokens correspond to delegated attributes.

First, we present the validation mechanism associated with Scenario 1 followed by the Scenario 2.

- **Secondary validation in Scenario 1:** Suppose, $U_m$ has committed $N$ assigned attribute tokens. Access will only be granted if the validity of each committed token is verified. Let us consider a token out of the $N$ commitments and assume that it is issued to $U_m$ by $AA_k$ for the attribute $\omega$. Then, the signed assignment token has the form $\{AT_m^k, SIG_m^k\}$ in which $SIG_m^k$ is given by,

$$SIG_m^k = X_k^{\left(H(AT_m^k)+t_{k,\omega}\right)^{-1}} = g^{\alpha_k\left(H(AT_m^k)+t_{k,\omega}\right)^{-1}}$$

  where $t_{k,\omega}$ corresponds to the secret attribute exponent defined by $AA_k$ for the assigned attribute $\omega$.

As the first step of secondary validation, PEP analyzes the expiration information in $AT_m^k$ to determine whether the token is not expired. If it is not expired, signature verification is carried out as follows.

- Suppose, the public key of the considered attribute at $AA_k$ is denoted with $T_{k,\omega} = g^{t_{k,\omega}}$. Then, with the help of $T_{k,\omega}$ and $AT_m^k$, PEP computes a helper string $\Omega$ where,

$$\Omega = g^{H(AT_m^k)} \cdot T_{k,\omega} = g^{H(AT_m^k)+t_{k,\omega}}.$$

- Then, PEP uses $SIG_m^k$, $\Omega$ and $X_k$ to check whether,

$$e(SIG_m^k, \Omega) \stackrel{?}{=} e(X_k, g)$$

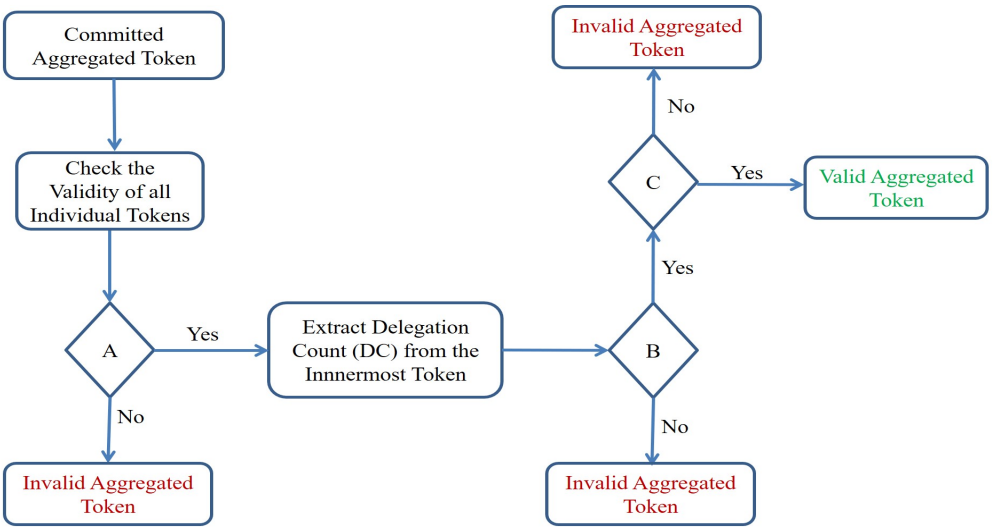
  to determine the attribute $\omega$ is valid and not revoked by the authority.

- Finally, if all committed signed assignment tokens are found to be valid, $U_m$ will be granted the access to the requested resource.

- **Secondary validation in Scenario 2:** In Scenario 2, we consider the case where $U_m$'s commitments are aggregated tokens. Flow diagram given in Figure 4.8 depicts the method associated with validating an aggregated token. For illustration purposes, assume that a committed aggregated token has $L$ tokens. The validation procedure of the aggregated token is as follows.

  PEP starts the validation process with the first token of the aggregated token. If the issuer of the first token is not an AA, the access request will be rejected, since the first token must correspond to an assigned attribute. Then, the expiration information in the first token will be examined, and if it is not expired, the signature of the first token will be verified as described in Scenario 1. If the signature is also verified, the assignment token (first token) in the aggregation is considered as valid. The validation of any intermediate token in the aggregated token proceeds as follows. Suppose, the $i^{th}$ token in the aggregation is denoted by $AT_i$. To confirm the validity of the intermediate token $AT_i$, the following conditions must be satisfied.

  - If $i = 2$, the token $AT_{i-1}$ must be valid and it must have a $DC$ value where $DC \geq 1$. If $i > 2$, the token $AT_{i-1}$ must be valid, and it must have a $DP$ value where $DP = 1$.




A: Are all individual tokens in the aggregated token are valid ?
B: $DC \geq L - 1$ ?
L = Total number of tokens in the aggregated token
C: Is the delegation chain is valid ?

Figure 4.8: Validation process of an aggregated token

- If $i > 1$, the receiver of $AT_{i-1}$ must be the delegator of $AT_i$.
- $AT_i$ should not be expired.
- If the aforementioned three conditions are satisfied, signature of $AT_i$ is verified as follows. To validate the signature, LHP first computes $\Gamma$ such that,

$$\Gamma = AT_1||AT_2||...||AT_i.$$

Note that the tokens $AT_1, ..., AT_{i-1}$ are obtained from the preceding (already verified) $(i-1)$ tokens in the aggregated token. Then, LHP can compute the helper string $\Omega$ such that,

$$\Omega = g^{H(\Gamma)} \cdot T'_{i,\omega} = g^{H(\Gamma)+t'_{i,\omega}}$$

where $T'_{i,\omega}$ is the public attribute key defined by the issuer of $AT_i$ for the attribute $\omega$. With the help of $\Omega$ and the signature of the token $AT_i$, $SIG_i = g^{\beta_i\left(H(\Gamma)+t'_{i,\omega}\right)^{-1}}$ where $\beta_i$ is the private key associated with the issuer of $AT_i$, LHP computes,

$$e(SIG_i, \Omega) = e\left(g^{\beta_i\left(H(\Gamma)+t'_{i,\omega}\right)^{-1}}, g^{H(\Gamma)+t'_{i,\omega}}\right).$$

If $e(SIG_i, \Omega) = e(X'_i, g)$, where $X'_i = g^{\beta_i}$ is the public key of the issuer of $AT_i$, the signature is verified.

If all the individual tokens in the aggregated token are valid, PEP will check the $DC$ value specified in the first token which represents the attribute assignment from the relevant AA. This is to determine whether or not the maximum permissible number of delegations specified by the AA (in relation to the attribute $\omega$) is violated or not by the respective delegatees.

If the following condition,

$$DC \geq L - 1$$

is held, where $L$ is the total number of individual tokens in the aggregated token, PEP can determine that the number of delegations in the chain is within the maximum specified limit.

Let us consider that there were $N$ aggregated tokens committed by $U_m$ to provide evidence for the ownership of $N$ valid attributes. Suppose that each of them constitutes valid delegation chains. Although all delegated attributes are valid, still PEP has to make sure that these attributes are not colluded attributes. Thus, PEP checks whether all the delegation chains are identical. We call two delegation chains are identical if the following conditions are satisfied.

- The number of tokens in each of the aggregated tokens is the same.
- The receiver of the first token in each of the aggregated token must be the same.
- Both the issuer and the receiver of every other token in respective aggregated tokens must be the same.

If all the delegation chains corresponding to $N$ committed aggregated tokens are identical, $U_m$ will be granted access to the requested resource.

We need to make a final remark before the conclusion of the AC mechanism. In the proposed AC mechanism, we have ensured that a user will be allowed to gain access to a particular EHR of a patient with delegated attributes, if and only if all the intermediate users in the delegation chain can access the particular EHR with the delegated attributes. We achieved this via prevention of colluding assignment tokens and aggregated tokens as well as checking whether the delegation chains are

identical or not. This induces more control over the process of access delegation while somewhat restricting the access flexibility. However, it is also possible to allow access with any combination of assignment and aggregated tokens by simply altering the rules associated with primary and secondary validation mechanisms.

### 4.4.6 Attribute Revocation Mechanism

Every attribute token includes expiration information which determines the valid period of the considered attribute token. However, if we depend upon this information completely, there is no way of revoking the attribute until it gets expired. Consider the following scenario. Suppose, $U_1$ with the attribute *Director* delegates it to $U_2$ for a period of one month since $U_1$ is away. If $U_1$ resumes his duties after two weeks, $U_1$ may need to revoke $U_2$'s privileges as the director of LHP immediately, without requiring to wait for another two weeks. Therefore, it is important to integrate attribute revocation in the proposed scheme.

We achieve attribute revocation as follows. In the proposed scheme, the token signatures are generated with the help of a secret attribute exponent defined by the token issuing entity, and therefore the related public attribute key is required to validate the signed token. When a specific user needs to be revoked from using a particular attribute, the issuer of the attribute (either an AA or a delegator) can enforce it by taking the following steps. Assume that, $AA_k$ wants to revoke $U_m$ from using the attribute $\omega$ which is managed by the $AA_k$.

- $AA_k$ first generates a new random secret exponent $t_{k,\omega} \in \mathbb{Z}_p^*$ for the attribute to be revoked ($\omega$) and based on the new secret, $AA_k$ generates and publishes the new public attribute key $g^{t_{k,\omega}}$.
- With the help of the new attribute exponent, $AA_k$ generates newly signed assignment tokens and forwards them to all the users who acquired the attribute $\omega$ from $AA_k$ except the user to be revoked ($U_m$).

Thus, the revoked user $U_m$ will not be able to use his old signed assignment token to get authorized since the associated signature will not be validated given that the public attribute key is modified. Note that the revocation of a delegated attribute can be achieved in the same way. Then, the revocation is handled by the user who delegated the attribute.

As we have described above, when an attribute is revoked from a given user, the adopted revocation mechanism requires issuing of new tokens (relevant for the revoking attribute) for each non revoked user. However, we can reduce the number

of revocations through proper handling of token expiration time. For instance, when a user is delegating an attribute to a fellow user, the delegation will most probably be temporary. Hence, the delegating user can issue the token with a substantially smaller token validity period to allow the delegated attribute to be naturally revoked through token expiration.

### 4.4.7 Security Analysis

In this subsection, we intend to show that Scheme 3 is secure against attribute forgery, replay attacks and attribute collusion which helps in preventing illegitimate disclosure of patients' EHR data to unauthorized parties.

**Resistance against attribute forgery:** We aim to demonstrate that the adopted signature scheme is universally unforgeable under the DL assumption which ensures that attribute tokens are unforgeable given that the DL assumption is held. We validate the universal unforgeability with the help of a game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$. The game proceeds in three steps.

- **Setup:** The challenger $\mathcal{C}$ acts as an AA, and generates the master secret set $MK_{\mathcal{C}}$ and the public tuple $PK_{\mathcal{C}}$ such that $MK_{\mathcal{C}} = \{\alpha_{\mathcal{C}}, t_{\omega}\}$ and $PK_{\mathcal{C}} = \{g^{\alpha_{\mathcal{C}}}, g^{t_{\omega}}\}$. Note that $t_{\omega}$ corresponds to the secret attribute exponent associated with the attribute $\omega$. $PK_{\mathcal{C}}$ is available to the adversary $\mathcal{A}$ along with the global parameters $(\mathbb{G}_0, \mathbb{G}_1, e, g, p, H)$.

- **Challenge:** The challenger $\mathcal{C}$ generates a message $\hat{m}$ and challenges the adversary $\mathcal{A}$ to generate a valid signature of $\hat{m}$.

- **Forgery:** The adversary $\mathcal{A}$ outputs the signature $(\hat{m}, SIG_{\hat{m}})$ and wins the forgery game if, $SIG_{\hat{m}}$ is a valid signature of $\hat{m}$ for the attribute $\omega$.

Suppose, the forged signature of the adversary $\mathcal{A}$, $SIG_{\hat{m}}$ is given by $g^{\sigma\alpha_{\mathcal{C}}}$ where $\sigma \in \mathbb{Z}_p^*$. If $SIG_{\hat{m}}$ is a valid signature of $\hat{m}$ for the attribute $\omega$, the following condition must be held.

$$e\big(g^{\sigma\alpha_{\mathcal{C}}}, g^{t_{\omega}} \cdot g^{H(\hat{m})}\big) = e(g^{\alpha_{\mathcal{C}}}, g) \tag{4.4}$$

From (4.4), it is possible to deduce,

$$\sigma\alpha_{\mathcal{C}}\big(t_{\omega} + H(\hat{m})\big) = \alpha_{\mathcal{C}}$$

and therefore,

$$\sigma = \frac{1}{t_\omega + H(\hat{m})}.$$

This provides evidence for the fact that, for a successful universal forgery of a given message $\hat{m}$, $\mathcal{A}$ must generate $\sigma = {}^{1}/_{(t_\omega + H(\hat{m}))}$. Given that deducing $t_\omega$ from $g^{t_\omega}$ is DL hard, the security of the utilized signature scheme is reduced to the hardness of the DL problem in the group in which the signature is constructed.

The unforgeability of the signature scheme ensures that the signed assignment tokens (corresponding to assigned attributes) are unforgeable given the fact that a signed assignment token composed of an assignment token and its signature. An aggregated token (associated with a delegated attribute) composed of systematically combined two or more signed tokens. Hence, aggregated tokens will also be unforgeable, due to the fact that each signature in the aggregation is individually unforgeable.

**Resistance against replay attacks:** Assume that an adversary $\mathcal{A}$ intercepted a signed assignment token associated with the attribute $\omega$ owned by $U_m$. If $\mathcal{A}$ tried to replay the intercepted token, LHP would first request $\mathcal{A}$ to mutually authenticate with his PKI certificate. The adversary $\mathcal{A}$ can proceed in two ways. He can either use his PKI certificate or use the PKI certificate of $U_m$. If he attempts to use the PKI certificate of $U_m$, the authentication will fail since the private key associated with $U_m$'s PKI certificate is unavailable to the adversary $\mathcal{A}$. If he uses his PKI certificate, the authentication will be successful. However, when he replays the signed assignment token (after authentication), LHP will detect an impersonation, since the PKI certificate included in the token (receiver's PKI certificate) will be different from what he used to mutually authenticate with the LHP.

Similarly, it is also possible to show that the proposed scheme can guard against replay attacks mounted via delegated attributes. If we assume that $U_m$ delegates the attribute $\omega$ to $U_n$, then $U_n$ will receive an aggregated token from $U_m$. Note that the aggregated token includes the assignment token and the delegation token (which includes delegator's ($U_m$) PKI certificate, delegatee's ($U_n$) PKI certificate, etc.). If $\mathcal{A}$ intercepted this aggregated token and replayed it at a later time, LHP will request $\mathcal{A}$ to authenticate with the LHP using public, private keys associated with the delegatee's PKI certificate embedded in the delegation token ($U_n$'s PKI certificate). Given that the private key associated with $U_n$'s PKI certificate is unavailable to $\mathcal{A}$, the authentication will be a failure. Therefore, our scheme can guard against attacks mounted via replaying of both assignment tokens and aggregated tokens.

**Resistance against attribute collusion:** We show that the proposed scheme can guard against collusion attacks with the help of the attack scenarios discussed below. The attack scenarios are divided into 2 categories: collusion of assignment tokens and collusion of aggregated tokens.

- **Collusion of assignment tokens:** Suppose, $AA_k$ manages the attributes $\{\omega_1, \omega_2\}$ and the $EHR_{obj}$, $O$ is associated with an access structure $\mathcal{T} = \omega_1 \wedge \omega_2$. Further, assume that $U_m$ has a valid signed assignment token $ST_1$ corresponding to $\omega_1$ and $U_n$ has a valid signed assignment token $ST_2$ corresponding to the attribute $\omega_2$.
  - **Attack scenario 1:** $U_m$ acts as the attacker and submits the two assignment tokens to gain access to $O$. During the verification, $U_m$ should first authenticate with the LHP using the public, private key pair associated with his PKI certificate. Given that $U_m$'s PKI certificate is not embedded in the token $ST_2$ as the receiver, LHP can detect that $U_m$ does not own the attribute $\omega_2$. Therefore, $U_m$'s collusion attack will not be successful.
  - **Attack scenario 2:** Suppose, $U_m$ generates a forged assignment token for $\omega_2$ by embedding $U_m$'s PKI certificate as the receiver and $AA_k$'s PKI certificate as the issuer's PKI certificate of the forged token. If $U_m$ generates a valid signed assignment token $ST_{forged}$ from the forged assignment token and sends it along with $ST_1$ to LHP the attack will be successful, and $U_m$ will be granted the access to $O$. However, we have provided evidence for the fact that successful forging of a signature is subjected to solving the DL problem. Therefore, if we assume that the DL assumption is held, the proposed scheme is secure against the attack scenario 2.
- **Collusion of aggregated tokens:** Consider the following first level delegation. Suppose, the attribute $\omega_1$ is issued to $U_l$ and $U_l$ delegates it to $U_m$. Assume that the corresponding aggregated token is denoted with $AGT_1$. $AA_k$ also issues the attribute $\omega_2$ to $U_n$. In attack scenario 3, we analyze the possibility of collusion of attributes $\omega_1$, $\omega_2$ with aggregated tokens associated with first level delegation.
  - **Attack scenario 3:** To facilitate collusion, $U_n$ delegates $\omega_2$ to $U_m$ and issues a valid aggregated token $AGT_2$ to $U_m$. Then, $U_m$ acts as the attacker and submits $AGT_1$ and $AGT_2$ to LHP to gain access to the

object $O$. To have a successful authorization, both delegation chains must be valid, and they must be identical. Although the corresponding delegation chains are valid, the chains are not identical given that the issuer and the receiver of the first level delegation tokens are not the same (In $AGT_1$ the issuer of the first level delegation token is $U_l$ whereas in $AGT_2$ the issuer is $U_n$). Hence, LHP can determine the existence of an attribute collusion. Although we have analyzed the collusion attack with the help of a first level delegation, it is also valid for the collusion of aggregated tokens corresponding to higher order delegated attributes.

Note that, it is also possible to mount an attack via colluding valid aggregated tokens and forged aggregated tokens. However, this resembles the attack scenario 2 (collision of valid and forged signed assignment tokens).

### 4.4.8 Performance Evaluation

In this subsection, we intend to provide evidence for the feasibility and efficiency of the proposed AC scheme based on the simulation results obtained with respect to the associated computational cost.

In Scheme 3, the most computationally expensive processes are token signature generation and signature verification given that the aforementioned processes require exponentiation operations in $\mathbb{G}_0$ as well as pairings. Thus, we conducted simulations for determining the average computational cost (in terms of the computation time) for the above mentioned two processes for both assigned attributes (signed assignment tokens) and delegated attributes (aggregated tokens). The simulations were run on a Core i5, $2.5\ GHz$ PC with $8\ GB$ of RAM. In order to generate the necessary cyclic groups, we used the elliptic curve $y^2 = x^3 + x$ over a $512$-bit finite field.

The variation of the computational cost associated with generation and verification of a signed assignment token with respect to the order of cyclic groups (P) are shown in Figure 4.9. According to Figure 4.9, it is obvious that the computation cost increases with the group order. However, it is also noticeable that the computation cost is under manageable proportions. For instance, let us consider the setting having the group order of $160$ bits with the elliptic curve $y^2 = x^3 + x$ over a $512$-bit finite field which said to have the equivalent security of $1024$-bit RSA according to the NIST special publication for key management recommendations in [75]. With this parameter setting, a signed assignment token can be generated in $13\ ms$ (on average) while the signature can be verified in $21\ ms$ (on average).

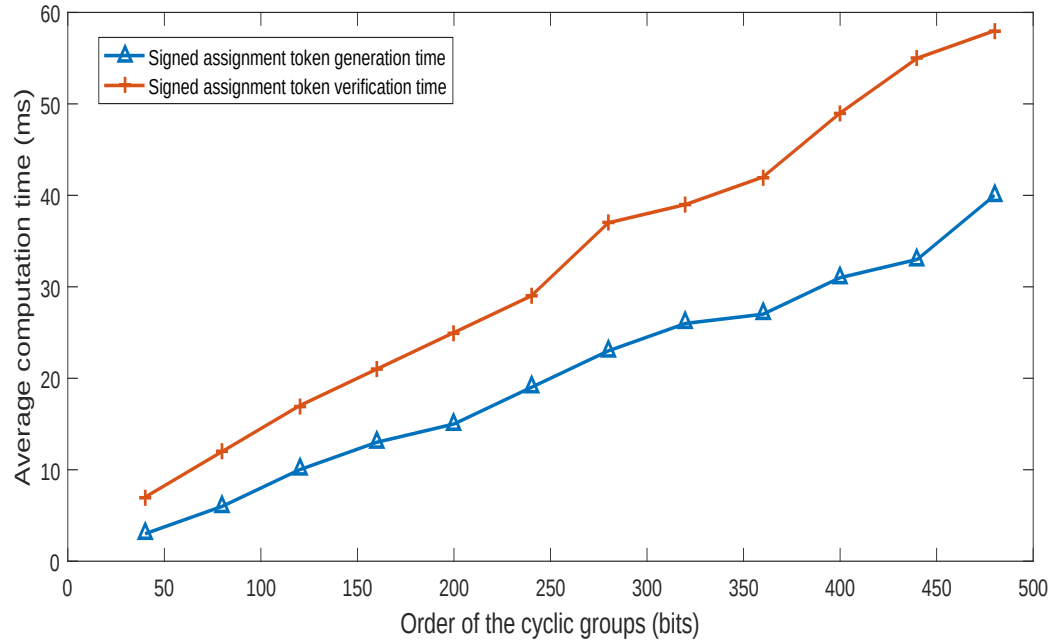



Figure 4.9: Variation of computational cost in relation to a signed assignment token with group order

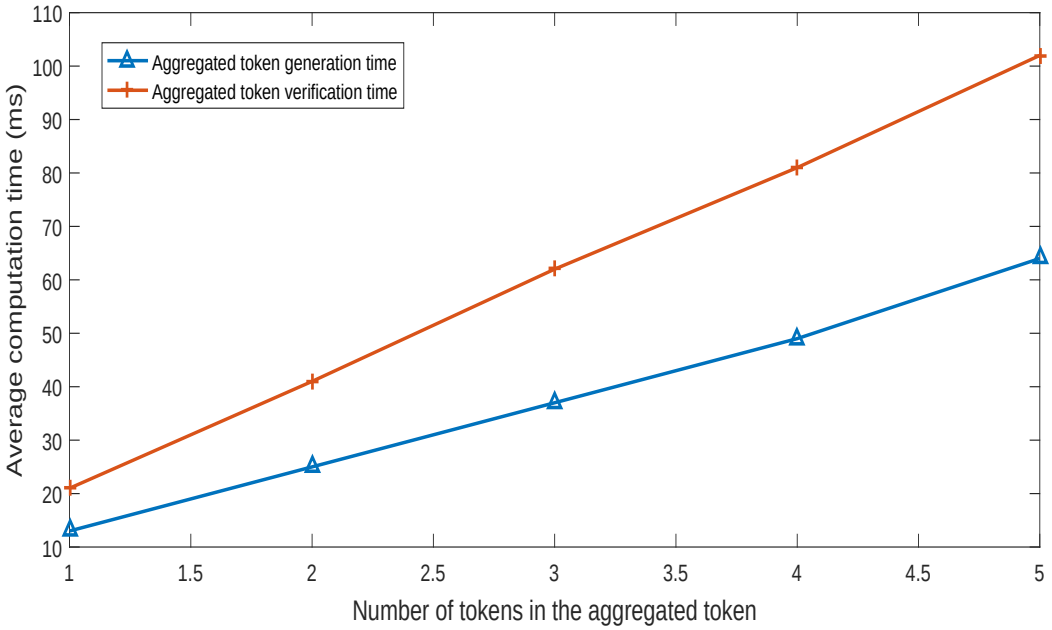



Figure 4.10: Variation of computational cost in relation to an aggregated token with the number of tokens in the aggregation

In order to analyze the behavior of computational cost associated with generation and verification of aggregated tokens, we carried out simulations while keeping the above stated parameter setting which yields equivalent security of 1024-bit RSA. The simulation results are shown in Figure 4.10 with respect to the number of tokens in an aggregated token. From Figure 4.10, it is evident that both aggregated token generation cost and verification cost exhibit a linear relationship with the number of tokens in the aggregation. This is because the number of exponentiation and pairing operations increase linearly with the number of tokens in the aggregated token. Hence, if the number of tokens in an aggregated token is $L$, then the average aggregated token generation time is given by $13L\ ms$ whereas the average verification time is given by $21L\ ms$. These simulation results show that the proposed AC scheme is computationally efficient and realistic.

## 4.5 Motivation for Scheme 4

In Scheme 3, users are allowed to further delegate their attributes in a controlled manner. We achieve the aforementioned control by allowing the delegating user to specify whether or not the delegatee is allowed to further delegate as well as bounding the maximum number of possible delegations for a given attribute. The scheme also supports attribute revocation where any user or an AA can revoke users who ascertain attributes from them. However, given that an attribute is assigned to a user in the form of an attribute token signed by the secret key associated with the attribute, revoking that user from the attribute requires issuing newly signed tokens to all the other users who ascertained the same attribute. This makes the process of revoking a user from an attribute before the expiration of the associated attribute token rather expensive. Also, Scheme 3 is not capable of provisioning user privacy given that their identities are revealed through the tokens.

In Scheme 4, we have addressed these issues through proposing a delegatable ABAC scheme conducive for a multi-domain e-health environment using blockchain technology. Blockchain [86–88] is a data structure represented with a series of data blocks such that each data block embeds a hash pointer to the previous block. The head of the block is simply a hash pointer to the most recent data block which is stored securely. This has been the foundation for recording cryptocurrency transactions [89–92] and we have adopted this to record attribute assignments, delegations as well as revocations.

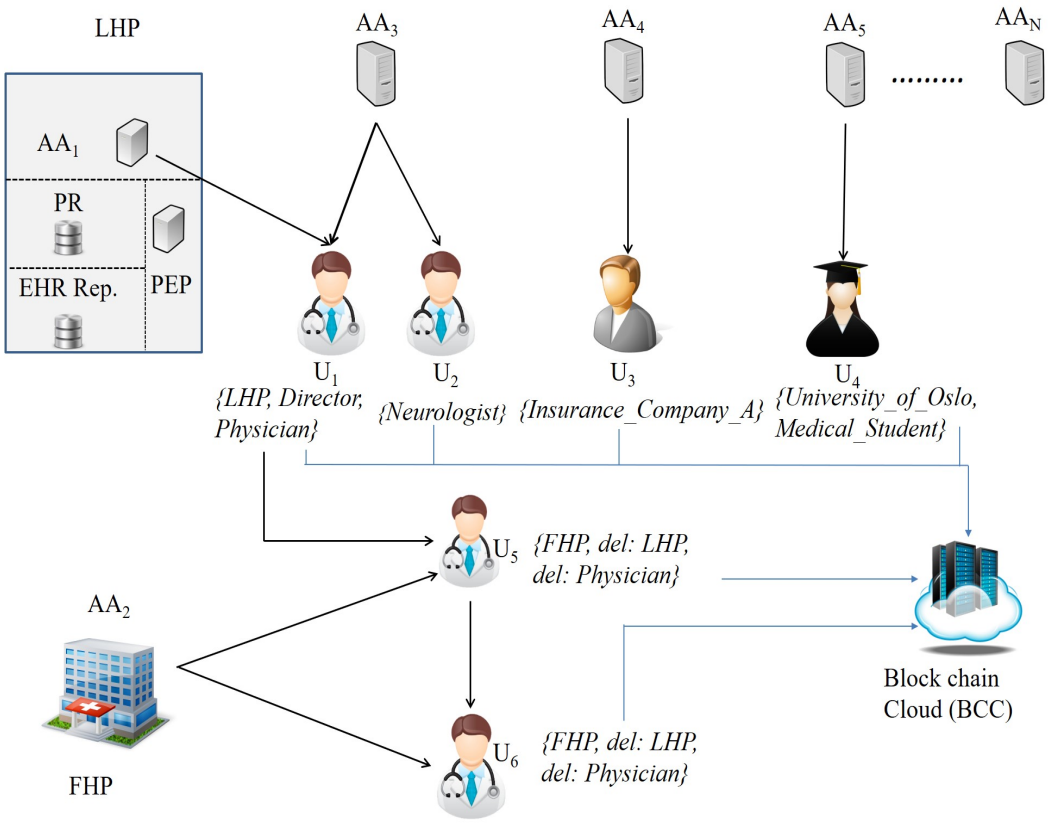



Figure 4.11: System model associated with Scheme 4

## 4.6 Proposed Scheme 4

Scheme 4 is presented starting with some preliminary information followed by the detailed description of the functionality of the proposed scheme. This section is concluded with the security analysis of Scheme 4.

### 4.6.1 Preliminaries

First of all, we modify the general system model associated with AS 1 (presented in Figure 2.2), to comply with Scheme 4. The modified system model is illustrated in Figure 4.11. In addition to the entities in the system model given in Figure 2.2, the system model corresponding to Scheme 4 includes an additional entity termed as the blockchain cloud (BCC). BCC maintains a set of blockchains which provide confirmations of attribute assignments, delegations as well as revocation of users from specific attributes. We assume that the BCC is semi-trusted meaning that it will follow the protocols specified accordingly while being curious about the information being stored. We have represented the BCC as a single server for ease of illustration, but it can simply be extended to a set of servers to provide better

availability. This would require an algorithm to achieve the consensus of stored blockchains in individual servers, and we consider this outside the scope of this thesis.

We define three types of attribute related transactions: assignment transactions, delegation transactions and revocation transactions as explained in the following.

**Assignment transactions:** If an attribute is assigned to a user by an AA, we consider such an attribute as an assigned attribute. The information that enters the blockchain representing the aforementioned attribute assignment is referred to as an assignment transaction.

**Delegation transactions:** We consider an attribute as a delegated attribute if the attribute is assigned to a particular user by another user who owns the attribute. As we have already defined for Scheme 3, if the current delegator obtained the attribute directly from an AA, we denote the current delegation as the first level delegation. If the delegator has obtained the considered attribute from a first level delegation, then the delegator's following delegations of the same attribute (if permitted) are denoted as second level delegations. The representations of these delegations in the blockchain are referred to as delegation transactions.

**Revocation transactions:** The information which represents a revocation of an attribute from a given user in the blockchain is defined as a revocation transaction. A revocation transaction can be initiated by both AAs and users to revoke attributes of the users who acquired attributes from them.

Now, we will describe the functionality of Scheme 4 from Section 4.6.2 - Section 4.6.5 by dividing the functionality into four phases as system initialization, attribute distribution with insertions to the blockchain, revoking attributes from users and AC mechanism.

### 4.6.2 System Initialization

Suppose, there exists $N$ AA's each managing a disjoint set of attributes, and each AA has a PKI certificate and an associated RSA public and private key pair. If the $k^{th}$ AA is denoted with $AA_k$, its PKI certificate is denoted with $PKI_k$ while $(PK_k, SK_k)$ are used to denote the respective public and private keys. Also, we assume that the BCC's PKI certificate is denoted with $PKI_{BC}$ and the associated RSA key pair is denoted with $(PK_{BC}, SK_{BC})$. Furthermore, BCC maintains $N$

blockchains such that the blockchain $BC_k$ includes information about attribute related transactions associated with the attributes of $AA_k$. Suppose, the $m^{th}$ user in the system is denoted with $U_m$ and to initialize, $U_m$ first generates a RSA key pair $(PK'_m, SK'_m)$ and computes his pseudo-identity $PI_m$ such that $PI_m = H(PK'_m)$, where $H$ is a secure cryptographic hash function. Note that a user has the capability to generate a new pseudo-identity at any time by simply generating a new RSA key pair.

### 4.6.3 Attribute Distribution

We subdivide the discussion on attribute distribution into two categories as the process of assigning attributes and delegating attributes.

**Assigning attributes:** Let us assume that the user $U_m$ wants to acquire the attribute $\omega$ from $AA_k$, and the associated process is described below.

- $U_m$ first mutually authenticates with $AA_k$ using their respective RSA keys.
- Then, $U_m$ requests for the attribute $\omega$ from $AA_k$, by providing evidence for the fact that he is eligible to ascertain the requested attribute.
- Given that $AA_k$ is satisfied with the eligibility of $U_m$ to acquire the requested attribute, $AA_k$ constructs the assignment transaction block as shown in Figure 4.12. The assignment block has an input which is set to NULL to represent that this transaction is an attribute assignment. In the output, $AA_k$ specifies the assigning attribute $\omega$, pseudo-identity of the receiver $(PI_m)$, expiration timestamp $(TS^k_m)$ (defines the period of validity) and a delegation count index, $DC$ where $DC \in \mathbb{Z}^{\geq 0}$. If $DC = 0$, $U_m$ is not allowed to delegate the attribute $\omega$ and if $DC = l, l \in \mathbb{Z}^+$, the length of the delegation chain (maximum number of re-delegations permitted starting with this attribute assignment) is constrained to $l$. Note that, if $U_m$ has requested $r$ attributes from $AA_k$, $AA_k$ will include a separate output vector for each assigning attribute as described above. Hence, there will be $r$ outputs in the assignment transaction block. In addition, a random seed $S^k_m$ is also generated and inserted into the block along with the issuer's public key $PK_k$.
- Then, $AA_k$ generates the hash of the block contents $M^k_m$ such that,

$$M^k_m = H(Input||Output||PK_k||S^k_m)$$

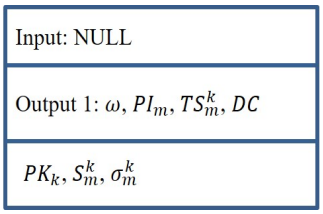



Figure 4.12: Signed assignment block generated by $AA_k$

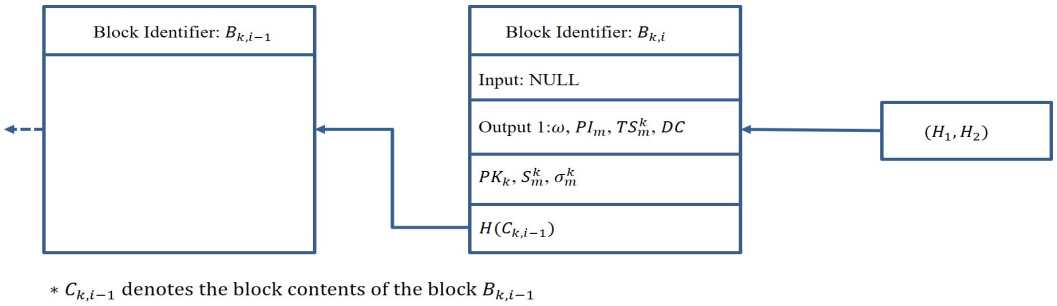



Figure 4.13: Blockchain after inserting the signed assignment transaction block generated by $AA_k$

and signs $M_m^k$ with the secret key $SK_k$ to generate the transaction signature, $\sigma_m^k = [M_m^k]_{SK_k}$. Thereafter, $AA_k$ includes the generated block signature in the transaction block.

- To transfer the aforementioned assignment block to the blockchain $BC_k$, $AA_k$ mutually authenticates with BCC using the associated RSA keys and forwards the signed transaction block.

- Suppose, there exists $(i-1)$ number of blocks in the blockchain $BC_k$. First, the BCC will validate the signature $\sigma_m^k$ with $AA_k$'s public key $PK_k$. If validated, BCC assigns a unique block identifier $B_{k,i}$ ($B_{k,i} = B_{k,i-1} + 1$) and inserts it into the received assignment transaction block. Then, the validated transaction block is connected to the blockchain $BC_k$ by inserting a hash pointer to the most recent block ($B_{k,i-1}$) as shown in Figure 4.13.

- Then, BCC forwards the block $B_{k,i}$ to $AA_k$ which allows $AA_k$ to generate the header of the blockchain $BC_k$, $(H_1, H_2)$ where $H_1 = H(C_{k,i})$ and $H_2 = [H_1]_{SK_k}$ in which $C_{k,i}$ denotes the block contents of the block $B_{k,i}$. $AA_k$ sends the chain header $(H_1, H_2)$ to the BCC. Finally, BCC validates the received block header and appends it to the blockchain $BC_k$ as shown in Figure 4.13. Given that each block in the blockchain includes the hash of the preceding block coupled with the fact that the header of the block includes the $AA_k$'s

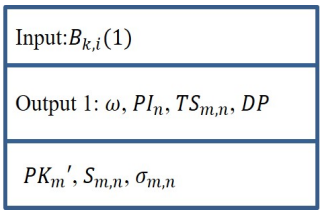



Figure 4.14: Signed delegation block generated by $U_m$

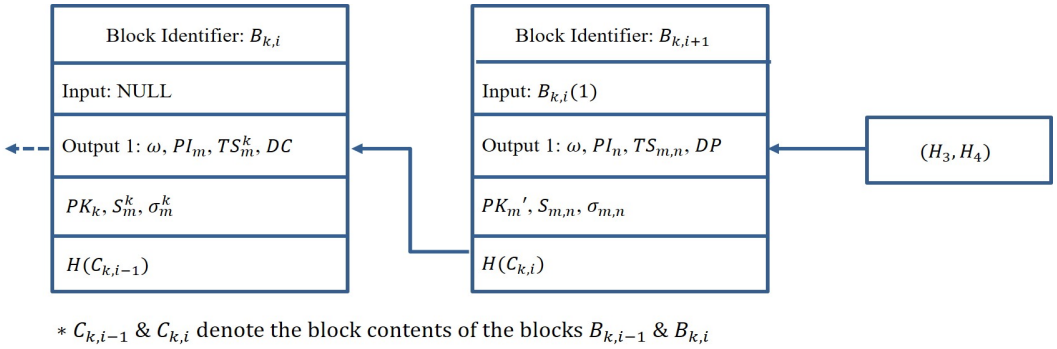



Figure 4.15: Blockchain after inserting the signed delegation transaction block generated by $U_m$

signature of the preceding block's hash value, the blockchain will be tamper resistant.

**Delegating attributes:** In order to explain the procedure associated with delegating attributes, we will extend the previously discussed assigned attribute scenario to a first level delegation, in which $U_m$ delegates the attribute $\omega$ obtained from $AA_k$ to a fellow user $U_n$. The delegation process is as follows.

- Suppose, $U_m$ has the pseudo-identity $PI_n$ of the delegatee $U_n$. Then, $U_m$ generates the delegation transaction block as illustrated in Figure 4.14. Similar to an assignment block, delegation transaction block also includes an input, output, public key of the delegator $(PK'_m)$, a random seed $(S_{m,n})$ and a delegation permission index $DP$ where $DP \in \{0, 1\}$. If $DP = 0$, $U_n$ is not allowed to re-delegate the attribute and otherwise $U_n$ will have the permission to re-delegate further. Furthermore, it also includes the signature of the block generated with the help of $U_m$'s secret key $SK'_m$. Given that this delegation should reference the initial assignment transaction from $AA_k$ to $U_m$, $U_m$ sets the input of the delegation transaction to $B_{k,i}(1)$ (i.e. the first output of the block with the identity $B_{k,i}$ of the blockchain $BC_k$). The output vector includes the delegating attribute $\omega$, pseudo-identity of the delegatee $(PI_n)$, the expiration timestamp $(TS_{m,n})$ and the delegation permission index $DP$.

- Thereafter, $U_m$ generates the hash of the block contents $M_{m,n}$ such that,

$$M_{m,n} = H(Input||Output||PK'_m||S_{m,n})$$

and signs $M_{m,n}$ with the secret key $SK'_m$ to generate the transaction signature, $\sigma_{m,n} = [M_{m,n}]_{SK'_m}$. Afterwards, $U_m$ includes the generated block signature in the delegation transaction block as shown in Figure 4.14. The signed delegation transaction block is then forwarded to the BCC.

- After receiving the signed block, BCC checks whether this first level delegation transaction is initiated by a user who has already ascertained the attribute $\omega$ from $AA_k$. To realize this, BCC examines the input vector of the delegation transaction block (which points to the associated attribute assignment) and checks whether the receiver of the assigned attribute is $U_m$. If so, BCC uses the public key of $U_m$, $PK'_m$ and validate the signature of the received delegation transaction block to ensure that it has been generated by $U_m$. If the signature is verified, coupled with the fact that both conditions $DC > 0$ and $TS^k_m \geq TS_{m,n}$ are satisfied, BCC deems the delegation as valid. Otherwise, the delegation transaction block is discarded.

- If the delegation block is valid, BCC inserts it into the blockchain after adding its block identifier $(B_{k,i+1})$ as well as making a hash pointer to the previous block $B_{k,i}$ as shown in Figure 4.15.

- To generate the new header of the blockchain, BCC mutually authenticates with $AA_k$ using their respective RSA key pair and forwards the accepted delegation transaction block $B_{k,i+1}$ to $AA_k$. $AA_k$ computes $(H_3, H_4)$ such that $H_3 = H(C_{k,i+1})$ and $H_4 = [H_3]_{SK_k}$ in which $C_{k,i+1}$ denotes the block contents of the block $B_{k,i+1}$. Then, $AA_k$ sends the newly minted chain header $(H_3, H_4)$ to BCC. Finally, BCC validates the received block header and appends it to the blockchain $BC_k$ as shown in Figure 4.15.

Although we described the construction of delegation transactions with respect to a first level delegation, it can simply be extended to higher level delegations. For instance, let us consider a $L^{th}$ level delegation of the attribute $\omega$. This delegation transaction should have an input which points to the block that represents the associated $(L-1)^{st}$ level delegation transaction on the blockchain, an output which provides the information on the delegating attribute $\omega$, pseudo-identity of the delegatee, expiration timestamp for this delegation and the DP index. Similar to the first level delegation transaction block, this block should also include the delegator's

public key, a random seed and the block signature. When this signed block reaches the BCC, it accepts the block, if and only if the following conditions are satisfied.

- The signature of the new delegation transaction block should be validated with the public key of the user who is the delegatee in the associated $(L-1)^{st}$ level delegation transaction which is already on the blockchain.
- $(L-1)^{st}$ level delegation transaction should have a $DP$ index of 1.
- $L^{th}$ level delegation should expire on or before the expiry of the $(L-1)^{st}$ level delegation (i.e. $TS_{L-1} \geq TS_L$ where $TS_{L-1}$, $TS_L$ correspond to the timestamps embedded in the respective delegation transactions).
- The associated attribute assignment block should have a $DC$ index such that $DC \geq L$.

### 4.6.4 Attribute Revocation Mechanism

Although both attribute assignments and delegations include expiration timestamp which determines the valid period of the considered attribute, in certain situations, it may be necessary to revoke the attributes before they get expired. We introduce revocation transactions to achieve this requirement. Let us assume that, $U_m$ wants to revoke $U_n$ from using the attribute $\omega$. The associated process is described below.

- $U_m$ first generates a revocation block as shown in Figure 4.16. In this case, the input of the revocation block must reference the transaction that reflects the delegation of the attribute from $U_m$ to $U_n$. Hence, the input is set to $B_{k,i+1}(1)$. To denote the revocation, the output of the newly minted revocation block is set to NULL. $U_m$ also generates a new random seed $S'_{m,n}$ and embeds it to the block along with his public key $PK'_m$.
- Then, $U_m$ generates the hash of the block contents $M'_{m,n}$ where,

  $$M'_{m,n} = H(Input||Output||PK'_m||S'_{m,n})$$

  and signs $M'_{m,n}$ with the secret key $SK'_m$ to generate the transaction signature $\sigma'_{m,n} = [M'_{m,n}]_{SK'_m}$ and forwards the signed revocation transaction block to the BCC.
- When BCC receives the signed revocation block, it verifies the signature, and if validated, the block is appended to the blockchain with a hash pointer to the previous block (i.e. $B_{k,i+1}$) as shown in Figure 4.17.

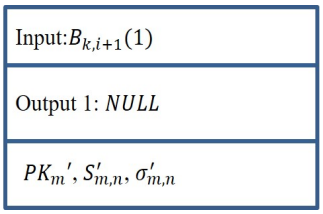



Figure 4.16: Signed revocation block generated by $U_m$

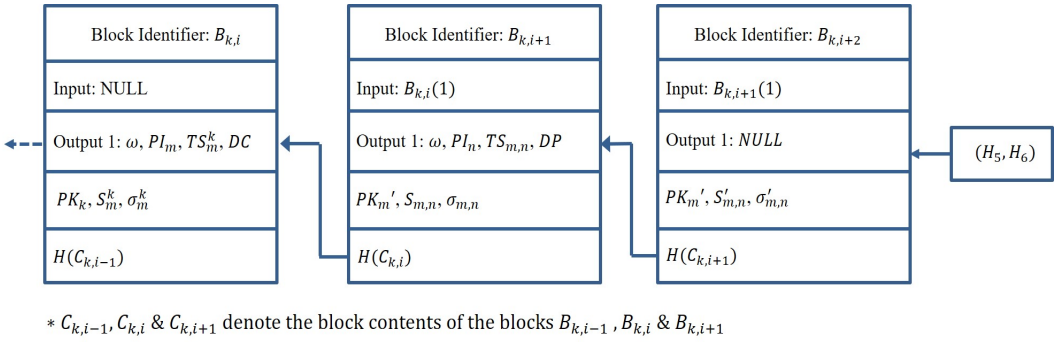



Figure 4.17: Blockchain after inserting the signed revocation transaction block generated by $U_m$

- As described previously in Section 4.6.3, BCC fetches the updated chain header from $AA_k$ via sending the newly accepted revocation block to $AA_k$. If the new block header is denoted by $(H_5, H_6)$, then, $H_5 = H(C_{k,i+2})$ and $H_6 = [H_5]_{SK_k}$ where $C_{k,i+2}$ denotes the block contents of the block $B_{k,i+2}$.
- Finally, BCC validates the received block header and appends it to the blockchain $BC_k$.

### 4.6.5 AC Mechanism

In this subsection, we explain how access decisions are made at the LHP by verifying whether the access requester $U_m$, possesses a set of attributes that satisfy the governing access policy. First, $U_m$ sends an EHR access request to LHP, indicating the $EHR_{id}$, $EHR_{obj}$ and actions intended to be performed on the requested $EHR_{obj}$ along with his pseudo-identity $PI_m$. Upon receiving the request, the PEP of the LHP fetches the associated Boolean statement $\mathcal{T}$ from PR and challenges $U_m$ to mutually authenticate with LHP using the RSA keys associated with the pseudo-identity $PI_m$. If successfully authenticated, LHP examines the blockchains published by BCC to determine whether $U_m$ has a set of attributes that satisfies $\mathcal{T}$.

First, we describe the process followed by the LHP, to validate $U_m$'s ownership of the attribute $\omega$ which is managed by $AA_k$. Given that it is managed by $AA_k$, LHP

examines the blockchain $BC_k$ (starting from the rightmost block) to identify a block with an output vector having $\omega, PI_m$ as the attribute and the receiver of the attribute respectively. If found, LHP checks the input of the block to determine whether it is an assignment transaction or a delegation. In the following, we describe the process associated with the validation of an assigned attribute and a delegated attribute.

**Validating an assigned attribute:** LHP determines the validity of an assigned attribute, if and only if the following conditions are satisfied.

- The timestamp (embedded in the output of the block) is valid.
- The assignment is not been revoked with a revocation transaction. Suppose, the attribute assignment is represented in the first output of the block $B_{k,i}$. To determine the attribute assignment is revoked or not, LHP searches for a block (starting from the block $B_{k,i}$) which is having the input $B_{k,i}(1)$ and an output of NULL. If such block is found, the assignment transaction is deemed as revoked.

**Validating a delegated attribute:** If the block $B_{k,i}$ (which represents the delegation transaction) has an input vector pointing to an output of a different block, then it represents a delegation transaction. The associated procedure for validating a delegation transaction is as follows.

- First, the transaction in $B_{k,i}$ is validated using the two steps used for validating an assigned attribute. This will only provide evidence for the fact that the current delegation to $U_m$ is valid and the user who delegated the attribute to $U_m$ has not revoked it.
- However, still, it is necessary to check whether the delegator still owns the attribute. LHP examines the input of the block $B_{k,i}$, to find out the block that embeds the transaction which shows the delegator's ownership of the attribute $\omega$. Then, the second step associated with validating an assigned attribute is carried out to figure out whether the delegator is revoked from the attribute or not.
- The aforementioned process is continued until an assignment transaction is reached and the same mechanism is followed to check whether the AA who issued the attribute has not revoked the initial attribute assignment.
- If all the above conditions are satisfied, the delegated attribute is deemed to be valid.

According to the protocols mentioned above, LHP evaluates whether the user with pseudo-identity $PI_m$ owns a valid set of attributes (either assigned or delegated) to satisfy the access requirement of the requested $EHR_{obj}$ and thereby makes the decision on provisioning or denying the access.

### 4.6.6 Security Analysis

In this subsection, we intend to show that Scheme 4 is secure against attribute forgery, attribute collusion while providing user privacy by enabling pseudo-anonymity guarantees to users. We use the following hardness problem and the security assumption in the analysis to follow.

**Definition 4 (RSA Problem):** Let $\bar{n} \in \mathbb{Z}^+$ be the product of two distinct odd primes $\bar{p}, \bar{q}$. Let $\bar{e}$ be a randomly chosen positive integer such that $\bar{e} < \phi(\bar{n})$ and $GCD(\bar{e}, \phi(\bar{n})) = 1$ where $\phi(\bar{n}) = (\bar{p} - 1)(\bar{q} - 1)$. Given $(\bar{n}, \bar{e})$ and a randomly chosen $\bar{y} \in \mathbb{Z}^*_{\bar{n}}$, the RSA hardness problem is defined as the computation of $\bar{x}$ such that $\bar{x}^{\bar{e}} \equiv \bar{y} \ mod \ \bar{n}$ [93–95].

**Assumption 3 (RSA Assumption):** There is no probabilistic polynomial-time algorithm which can solve the RSA hardness problem given in Definition 4 with non-negligible probability [93–95].

**Resistance against attribute forgery:** Let us consider an adversary $\mathcal{A}$ intends to forge the attribute $\omega$ with a first level delegation. Further note that $\mathcal{A}$ has the knowledge that the user with pseudo-identity $PI_m$ has already ascertained it from the AA which manages $\omega$. If $\mathcal{A}$ to succeed with forgery of the first level delegation, $\mathcal{A}$ should be able to generate a delegation transaction block and signs it with the secret key associated with the pseudo-identity $PI_m$. This requires a universal forgery (i.e. The ability of an adversary to sign a given message while having only access to the public parameters) of the RSA signature. Given that the RSA signature scheme is universally unforgeable under the RSA assumption, the attribute forgery will be a failure. Although we considered a delegation transaction, every attribute related transaction (assignment, delegation and revocation) is secure against forgery under the RSA assumption, given that the generation of every transaction requires the RSA signature of the entity which generates the transaction.

**Resistance against attribute collusion:** Scheme 4 inherently supports resistance against the collusion of attributes. This is due to the fact that a user will only be able

to prove the ownership of attributes which are associated with the pseudo-identity that he used to mutually authenticate with the LHP.

**Pseudo-anonymity:** In this scheme, users generate their own pseudo-identities which are included in the attribute related transactions that get published in the blockchains. Although users' pseudo-identities are included in plain in the public blockchains, it is not feasible for a third-party to simply relate a given pseudo-identity to the real world identity of the user. Hence, Scheme 4 provisions pseudo-anonymous guarantees to users.

## 4.7 Motivation for Scheme 5

When we consider the proposed solutions in Scheme 3 and Scheme 4, it is evident that both schemes capable of provisioning controlled delegatable access to users. However, Scheme 3 is not capable of providing user anonymity whereas Scheme 4 is restricted with pseudo-anonymity. We have shown that attribute credentials allow us to efficiently construct AC schemes with full anonymity in Section 3.5. Hence, our motivation for Scheme 5 is to construct a delegatable ABAC scheme with better user anonymity using attribute credentials.

Among the existing delegatable credential schemes, the schemes proposed in [96–98], do not associate with attributes. Furthermore, extending them to support multi-level delegation is also not straightforward and will definitely not improve their efficiency [84]. In contrast, the scheme proposed in [84] is the first and the only existing anonymous attribute based credential scheme that supports credential delegations. The main drawback of this construction is that it induces higher end-user computational overhead when disclosing delegated credentials to a verifier. Also, the scheme does not provide control over delegation, meaning that there is no mechanism to control re-delegation of credentials by delegatees.

In Scheme 2, we proposed an anonymous, multi-show unlinkable attribute based credential scheme which exhibits superior performance and lower end-user computational complexity compared to existing multi-show unlinkable credential schemes. In Scheme 5, we extended Scheme 2 to facilitate delegation of attributes via controlled credential delegation, and we show that Scheme 5 has substantially lower end-user computational complexity when disclosing delegated credentials in comparison to the existing attribute based credential schemes with delegatability.

## 4.8 Proposed Scheme 5

We present the functionality of the delegatable credential scheme, by dividing it into seven phases: issuer initialization, credential issuance, credential disclosure and verification, credential delegation, delegated credential disclosure and verification, extending to multi-level credential delegation as well as enforcing user accountability. This section is concluded with the security analysis, performance evaluation of the credential scheme followed by the applicability of the proposed delegatable credential scheme for AS 1. We consider the three entities defined in Scheme 2: issuer $(I)$ which issues credentials, users $(U)$ who ascertain credentials and a verifier $(V)$ which validates the attributes in credentials to describe the proposed delegatable credential scheme.

### 4.8.1 Issuer Initialization

Similar to the initialization of $I$ in Scheme 2, $I$ defines a cyclic group $\mathbb{G}_0$ of prime order $p$ and two cyclic groups $\mathbb{G}_1$, $\mathbb{G}_T$ of prime order $p$ with $q$ being a generator of $\mathbb{G}_1$. A bilinear map $e : \mathbb{G}_0 \times \mathbb{G}_1 \rightarrow \mathbb{G}_T$ is also generated along with a secure hash function $H : \{0,1\}^* \rightarrow \mathbb{Z}_p^*$. Furthermore, $I$ also generates a random secret exponent $\alpha \in \mathbb{Z}_p^*$. Suppose, $A_I$ denotes the set of attribute indices associated with the attributes managed by $I$. Then, $I$ generates exponents $\{\{a_{1j}\}_{j=1,2,...,l_1}, \{a_{2j}\}_{j=1,2,...,l_2}, a_i\}_{i=3,4,...,|A_I|}$, in which $a_{1j}, a_i \in \mathbb{Z}_p^*, a_{2j} \in \mathbb{Z}_p$. The set of values $\{a_{1j}\}_{j=1,2,...,l_1}$ defines the credential expiration values meaning that each credential issued by $I$ includes one expiration value as the first attribute $(a_1)$. Furthermore, $\{a_{2j}\}_{j=1,2,...,l_2}$ denotes the set of values that defines the maximum number of times a particular credential is allowed to be re-delegated. Similarly, during credential issuance, $I$ includes the appropriate $a_{2j}$ exponent as the second attribute $(a_2)$. If $I$ sets $a_2 = 0$, then the relevant credential cannot be delegated further whereas if $a_2 = l, l \in \mathbb{Z}_p^*$ the credential is allowed to be re-delegated at most $l$ times. Finally, $I$ generates a set of generators $\{g_0, g_i\}_{i \in A_I}$ of $\mathbb{G}_0$ and publishes its public tuple $PK_I = \{q, g_0, \{g_i\}_{i \in A_I}, \{\{a_{1j}\}_{j=1,2,...,l_1}, \{a_{2j}\}_{j=1,2,...,l_2}, a_i\}_{i=3,4,...,|A_I|}, q^\alpha, e, H\}$ and keeps $SK_I = \alpha$ as its secret key.

### 4.8.2 Credential Issuance Protocol

Suppose, the user $U_m$ wants to acquire a credential corresponding to the set of attribute indices $A_m$ where $A_m \subseteq A_I$. The protocol is identical to the credential issuance protocol for Scheme 2 given in Section 3.5.2. However, in Scheme 2,

the first attribute exponent $a_1$ in an issued credential corresponds to the credential expiration information whereas, in Scheme 5, the first two attribute exponents $a_1, a_2$ correspond to the credential expiration information and the maximum number of times the credential is permitted to be re-delegated.

### 4.8.3 Credential Disclosure and Verification Protocol

Let us assume that, $U_m$ wants to provide evidence to the verifier $V$ that he owns the attribute set $\{a_i\}_{i \in A_m^R}$, in which $A_m^R$ is the set of indices associated with the disclosing attribute set, where $A_m^R \subseteq A_m$. The protocol involves two phases: Phase 1 and Phase 2, wherein Phase 1 the validity of the credential is examined (whether or not it is issued by $I$, via signature verification) whereas, in Phase 2, a proof is generated to determine whether the user owns the disclosing attribute set. These mechanisms are the same as the credential disclosure and verification protocol for Scheme 2 given in Section 3.5.3.

### 4.8.4 Credential Delegation Protocol

Now, let us see how the credential scheme can be extended to support first level delegation. Suppose, $U_m$ wants to delegate the set of attributes $A_n \subseteq A_m$ to the user $U_n$ ($A_m$ denotes the set of attributes embedded in the credential of $U_m$ issued by $I$). The delegation protocol is divided into three parts as the delegated credential generation, the generation of proof of delegatability (POD) and the process of tagging which links the delegated credential and the POD.

If $U_m$ simply issues a new credential embedding attributes in $A_n$ (as given in Section 3.5.2), the verification upon disclosure of the new credential will not be successful. This is due to the fact that $V$ will not recognize $U_m$ as the authorized entity to issue a credential with the set of attributes $A_n$. To overcome this issue, $U_m$ generates the proof POD which provides evidence that he has a valid credential with attributes $A_n$, and it was issued by $I$. This proof allows $V$ to determine the existence of a valid delegation chain. In addition, we use an additional tag, which helps to tie the delegated credential with the POD, providing a proof that the delegated credential and the POD is constructed by the same entity $U_m$ (delegator). The construction of the POD, delegated credential and the tag are described below.

**Generation of the proof of delegatability (POD):** Let us assume that, $U_m$ has a credential $\{C_m, S_{m,1}, S_{m,2}\}$ associated with the set of attributes $A_m$. Given that $U_m$ wants to delegate the set of attributes $A_n \subseteq A_m$ to $U_n$, $U_m$ generates $\{D', \{a_i\}_{i \in A_m^R},$

$\{\hat{a}_i\}_{i\in A_m/A_m^R}, \hat{s}_m, H(L', N_m)\}$, in which $N_m$ is a nonce generated by $U_m$. Note that $A_m^R = A_n \cup \{a_1, a_2\}$ is the disclosing set of attribute indices while the generation of $D', \hat{s}_m, L'$ is given in Section 3.5.3. $U_m$ must include both $a_1$ and $a_2$ in the disclosure proof, since it provides the information about the validity of the credential $\{C_m, S_{m,1}, S_{m,2}\}$ as well as whether it is permitted to delegate further or not. If the POD issued by $U_m$ to $U_n$ is given by $POD_{m,n}$ then,

$$POD_{m,n} = \{C_m{}^K, S_{m,1}{}^K, S_{m,2}{}^K\}||\{D', \{a_i\}_{i\in A_m^R}, \{\hat{a}_i\}_{i\in A_m/A_m^R}, \hat{s}_m, H(L', N_m)\}|| \{N_m\}_{PK_V}||PK_I \quad (4.5)$$

where $\{C_m{}^K, S_{m,1}{}^K, S_{m,2}{}^K\}$ denotes the randomized credential of $U_m$ as given in (3.8) - (3.10), $\{N_m\}_{PK_V}$ denotes the encryption of the nonce $N_m$ using the public key $PK_V$ of the verifier $V$ and $PK_I$ denotes the public tuple of $I$. Furthermore, $D', \{\hat{a}_i\}_{i\in A_m/A_m^R}, \hat{s}_m, L'$ are as given in (3.14) - (3.18), except for the fact that the nonce $N_0$ should be replaced with the nonce $N_m$.

**Generation of the delegated credential:** To generate the delegated credential, $U_m$ and $U_n$ engages in the credential issuance protocol as mentioned in Section 3.5.2. First, $U_m$ generates two attribute exponents $a_1 \in \mathbb{Z}_p^*$ and $a_2 \in \{0, 1\}$. Among them, $a_1$ denotes the expiration information of the delegated credential, whereas with $a_2$, the delegator can permit ($a_2 = 1$) or deny ($a_2 = 0$) the delegatee to re-delegate the delegated credential. Hence, the set of attributes embedded in the delegated credential is given by $A_n = \{a_1, a_2\} \cup A_n$. If the delegated credential is denoted with $\{C_{mn}, S_{mn,1}, S_{mn,2}\}$, according to Section 3.5.2, the credential public key,

$$C_{mn} = (g_0 \prod_{i\in A_n} {g_i}^{a_i})^{s_n}$$

where $s_n \in \mathbb{Z}_p^*$ is randomly selected by $U_n$, and the credential private key is given by $\beta_n = {s_n}^{-1}$. Furthermore, the signature components $S_{mn,1}, S_{mn,2}$ are generated using $U_m$'s credential private key $\beta_m$ as follows.

$$S_{mn,1} = {C_{mn}}^{\frac{r_n}{\beta_m}}$$

$$S_{mn,2} = {C_{mn}}^{\beta_m - r_n}$$

where $r_n \in \mathbb{Z}_p^*$ is a random identifier generated by the delegator $U_m$.

**Generation of the tag:** Let the tag issued by $U_m$ to $U_n$ is denoted by $T_{m,n}$ then,

$$T_{m,n} = \{q^{\beta_m}, q^{\bar{s}_m}\}$$

where $\beta_m$ is $U_m$'s credential private key, and $\bar{s}_m$ is associated with $\hat{s}_m = \bar{s}_m + D \cdot \beta_m$. This allows the verifier $V$ to use the tag $T_{m,n}$ along with $\hat{s}_m$ in $POD_{m,n}$ to compute a proof that enables him to determine that $POD_{m,n}$ is constructed by the owner of the credential private key $\beta_m$. Finally, $U_m$ will send $\{C_{mn}, S_{mn,1}, S_{mn,2}\}||POD_{m,n}||T_{m,n}$ to $U_n$ to complete the delegation process.

### 4.8.5 Delegated Credential Verification Protocol

Now, we extend the first level delegation scenario from the previous subsection to describe how the user $U_n$ interacts with the verifier $V$ to provide evidence of the ownership of the delegated attributes $A'_n$ via presenting the delegated credential. Let us assume that, $U_n$ wants to disclose the set of attributes $A'_n \subseteq A_n$, where $A_n$ is the set of attributes embedded in the delegated credential $\{C_{mn}, S_{mn,1}, S_{mn,2}\}$.

First, $U_n$ generates the randomized credential $\{C'_{mn}, S'_{mn,1}, S'_{mn,2}\}$ using a randomization exponent $K' \in \mathbb{Z}_p^*$ such that,

$$\begin{aligned} C'_{mn} &= {C_{mn}}^{K'} \\ S'_{mn,1} &= {S_{mn,1}}^{K'} \\ S'_{mn,2} &= {S_{mn,2}}^{K'} \end{aligned}$$

and forwards $\{C'_{mn}, S'_{mn,1}, S'_{mn,2}\}||POD_{m,n}||T_{m,n}$ to $V$. The verification protocol runs in two phases. In Phase 1, $V$ checks whether there exists a valid delegation chain while in Phase 2, $V$ verifies $U_n$'s disclosed set of attributes.

**Phase 1:** According to (4.5) $POD_{m,n}$ received by $V$ is given by,

$$POD_{m,n} = \{{C_m}^K, {S_{m,1}}^K, {S_{m,2}}^K\}||\{D', \{a_i\}_{i \in A_m^R}, \{\hat{a}_i\}_{i \in A_m / A_m^R}, \hat{s}_m, H(L', N_m)\}|| \{N_m\}_{PK_V}||PK_I.$$

First of all, $V$ extracts $\{{C_m}^K, {S_{m,1}}^K, {S_{m,2}}^K\}$ (credential public key and its signature components issued by $I$ to the delegator). Then, using (3.11) - (3.13), $V$ verifies whether $\{{C_m}^K, {S_{m,1}}^K, {S_{m,2}}^K\}$ was issued by $I$ with the help of the public tuple $PK_I$. If validated, $V$ goes ahead with the next stage, which is determining whether the delegator owns the set of attributes which he claims to own through $\{a_i\}_{i \in A_m^R}$ in

$POD_{m,n}$. To determine this, $V$ computes $\hat{L}$ using $\{D', \{a_i\}_{i \in A_m^R}, \{\hat{a}_i\}_{i \in A_m / A_m^R}, \hat{s}_m, H(L', N_m)\}$ given in $POD_{m,n}$ such that,

$$\hat{L} = g_0^{-D} \cdot ({C_m}^K)^{\hat{s}_m} \cdot \prod_{i \in A_m^R} g_i^{-D \cdot a_i} \cdot \prod_{i \in A_m / A_m^R} g_i^{\hat{a}_i}$$

where, $D = H(D', N_m)$. Thereafter, $V$ checks whether,

$$H(\hat{L}, N_m) \stackrel{?}{=} H(L', N_m)$$

to determine the validity of the delegator's set of disclosed attributes. Afterwards, $V$ examines the attribute exponents $a_1, a_2$ to determine the validity of the credential issued from $I$ to the delegator and whether it is allowed to be delegated by the issuer $I$ (i.e. $a_2 \geq 1$).

Upon validation of the disclosed attributes of the credential issued by $I$ to the delegator $\{{C_m}^K, {S_{m,1}}^K, {S_{m,2}}^K\}$, $V$ extracts the randomized credential $\{C'_{mn}, S'_{mn,1}, S'_{mn,2}\}$ and determine its validity by checking whether,

$$e(C'_{mn}, q^{\beta_m}) \stackrel{?}{=} e(S'_{mn,1}, q^{\beta_m}) \cdot e(S'_{mn,2}, q)$$

where $q^{\beta_m}$ is embedded in the tag $T_{m,n}$. Finally, $V$ ends the Phase 1 validation by determining whether $POD_{m,n}$ and the delegated credential $\{C'_{mn}, S'_{mn,1}, S'_{mn,2}\}$ are generated by the same entity. For this, $V$ extracts $\hat{s}_m, D'$ from $POD_{m,n}$ and checks whether,

$$q^{\hat{s}_m} \stackrel{?}{=} q^{\bar{s}_m} \cdot q^{\beta_m \cdot D} \tag{4.6}$$

where $q^{\bar{s}_m}, q^{\beta_m}$ are extracted from the tag $T_{m,n}$. This ensures $V$ regarding the existence of a valid delegation chain. If Phase 1 validation is successful, $V$ proceeds with the Phase 2 validation as follows.

**Phase 2:** $V$ sends a nonce $N_0$ to the delegatee $U_n$, and requests him to disclose the set of attributes $A'_n$ while generating a proof of knowledge of the private key associated with the delegated credential $\beta_n$. This process is identical to the Phase 2 validation in the credential disclosure and verification protocol given in Section 3.5.3. If the proof is valid and the delegated credential is not expired (identified through examining the first attribute exponent of the disclosed set of attributes ($a_1$)), $V$ concludes that the delegatee $U_n$ owns the set of attributes $A'_n$.

### 4.8.6 Multi-level Credential Delegation

In Section 4.8.4 and Section 4.8.5, we presented the credential delegation and delegated credential verification mechanisms with the help of a first level delegation instance. In this subsection, we intend to provide evidence that the proposed scheme enables multi-level (second level and higher) delegations. Consider the following scenario. Suppose, $I$ issues the credential $CR_1 = \{C_1, S_{1,1}, S_{1,2}\}$ to the user $U_1$ for a set of attributes $A_1$, and it is being delegated to $U_L$ after $(L-1)$ number of re-delegations (i.e. $U_1, U_2, ..., U_{L-1}$ are intermediate entities in the delegation chain). Furthermore, assume that $CR_i$ denotes the credential received by the user $U_i$ for the set of attributes $A_i$ where $A_{i+1} \subseteq A_i$.

Then, according to the credential delegation protocol given in Section 4.8.4, $U_L$ should receive the delegated credential $CR_L$ from $U_{L-1}$ along with,

$$\{POD_{1,2}, T_{1,2}\}||...||\{POD_{i,i+1}, T_{i,i+1}\}||...||\{POD_{L-1,L}, T_{L-1,L}\}$$

where $\{POD_{i,i+1}, T_{i,i+1}\}$ denotes the POD and the tag relevant for the credential $CR_i$ received by $U_i$. This allows $U_L$ to provide evidence regarding the existence of a valid chain of $(L-1)$ delegations to the verifier $V$.

During a credential disclosure session, $U_L$ sends the randomized delegated credential $CR'_L$ along with,

$$\{POD_{1,2}, T_{1,2}\}||...||\{POD_{i,i+1}, T_{i,i+1}\}||...||\{POD_{L-1,L}, T_{L-1,L}\}$$

to the verifier $V$. Let us assume that $a_1^i$ denotes the attribute exponent associated with the expiration of credential $CR_i$. Further assume that, the exponent $a_2^1$ denotes the attribute exponent associated with the maximum permissible number of re-delegations defined in $CR_1$ whereas $a_2^i$ when $i = 2, 3, ..., L$ denotes the attribute exponent which specifies whether the receiver of the credential $CR_i$ have the permission to re-delegate the attributes embedded in $CR_i$. $V$ proceeds with verification using the delegated credential verification protocol given in Section 4.8.5 as follows.

- First of all, $V$ uses $\{POD_{1,2}, T_{1,2}\}$ and examine the validity of $CR_1$ and $U_1$'s ownership of the set of attributes $A_2$ ($U_1$ delegates $A_2$ to $U_2$ with $CR_2$) using the Phase 1 validation mechanism given in Section 4.8.5. Note that validating $CR_1$ includes both examining the attribute exponent $a_1^1$ to determine whether $CR_1$ is expired or not as well checking $a_2^1 \geq (L-1)$ to determine whether the number of delegations is within the permissible limits. In addition, $V$ uses $\{POD_{1,2}, T_{1,2}\}$ and $\{POD_{2,3}, T_{2,3}\}$ in accordance with (4.6) to determine

whether the receiver of $CR_1$ is the delegator of $CR_2$.

- Then, $V$ validates each delegated credential $CR_i$ ($i > 1$) and $U_i$'s ownership of the set of attributes $A_{i+1}$ using $\{POD_{i,i+1}, T_{i,i+1}\}$ by following the mechanism given in the above step. $V$ determines the validity of each delegated credential $CR_i$ with the help of the attribute exponent $a_1^i$. In addition, $V$ checks whether $a_2^i = 1$ (to determine whether $U_i$ has the permission to further delegate the attributes in $CR_i$) as well as $A_{i+1} \subseteq A_i$ is held or not. As mentioned in the above step, $V$ also ensures whether the receiver of $CR_i$ is the delegator of $CR_{i+1}$, with the help of (4.6) in Phase 1 validation given in Section 4.8.5.

- If all the aforementioned verifications are successful, $V$ realizes the existence of a valid chain of delegations, and requests $U_L$ to disclose the required attributes in $CR'_L$ by running the Phase 2 validation given in Section 4.8.5 with $U_L$.

### 4.8.7 Enforcing User Accountability

It is possible to simply extend the mechanism used to revoke the anonymity of credentials in Scheme 2 for revoking anonymity of credentials in the delegatable credential scheme. To facilitate this, each delegator should also have a table similar to $I$ (as mentioned in Section 3.5.4) when delegating credentials. Then, to revoke the anonymity of a given delegated credential, $V$ must send the randomized credential and the set of PODs and tags received by $V$ during the execution of a delegated credential verification protocol. Given that the first POD corresponds to the issuance of the credential by $I$, $I$ use the information in this POD to reveal the identity of the first user (who received the credential in question from $I$) using the mechanism described in Section 3.5.4. Thereafter, $I$ can contact the first user in the chain and request him to reveal his delegatee (using the same mechanism) and so on, until all the entities in the delegation chain are revealed.

### 4.8.8 Security Analysis

In this subsection, our intention is to show that the proposed delegatable credential scheme exhibits resistance against attribute forgery via credential unforgeability and resistance against replay attacks.

**Credential unforgeability:** The credential issuance protocol used in Scheme 5 is identical to the credential issuance protocol of Scheme 2. Hence, credentials are unforgeable under the DDH assumption, as we have shown in Section 3.5.5.

**Resistance against replay attacks:** We consider two possible scenarios, where an adversary $\mathcal{A}$ replays proofs of non-delegated credentials and delegated credentials to prove the ownership of attributes. Given that the case of replaying non-delegated credential proofs resembles the replay attack scenario discussed in relation to Scheme 2, we refer the readers to Section 3.5.5 for the details.

Now, let us consider the scenario where the adversary $\mathcal{A}$ listened to a delegated credential verification session between $U_L$ and $V$ where $U_L$ uses the $(L-1)$ level delegated credential $CR_L$ (as given in Section 4.8.6) to prove the ownership of a set of delegated attributes. Through listening to the delegated credential verification protocol between $U_L$ and $V$, $\mathcal{A}$ learns the randomized credential $CR'_L$, the set of PODs and tags $PT$ such that,

$$PT = \{POD_{1,2}, T_{1,2}\}||...||\{POD_{i,i+1}, T_{i,i+1}\}||...||\{POD_{L-1,L}, T_{L-1,L}\}$$

as well as the proof $P$,

$$P = \{D', \{a_i\}_{i \in A_L^R}, \{\hat{a}_i\}_{i \in A_L / A_L^R}, \hat{s}_L, H(L', N_0)\}.$$

$P$ is the proof generated when disclosing the set of attributes $A_L^R$ during Phase 2 validation using the nonce $N_0$ issued by $V$. If the adversary $\mathcal{A}$ establishes a session with $V$ and mounts a replay attack by replaying $CR'_L$ and $PT$, $V$ will carry out the Phase 1 of the delegated credential verification protocol and finds out that there exists a valid chain of delegations. This is due to the fact that PODs are non-interactive proofs. This will trigger the Phase 2 validation and $V$ requests for a proof with disclosing the set of attributes $A_L^R$ using a new nonce $N_1$. Given that $\mathcal{A}$ has no knowledge of the private key associated with the $U_L$'s credential $CR_L$, $\mathcal{A}$ will not be able to construct a valid proof using $N_1$.

### 4.8.9 Performance Evaluation

In this subsection, we compare the end-user computational complexity of the proposed delegatable credential scheme with the only existing attribute based delegatable credential scheme proposed in [84]. For the comparison, we have tabulated the exponentiation and pairing count (associated with end-user computations) when

Table 4.1: Comparison of end-user computational complexity associated with delegated credential disclosure and verification of the proposed credential scheme with existing delegatable credential schemes

| Scheme | Disclosure of a $L^{th}$ level credential | | Delegatability |
|---|---|---|---|
| | $G_{EC}$ | $e$ | |
| Proposed Scheme 5 | $h_L + 5$ | 0 | Controlled |
| [84] | $\sum_{i=0}^{L}(3 \cdot n_i + 7)$ | 0 | No control |

disclosing a $L^{th}$ level delegated credential for verification in Table 4.1. The columns $G_{EC}$ and $e$ show the amount of exponentiations in elliptic curves and the number of pairing operations a user has to compute during the execution of the delegated credential disclosure and verification protocol. The notation $n_i$ denotes the number of attributes in an $i^{th}$ level credential whereas the notations $d_i, h_i$ denote the number of disclosed and undisclosed attributes in a credential of level $i$ ($n_i = d_i + h_i$).

In [84], it is observable that the end-user computational complexity for disclosing a delegated credential increases with the length of the delegation chain. In contrast, in the proposed scheme, this complexity parameter is independent of the length of the delegation chain. Hence, our scheme performs better in relation to disclosure of delegated credentials while enforcing controlled delegation which is not the case with [84].

### 4.8.10 Applying the Delegatable Credential Scheme for AS 1

The application of the proposed delegatable credential scheme for AS 1 can be achieved as the same way of applying the anonymous credential scheme (proposed in Scheme 2) for AS 1, as described in Section 3.5.7. The only exception is that users will be able to commit one or more attribute credentials (non-delegated or delegated credentials) while disclosing the minimal set of attributes to LHP that is sufficient to satisfy the $\mathcal{T}$ associated with the $EHR_{obj}$. Then, PEP of LHP can verify each committed credential using credential verification protocols presented in Section 4.8.3 and Section 4.8.6 (depending on whether the credential is a non-delegated or a delegated credential) to determine whether the user owns a valid set of attributes that enable the access to the requested $EHR_{obj}$.

## 4.9 Chapter Summary

In this chapter, we have proposed three ABAC schemes (Scheme 3, Scheme 4 and Scheme 5) integrated with multi-level access delegation capabilities to meet the access demands associated with AS 1. All three proposed schemes enable con-

trolled access delegation through allowing the delegating users to control further re-delegations by their delegatees as well as bounding the maximum number of possible re-delegations (length of the delegation chain).

Among the proposed schemes, Scheme 3 is resistant against attribute forgery, attribute collusion and replay attacks. It is also equipped with an attribute revocation mechanism. However, this mechanism imparts high computational overhead when revoking a specific attribute from a user, since this requires issuing new attribute tokens to all users who share the revoking attribute except the user to be revoked. In addition, Scheme 3 induces higher end-user key management overhead, given that a user needs to manage a separate secret key for each of the attributes that he is delegating to others (delegating user acts as a virtual AA). In Scheme 4, we have addressed the aforementioned drawbacks via proposing a delegatable AC mechanism using blockchain technology. Given that all attribute related transactions are recorded in a blockchain, revocation can simply be achieved through submitting a signed revocation transaction. Furthermore, Scheme 4 has the least end-user key management overhead, since each user needs to store only a single secret key and it is independent of whether the user is involved in the delegation or not. However, Scheme 4 does not support selective disclosure of attributes given that the blockchains reveal all the attributes associated with the user's pseudo-identity. We constructed Scheme 5 by extending the anonymous credential scheme proposed in Scheme 2 to facilitate controlled delegation capabilities. Hence, Scheme 5 enforces full access anonymity, in comparison to Scheme 3 (which does not support access anonymity) and Scheme 4 (which only guarantees pseudo-anonymity) while allowing users to selectively disclose their attributes. Furthermore, we have shown that the delegatable credential scheme proposed in Scheme 5 is secure against attribute forgery, replay attacks and has a superior performance in comparison to the existing anonymous credential schemes with delegatability. Given that Scheme 5 is extended from Scheme 2, there is a possibility of colluding credentials of different entities as mentioned in Section 3.6 and we plan to address this in our future work.

# Chapter 5

# Anonymous ABAC Scheme for AS 2

*In this chapter, we present an anonymous ABAC scheme (Scheme 6) compatible with the access demands associated with AS 2. This scheme uses ABE as the underlying AC mechanism, and therefore, we start the chapter by outlining the motivations for utilizing ABE. Then, we provide a summary of the existing research efforts which propose ABE based AC mechanisms for similar health information sharing applications. Afterwards, the proposed Scheme 6 is presented in detail before the chapter is concluded with a summary. Scheme 6 is based on our work in Paper 7 and Paper 8.*

## 5.1 Motivation for Utilizing ABE

The recent advancements in cloud computing have been a significant factor in influencing healthcare providers to store EHRs in third-party, semi-trusted cloud platforms instead of storing them locally. This approach potentially leads to the better availability of health data as well as relieving the care providers from the burden of maintaining them. However, as we have elaborated in Section 1.1, outsourcing of health information lifts the care providers' control over the data; hence encryption of data to preserve the confidentiality is of paramount importance for preventing insider attacks. Furthermore, the encryption has to be one-to-many given that a particular record could potentially need to be shared among a group of users to enable access flexibility.

ABE [46, 99–105] is a promising mechanism to enforce access flexibility while preserving the data confidentiality since it allows EHR data to be encrypted by the care provider according to an access policy which determines the potential users

who are eligible to access before being stored in a cloud infrastructure. Thus, a user who is having a set of valid attributes that satisfy the governing access policy could potentially decrypt the encrypted EHR data anonymously, with the associated attribute keys upon downloading the relevant ciphertexts from the cloud. ABE schemes can be divided into two categories based on their functionality, as key-policy attribute based encryption (KP-ABE) schemes [46, 99] and ciphertext-policy attribute based encryption (CP-ABE) schemes [100, 101, 103, 104, 106]. In a KP-ABE scheme, the ciphertext is associated with a set of attributes and users' secret keys are encoded with attribute based access structures [46]. If the access structure associated with a user's secret key satisfies the set of attributes which is used to generate a specific ciphertext, the user will be able to decrypt the ciphertext with the help of his secret keys. CP-ABE can be considered as the dual of KP-ABE, where the ciphertext is encoded with the access structure while the users' secret keys are encoded with attributes [105]. In relation to an e-health data sharing application, CP-ABE schemes are more conducive compared to KP-ABE schemes, given that the data owner (healthcare provider) will be able to specify the set of recipients through an attribute based access structure while a user who possesses a set of attributes that satisfies the access structure could potentially decrypt the encrypted data using the relevant secret keys.

## 5.2 Related Work

In this section, we summarize the most prominent existing research work on utilizing CP-ABE methods for secure sharing of health information while discussing associated weaknesses of the considered solutions.

The personal health information sharing scheme in [107] used the CP-ABE scheme in [106] while introducing the concept of personal and public domains. The solution consists with a trusted authority (TA) for managing attributes in the public domain while the data owner or the patient himself acting as the TA for the personal domain for the purpose of issuing attributes relevant for the personal domain. Thus, the data owner can encrypt the private health data using an attribute based access structure, allowing only the users who have attributes that satisfy the associated access structure can successfully decrypt the data. However, the main issue is the use of a single TA for administrating the user attributes of the public domain. This approach could not only lead to a single point of failure but also may cause key-escrow problems given the fact that the TA can access all the encrypted data. Besides, the adoption of a single TA for managing all attributes in the public domain may also not be a realistic assumption with respect to an e-health environ-

ment which is (generally) inherently distributed. For instance, consider a scenario where a patient's health record requires to be encoded with attributes belonging to two healthcare providers. In such a situation, it is not realistic to assume that the attributes related to both organizations are handled by the same central TA, while it is more realistic to think of a scenario where each organization acts as an AA to issue own attributes. We have noticed that cloud based personal health information sharing schemes for a similar setting but with a central TA are proposed in [108–112].

In the quest for dealing with the above stated issue, authors in [23] proposed an ABE based health information sharing scheme using multiple authorities such that each authority administrates a disjoint set of attributes. Thus, users belonging to the public domain can ascertain required attributes from the relevant AA while users in the private domain ascertain the attributes from the data owner similar to [107]. In this solution, the authors have utilized the multi-authority attribute based encryption (MA-ABE) scheme proposed in [100] to achieve the secure sharing of health records. However, the main drawback of this MA-ABE scheme is that it requires users to obtain at least one attribute from each AA for the proper functioning of the encryption scheme.

In order to realize a flexible cloud based EHR sharing scheme, it is necessary to utilize a flexible and scalable multi-authority ABE scheme. In Scheme 6, we address this issue via constructing a distributed multi-authority CP-ABE scheme (MA-CP-ABE) influenced by the single authority CP-ABE scheme proposed in [106].

## 5.3 Security and Privacy Requirements

We require the following security and privacy requirements to be satisfied in the proposed Scheme 6.

- *Confidentiality of EHR data*: EHR data of patients must be kept secret from unauthorized parties. Hence, we require the proposed encryption scheme to be IND-CPA secure (indistinguishable under chosen plaintext attacks).

- *Resistance against attribute collusion*: Multiple users should not be able to collude their attributes and decrypt EHR data.

- *Attribute revocation*: Whenever an attribute of a certain user is no longer valid, the user should not be able to decrypt EHR data using the secret keys associated with the revoked attribute.

- *Patient privacy:* To ensure patient privacy, any user who does not possess enough attributes to satisfy the access requirements associated with EHRs of patients must be prevented from accessing them. The property of EHR data confidentiality along with resistance against attribute collusion contributes to ensuring the patient privacy.

- *User privacy:* Users should be able to gain access to EHR data while preserving the unlinkability between the identity of users and data.

## 5.4 Proposed Scheme 6

In this section, we describe Scheme 6, starting with an overview, five phases of the proposed scheme: system initialization, key distribution, EHR encryption, EHR decryption and attribute revocation followed by the security analysis and the performance evaluation of Scheme 6.

### 5.4.1 Overview of Scheme 6

We refer to the system model corresponding to AS 2 illustrated in Figure 2.3. In this model, each AA manages a disjoint set of attributes and issues attributes for the users upon validating their eligibility. Furthermore, it is assumed that the secret attribute keys are securely handed over to the corresponding user. To outsource an $EHR_{obj}$ to HC, LHP first constructs the access structure $\mathcal{T}$. Note that the attributes in $\mathcal{T}$ can have a combination of attributes from one or more AAs. Then, the $EHR_{obj}$ is encrypted with the help of public attribute keys corresponding to the attributes in $\mathcal{T}$ (details will be given in Section 5.4.4). The generated ciphertext along with $\mathcal{T}$ is sent to the HC to be stored. When a user is required to access a specific $EHR_{obj}$, an EHR access request is sent to the HC indicating the $EHR_{id}$ and the relevant $EHR_{obj}$ information. The user will only be able to decrypt the encrypted $EHR_{obj}$, if and only if the user has secret keys corresponding to a set of attributes that satisfy the $\mathcal{T}$ associated with the encrypted $EHR_{obj}$.

### 5.4.2 System Initialization

To globally initialize the system, similar to Scheme 1, AAs agree on two multiplicative cyclic groups $\mathbb{G}_0$, $\mathbb{G}_1$ of prime order $p$ with $g$ being a generator of $\mathbb{G}_0$ and a bilinear map $e : \mathbb{G}_0 \times \mathbb{G}_0 \rightarrow \mathbb{G}_1$ along with a secure hash function $H : \{0,1\}^* \rightarrow \mathbb{Z}_p^*$ that maps each user identity string to a unique value in $\mathbb{Z}_p^*$. The user identity has

to be unique, such as the public key corresponding to the user's PKI certificate. Also, AAs agree on a shared secret $\xi \in \mathbb{Z}_p^*$. Then, each AA publishes the set of global public parameters $(\mathbb{G}_0, \mathbb{G}_1, H, e, g, p)$. This allows any new AA to be simply globally initialized by acquiring the set of global parameters as well as the shared secret $\xi$ which are shared by the existing AAs. After the sharing of global parameters, each AA should be locally initialized, and the initialization procedure is described below by considering $AA_k$.

- $AA_k$ chooses two random exponents $\alpha_k, \beta_k \in \mathbb{Z}_p^*$ and computes $X_k = g^{\beta_k}$, $Y_k = e(g,g)^{\alpha_k}$.
- Let us assume that $AA_k$ is responsible for the attribute set $A^k$. Then, a unique identifier $t_{k,i} \in \mathbb{Z}_p^*$ for each element $i$ in $A^k$ is also randomly selected. Furthermore, each attribute administered by $AA_k$ is also bound with a public attribute key $T_{k,i}$, where $T_{k,i} = g^{t_{k,i}}$.
- $AA_k$ will keep $\{\xi, \alpha_k, \beta_k, t_{k,i}\}_{i=1,2,\ldots,|A^k|}$ as the master secret set $MK_k$ and publish $\{X_k, Y_k, T_{k,i}\}_{i=1,2,\ldots,|A^k|}$ as its public tuple $PK_k$.

### 5.4.3 Attribute Key Distribution

Let us assume that, the user $U_m$ wants to acquire attribute keys for the set of attributes $A_m$. In addition, assume that $A_m^k \subseteq A_m$ denotes the subset of attributes in $A_m$ which should be acquired from $AA_k$. Suppose that $AA_k$ has already validated the eligibility of $U_m$ for ascertaining the requested attributes. The process of attribute key distribution is as follows.

- $AA_k$ first uses the hash function $H$ to map the identity of $U_m$ to a unique identifier $\bar{r}_m \in \mathbb{Z}_p^*$ and computes $r_m = \bar{r}_m + \xi$.
- Then, a secret key for each requesting attribute is generated as described below. If the secret key set is denoted by $SK_m^k$,

$$SK_m^k = \{sk_0^k, sk_i^k\}_{i=1,2,\ldots,|A_m^k|}$$

and,

$$sk_0^k = g^{\frac{\alpha_k - r_m}{\beta_k}} \tag{5.1}$$

$$sk_i^k = g^{\frac{r_m}{t_{k,i}}} \tag{5.2}$$

where $t_{k,i}$ is the secret exponent of the $i^{th}$ attribute in $A_m^k$. Note that the secret key component $sk_0^k$ relates the user identity to the identity of the issuing authority $AA_k$ whereas the secret key component $sk_i^k$ relates the user identity to the $i^{th}$ attribute in $A_m^k$.

- Finally, $SK_m^k$ is securely transferred to $U_m$.

### 5.4.4 EHR Encryption

Let us assume that LHP wants to encrypt EHR data $M \in \mathbb{G}_1$, which corresponds to the $EHR_{obj} = O$, of a given patient's EHR $I$. First, LHP generates the access structure $\mathcal{T}$ and deduces a set of access sub-structures $\{\mathcal{T}_k\}_{k=1,2,...,q}$ as explained in Section 2.1.3. Thus, the ciphertext of $M$ encrypted with $\mathcal{T}$ is given by,

$$E(M) = (\mathcal{T}, \{E_k\}_{k=1,2...,q})$$

where $E_k$ denotes the ciphertext of $M$ encrypted with the access sub-structure $\mathcal{T}_k$. The process of computing $E_k$ is described below. Furthermore, we have illustrated the structure of the ciphertext $E(M)$ in Figure 5.1.

Let us assume that $k^{th}$ sub-structure $\mathcal{T}_k$ contains $s$ attributes and they are managed by $l$ AAs such that, $l \leq N$, where $N$ is the total number of AAs in the system. Note that any AA may manage more than one attribute of the considered $s$ attributes. Then, we can represent the ciphertext $E_k$ using the ciphertext components $C_0$, $\{C_i'\}_{i=1,2,...,l}$ and $\{C_i''\}_{i=1,2,...,s}$, such that,

$$E_k = (\mathcal{T}_k, C_0, \{C_i'\}_{i=1,2,...,l}, \{C_i''\}_{i=1,2,...,s}).$$

Ciphertext components of $E_k$ are computed as follows. LHP first generates a random exponent $h \in \mathbb{Z}_p^*$ and using the public keys of $l$ AAs, it computes the ciphertext components $C_0$ and $\{C_i'\}_{i=1,2,...,l}$ such that,

$$C_0 = M \prod_{i=1}^{l} Y_i^h = M \cdot e(g,g)^{h \sum_{i=1}^{l} \alpha_i} \tag{5.3}$$

$$C_i' = X_i^h = g^{\beta_i h}. \tag{5.4}$$

To compute $\{C_i''\}_{i=1,2,...,s}$, a secret share of $h$ is assigned for each attribute in $\mathcal{T}_k$ by following the steps given below.

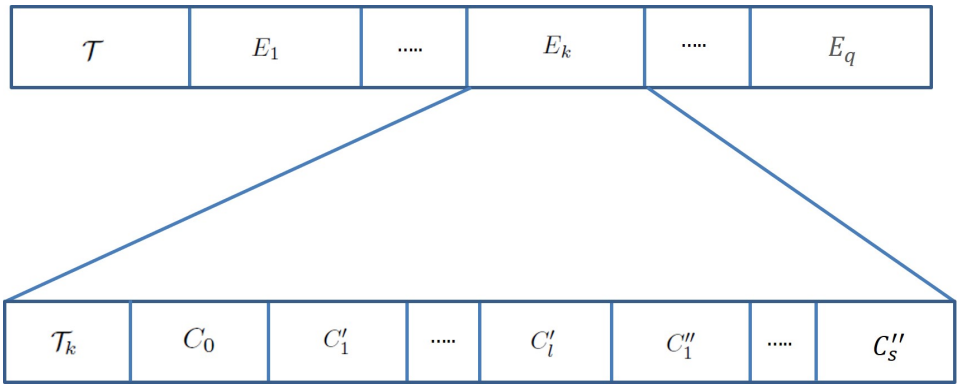



Figure 5.1: Structure of the ciphertext $E(M)$ in Scheme 6

- For each attribute in $\mathcal{T}_k$ except the last, a random exponent $h_i \in \mathbb{Z}_p^*$ is assigned while the last element is assigned the value equals to $l \cdot h - \sum_{i=1}^{s-1} h_i$.
- Then, the LHP computes $\{C''_i\}_{i=1,2,...,s}$ such that,

$$C''_i = T_i^{h_i} = g^{t_i h_i} \tag{5.5}$$

where $T_i$ corresponds to the public attribute key of the $i^{th}$ attribute in $\mathcal{T}_k$.

Similarly, LHP generates ciphertexts relevant for all the sub-structures of $\mathcal{T}$ and uploads $E(M)$ along with the EHR identification $I$ and object information $O$ to HC.

### 5.4.5 EHR Decryption

Suppose, $U_m$ wants to access the $EHR_{obj}$, $O$ associated with the EHR, $I$ stored in the HC. $U_m$ should first send an access request indicating the $EHR_{id}$ and $EHR_{obj}$ information corresponding to the access required object to HC. Then, HC fetches the corresponding $\mathcal{T}$ associated with the requested $EHR_{obj}$ and sends it back to $U_m$. Afterwards, $U_m$ determines the smallest subset of attributes $A'_m \subseteq A_m$ that satisfies the received $\mathcal{T}$. Based on $A'_m$, $U_m$ generates a sub-structure $\mathcal{T}'$ and sends it to the HC. According to the received $\mathcal{T}'$, HC fetches the corresponding EHR ciphertext $E'$ and sends it back to $U_m$ which enables him to decrypt the encrypted data using the relevant attribute secret keys. The decryption process is as follows.

For illustrative purposes, let us assume that the received ciphertext $E'$ is encoded with $s$ attributes which are administered by $l$ AAs. Then,

$$E' = (\mathcal{T}', C_0, \{C'_i\}_{i=1,2,...,l}, \{C''_i\}_{i=1,2,...,s})$$

where $C_0$, $C'_i$ and $C''_i$ are given in (5.3) - (5.5) respectively. Further, assume that $\{sk_i\}_{i=1,2...,s}$ denotes the relevant attribute secret key set owned by $U_m$ which relates

the identity of $U_m$ to the attributes in $A'_m$, and $\{sk_0^i\}_{i=1,2,...,l}$ refers to the set of secret key components which relates the identity of $U_m$ to the $l$ AAs who issued the $s$ attributes. According to (5.1) and (5.2), $sk_i = g^{r_m/t_i}$ and $sk_0^i = g^{(\alpha_i - r_m)/\beta_i}$, in which $t_i$ denotes the secret attribute exponent of the $i^{th}$ attribute in $A'_m$. Then, $U_m$ will be able to decrypt $E'$ and discover the EHR data $M$ as follows.

$$\frac{C_0}{\prod_{i=1}^{s} e(C''_i, sk_i) \cdot \prod_{i=1}^{l} e(C'_i, sk_0^i)} = \frac{M \cdot e(g,g)^{h\sum_{i=1}^{l}\alpha_i}}{e(g,g)^{h\sum_{i=1}^{l}\alpha_i}} = M.$$

### 5.4.6 Attribute Revocation Mechanism

In Scheme 6, the revocation process is handled by the AA which is responsible for the attribute to be revoked. We present the revocation process as follows. Suppose, $AA_k$ requires to revoke the attribute $\omega$ from $U_m$. In addition, assume that the secret exponent associated with the attribute $\omega$ defined by $AA_k$ is given by $t_\omega$.

- First of all, a new random secret exponent $t'_\omega$ for the attribute to be revoked $\omega$ is selected and based on the new secret, the associated public attribute key $g^{t'_\omega}$ is generated and published.
- According to (5.2), it is evident that modification to the secret attribute exponent of a given attribute affects the secret keys associated with the considered attribute. Hence, the relevant secret keys need to be updated accordingly. Therefore, new secret keys are generated (using the new secret exponent $t'_\omega$) and sent to the users who obtained the attribute $\omega$ previously except the user to be revoked ($U_m$).
- Given that the public attribute key related to the revoked attribute is modified, data encrypted with the attribute $\omega$ will be contaminated. We elaborate this further through the following example.

Consider the encryption of a message $M$, with a sub-structure $\mathcal{T}_1 = \omega$. Let us assume that $E(M)$ represents the encryption of $M$ prior to the revocation of attribute $\omega$. Then, according to Section 5.4.4, $E(M) = (\mathcal{T}_1, C_0, C', C'')$, where $C_0 = M \cdot Y_k^h$, $C' = X_k^h$ and $C'' = g^{t_\omega h}$. Note that the alteration of the public attribute key of the attribute $\omega$ will only contaminate the ciphertext component $C''$ given that $C_0, C'$ ciphertext components are independent of the public attribute key of the attribute $\omega$ (revoking attribute). Hence, we use a re-encryption mechanism to update the contaminated ciphertext as follows.

- $AA_k$ first generates the re-encryption key $RE_{key} = {}^{t'_\omega}/_{t_\omega}$.
- Then, $AA_k$ sends $RE_{key}$ to HC which enables the HC to re-encrypt the contaminated ciphertext components. If the corresponding updated ciphertext component is given by $C''_{new}$ then,

$$C''_{new} = C''^{RE_{key}} = g^{t_\omega h \frac{t'_\omega}{t_\omega}} = g^{t'_\omega h}.$$

- Thus, the ciphertext corresponding to the encryption of $M$ after the revocation is given by $E_{new}(M)$, where,

$$E_{new}(M) = (\mathcal{T}_1, C_0, C', C''_{new}).$$

After the revocation, the revoked user ($U_m$) will not be able to use his old secret key corresponding to the attribute $\omega$ since the public attribute key related to the attribute $\omega$ is already modified. Given that other users who have ascertained the attribute $\omega$ from $AA_k$ are issued with new secret keys, they will be able to use the new secret keys for future access sessions.

### 5.4.7 Security Analysis

In this subsection, we provide evidence for the fact that the MA-CP-ABE construction exhibits IND-CPA security while being able to resist the attacks mounted via attribute collusion. First, we introduce the following hardness problem and the security assumption on which the IND-CPA security is based upon.

**Definition 5 (Decisional Bilinear Diffie-Hellman (DBDH) Assumption):** Suppose $\mathbb{G}_0$, $\mathbb{G}_1$ are cyclic groups of order $p$. Let $g$ be a generator of $\mathbb{G}_0$ and $e : \mathbb{G}_0 \times \mathbb{G}_0 \rightarrow \mathbb{G}_1$ be a bilinear mapping function. Given that $a, b, c, z \in \mathbb{Z}_p^*$ are randomly chosen, the DBDH problem is to distinguish the tuple $(g, g^a, g^b, g^c, e(g,g)^{abc})$ from the tuple $(g, g^a, g^b, g^c, e(g,g)^z)$.

**Assumption 4 (DBDH Assumption):** Suppose $\mathbb{G}_0$, $\mathbb{G}_1$ are cyclic groups of order $p$. Let $g$ be a generator of $\mathbb{G}_0$ and $e : \mathbb{G}_0 \times \mathbb{G}_0 \rightarrow \mathbb{G}_1$ be a bilinear mapping function. Given that $a, b, c, z \in \mathbb{Z}_p^*$ are randomly chosen, there is no polynomial-time adversary that can distinguish the tuple $(g, g^a, g^b, g^c, e(g,g)^{abc})$ from the tuple $(g, g^a, g^b, g^c, e(g,g)^z)$ with non-negligible probability.

**Resistance against chosen plaintext attacks:** We define the model for IND-CPA security, with the help of the following game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$, where $\mathcal{C}$ simulates the protocol and answer queries from $\mathcal{A}$.

- **Init:** The adversary $\mathcal{A}$ selects the challenge access structure $\mathcal{T}^*$ and gives it to the challenger $\mathcal{C}$.
- **Setup:** $\mathcal{C}$ generates the master secret $MK_{\mathcal{C}}$ and the public tuple $PK_{\mathcal{C}}$, and $PK_{\mathcal{C}}$ is sent to $\mathcal{A}$.
- **Phase 1:** $\mathcal{A}$ sends attribute key requests to $\mathcal{C}$ for the attributes which are not elements in $\mathcal{T}^*$ and $\mathcal{C}$ responds with relevant secret keys.
- **Challenge:** $\mathcal{A}$ sends two messages $M_0, M_1$ to the challenger $\mathcal{C}$. $\mathcal{C}$ picks a random bit $v \in \{0, 1\}$ and generates the ciphertext $E_v$ using attributes in $\mathcal{T}^*$. Then, the ciphertext $E_v$ is forwarded to the adversary $\mathcal{A}$.
- **Phase 2:** Phase 1 is repeated.
- **Guess:** The adversary $\mathcal{A}$ outputs a guess $v' \in \{0, 1\}$.

The advantage of the adversary $\mathcal{A}$ in the aforementioned game is defined as,

$$\epsilon = |Pr[v' = v] - \frac{1}{2}|. \tag{5.6}$$

Our intention is to demonstrate that the proposed MA-CP-ABE scheme is IND-CPA secure, given that the DBDH assumption is held. To prove this, we show that if an adversary $\mathcal{A}$ can win the IND-CPA security game with an advantage $\epsilon$, it is possible to use this adversary $\mathcal{A}$ to build a simulator $\mathcal{S}$ that can solve the DBDH hardness problem with an advantage of $\epsilon/2$.

First of all, the challenger $\mathcal{C}$ generates the two cyclic groups $\mathbb{G}_0$ and $\mathbb{G}_1$ with $g$ being a generator of $\mathbb{G}_0$, an efficiently computable bilinear mapping function $e : \mathbb{G}_0 \times \mathbb{G}_0 \rightarrow \mathbb{G}_1$ and a set of random exponents $a, b, c, z \in \mathbb{Z}_p^*$. Then, the challenger $\mathcal{C}$ feds a DBDH instance $(g, g^a, g^b, g^c, R_\delta)$ to the simulator $\mathcal{S}$ in which $R_\delta$ is set through flipping a fair coin $\delta$ where,

$$R_\delta = \begin{cases} e(g,g)^{abc} & \text{if } \delta = 0 \\ e(g,g)^z & \text{otherwise.} \end{cases}$$

The simulator $\mathcal{S}$ acts as the adversary $\mathcal{A}$'s challenger and the game proceeds as follows.

**Init:** $\mathcal{A}$ selects a challenge access sub-structure $\mathcal{T}^*$ (which contains $s$ attributes from $l$ out of $N$ AAs) and gives it to $\mathcal{S}$. Note that we denote the attribute set in $\mathcal{T}^*$ with $A_{\mathcal{T}^*}$.

**Setup:** We assume that, the simulator $\mathcal{S}$ simulates on-behalf of all $N$ AAs. For each attribute $i \in A_{\mathcal{T}^*}$, the simulator $\mathcal{S}$ chooses a random element $q_i \in \mathbb{Z}_p^*$ and thereby sets the public attribute key for each above stated attribute as $T_i = g^{q_i}$. For all the other attributes ($i \notin A_{\mathcal{T}^*}$), the simulator $\mathcal{S}$ sets $T_i = g^{b/q_i}$. Furthermore, $\mathcal{S}$ selects a set of random exponents $\{\gamma_i, \beta_i\}_{i=1,2,..,N} \in \mathbb{Z}_p^*$ and by allowing, $e(g,g)^{\alpha_i} = e(g,g)^{ab/l} e(g,g)^{\gamma_i}$, $\mathcal{S}$ implicitly sets each AA's secret key $\alpha_i = \frac{ab}{l} + \gamma_i$. Then, the public parameters of the simulator are forwarded to $\mathcal{A}$.

**Phase 1**: The adversary $\mathcal{A}$ sends attribute key requests to the simulator $\mathcal{S}$ for a set of attributes such that each attribute $i \notin A_{\mathcal{T}^*}$. For the adversary $\mathcal{A}$, the simulator $\mathcal{S}$ generates the secret key $sk_0^i$ which relates the identity of the issuing authority and the identity of the adversary as,

$$sk_0^i = g^{\frac{\gamma_i - \hat{r}b}{\beta_i}} = g^{\frac{\alpha_i - (\hat{r}b + \frac{ab}{l})}{\beta_i}}$$

where $\hat{r} \in \mathbb{Z}_p^*$. Then, $\mathcal{S}$ should generate the attribute secret key $sk_i$ corresponding to the each requested attribute. To have a valid simulation of attribute secret keys, $sk_i$ must be in the form,

$$sk_i = g^{\frac{(\hat{r}b + \frac{ab}{l})q_i}{b}}$$

since the simulator $\mathcal{S}$ sets the secret exponent of any attribute $i \notin A_{\mathcal{T}^*}$ as $b/q_i$. Hence, $\mathcal{S}$ sets $sk_i = g^{\hat{r}q_i} \cdot g^{aq_i/l}$. It is evident that this is a valid simulation of secret keys, since, $sk_i = g^{(\hat{r}b + \frac{ab}{l})q_i/b} = g^{\hat{r}q_i} \cdot g^{aq_i/l}$. Then, $\mathcal{S}$ sends the secret keys ($sk_0^i$, $sk_i$) for each attribute $i \notin A_{\mathcal{T}^*}$.

**Challenge phase**: $\mathcal{A}$ sends two plaintexts $M_0, M_1 \in \mathbb{G}_1$ to $\mathcal{S}$. Then $\mathcal{S}$ will encrypt one of $M_0, M_1$ according to $\mathcal{T}^*$ by flipping a fair binary coin $v$. To encrypt $M_v$, the

simulator $\mathcal{S}$ first computes $C_0$ and $\{C_i'\}_{i=1,2,...,l}$ such that,

$$\begin{aligned} C_0 &= M_v \prod_{i=1}^{l} Y_i^c = M_v \cdot e(g,g)^{\sum_{i=1}^{l}(\alpha_i)c} \\ &= M_v \cdot e(g,g)^{\sum_{i=1}^{l}(\frac{ab}{l}+\gamma_i)c} = M_v \cdot R_\delta \cdot e(g,g)^{\sum_{i=1}^{l}(\gamma_i)c} \\ C_i' &= g^{\beta_i c}. \end{aligned}$$

For each attribute in $A_{\mathcal{T}^*}$ except the last, a random exponent $h_i \in \mathbb{Z}_p^*$ is chosen and assigns $g^{h_i}$, while the last element is assigned the value $g^{h_s} = {g^{lc}}/{\prod_{i=1}^{s-1} g^{h_i}}$. Then, $\mathcal{S}$ computes $\{C_i''\}_{i=1,2,...,s}$ such that, $C_i'' = g^{q_i h_i}$. Thus, the ciphertext of $M_v$ is given by,

$$E_v = (\mathcal{T}^*, C_0, \{C_i'\}_{i=1,2,...,l}, \{C_i''\}_{i=1,2,...,s}).$$

Then, $\mathcal{S}$ forwards the resulting ciphertext $E_v$ to $\mathcal{A}$.

**Phase 2**: The simulator $\mathcal{S}$ acts exactly as it did in Phase 1.

**Guess:** The adversary $\mathcal{A}$ submits a guess $v' \in \{0,1\}$. If $v' = v$, the simulator $\mathcal{S}$ will guess that $\delta' = 0$ and outputs a 0 indicating that $R_\delta = e(g,g)^{abc}$. Otherwise, the simulator $\mathcal{S}$ will guess that $\delta' = 1$ and outputs a 1 indicating $R_\delta = e(g,g)^z$.

In the case where $\delta = 0$ ($R_\delta = e(g,g)^{abc}$), $\mathcal{A}$ sees a valid encryption of $M_v$. Therefore, $\mathcal{A}$ has an advantage of $\epsilon$ in winning the game. Hence, according to (5.6),

$$Pr[v' = v | R_\delta = e(g,g)^{abc}] = \frac{1}{2} + \epsilon.$$

Since, the simulator $\mathcal{S}$ guesses that $\delta' = 0$ when $v' = v$ we have,

$$Pr[\delta' = \delta | \delta = 0] = \frac{1}{2} + \epsilon. \tag{5.7}$$

When $\delta = 1$ ($R_\delta = e(g,g)^z$), the adversary $\mathcal{A}$ will not have any advantage in the game, since $\mathcal{A}$ does not gain any information on $M_v$. Therefore,

$$Pr[v' \neq v | R_\delta = e(g,g)^z] = \frac{1}{2}.$$

Given that the simulator $\mathcal{S}$ guesses $\delta' = 1$ when $v' \neq v$ it is evident that,

$$Pr[\delta' = \delta | \delta = 1] = \frac{1}{2}. \tag{5.8}$$

Therefore, according to (5.7) and (5.8), the total advantage of the simulator $\mathcal{S}$ to solve the DBDH problem is given by,

$$\frac{1}{2}Pr[\delta' = \delta|\delta = 0] + \frac{1}{2}Pr[\delta' = \delta|\delta = 1] - \frac{1}{2} = \frac{\epsilon}{2}.$$

This provides evidence that, if there exists an adversary who can win the IND-CPA game with an advantage of $\epsilon$, it is possible to use this adversary to build a simulator that can solve the DBDH problem with an advantage of $\epsilon/2$. Hence, the proposed MA-CP-ABE scheme is IND-CPA secure under the DBDH assumption.

**Resistance against attribute collusion:** We ensure the prevention of collusion attacks via infusing identity related characteristic to each obtained secret key relevant for a given attribute. Suppose, two EHR users $U_1$ and $U_2$ wish to collude secret keys of two attributes $\omega_1, \omega_2$ which are owned by $U_1$ and $U_2$ respectively. Further, assume that $\omega_1$ is managed by $AA_1$ and $\omega_2$ is managed by $AA_2$ while $t_1, t_2$ denote the corresponding attribute secret exponents defined by the respective AA. Then, according to (5.3) - (5.5) the ciphertext $E$ for the plaintext $M$ encoded with the access sub-structure $\mathcal{T}' = \omega_1 \wedge \omega_2$ is given by,

$$E = (\mathcal{T}', C_0, \{C_i'\}_{i=1,2}, \{C_i''\}_{i=1,2})$$

in which $C_0 = M \cdot e(g,g)^{(\alpha_1+\alpha_2)h}$, $C_i' = g^{\beta_i h}$ and $C_i'' = g^{t_i h_i}$.

Also, the secret keys of $U_1$ and $U_2$ corresponding to the attributes $\omega_1$ and $\omega_2$ are given by $(g^{r_1/t_1}, g^{(\alpha_1 - r_1)/\beta_1})$ and $(g^{r_2/t_2}, g^{(\alpha_2 - r_2)/\beta_2})$ respectively. In the attempt to decrypt $E$, $U_1$ and $U_2$ can compute,

$$temp_1 = e(g^{t_1 h_1}, g^{r_1/t_1}) \cdot e(g^{t_2(2h-h_1)}, g^{r_2/t_2}) \tag{5.9}$$

$$temp_2 = e(g^{\frac{\alpha_1 - r_1}{\beta_1}}, g^{\beta_1 h}) \cdot e(g^{\frac{\alpha_2 - r_2}{\beta_2}}, g^{\beta_2 h}). \tag{5.10}$$

From (5.9) and (5.10) users can compute the helper string $\Omega$ such that,

$$\begin{aligned}\Omega &= temp_1 \cdot temp_2 \\ &= e(g,g)^{(\alpha_1+\alpha_2)h} \cdot e(g,g)^{r_1 h_1} \cdot e(g,g)^{r_2(2h-h_1)} \cdot e(g,g)^{-(r_1+r_2)h}.\end{aligned} \tag{5.11}$$

In order to recover $M$ from $C_0$, the computation result in (5.11) must be equiva-

lent to $e(g,g)^{(\alpha_1+\alpha_2)h}$. The aforementioned equivalence will only be possible if the following condition is held.

$$e(g,g)^{r_1 h_1} \cdot e(g,g)^{r_2(2h-h_1)} \cdot e(g,g)^{-(r_1+r_2)h} = 1 \tag{5.12}$$

The relation in (5.12) can only be maintained if and only if $r_1 = r_2$. Hence, it is infeasible to achieve a successful decryption via colluding attribute secret keys of two or more users.

### 5.4.8 Performance Evaluation

In this subsection, we evaluate the performance of the utilized MA-CP-ABE scheme which functions as the underlying AC mechanism in Scheme 6. Furthermore, we also compare Scheme 6 with similar health information sharing schemes which utilize ABE as the AC mechanism.

Computational overhead of the proposed MA-CP-ABE scheme heavily depends upon the overhead associated with encryption and decryption operations given that they require exponentiation and pairing operations in $\mathbb{G}_0$. Thus, we conduct simulations for determining the approximated computational cost (in terms of the computation time) for the above mentioned two processes. The simulations were run on a Core i5, $2.5\,GHz$ PC with $8\,GB$ of RAM. The necessary cyclic groups were generated with the help of the elliptic curve $y^2 = x^3 + x$ over a $512$-bit finite field having a group order of 160 bits, since it can generate keys having the equivalence security of $1024$-bit RSA keys [75].

For the analysis, we simulated a simple multi-authority environment with 5 AAs each managing 10 attributes. We conducted simulations to determine the behavior of the encryption cost and the decryption cost with the number of attributes in a given access sub-structure $\mathcal{T}'$ under the following four cases.

- Case 1: All the attributes in $\mathcal{T}'$ belong to the same AA.
- Case 2: Attributes in $\mathcal{T}'$ belong to 2 AAs.
- Case 3: Attributes in $\mathcal{T}'$ belong to 3 AAs.
- Case 4: Attributes in $\mathcal{T}'$ belong to 4 AAs.

The obtained results are illustrated in Figure 5.2 and Figure 5.3. Figure 5.2 represents the variation of the encryption cost with respect to the four aforementioned cases while Figure 5.3 represents the variation of the decryption cost. According to

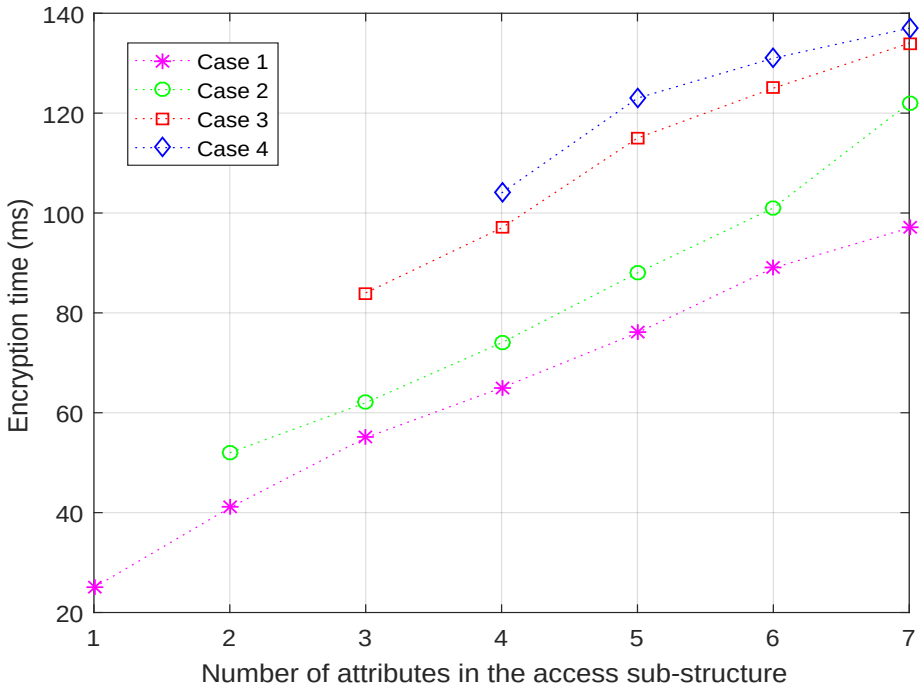



Figure 5.2: Encryption cost with the number of attributes in the access sub-structure

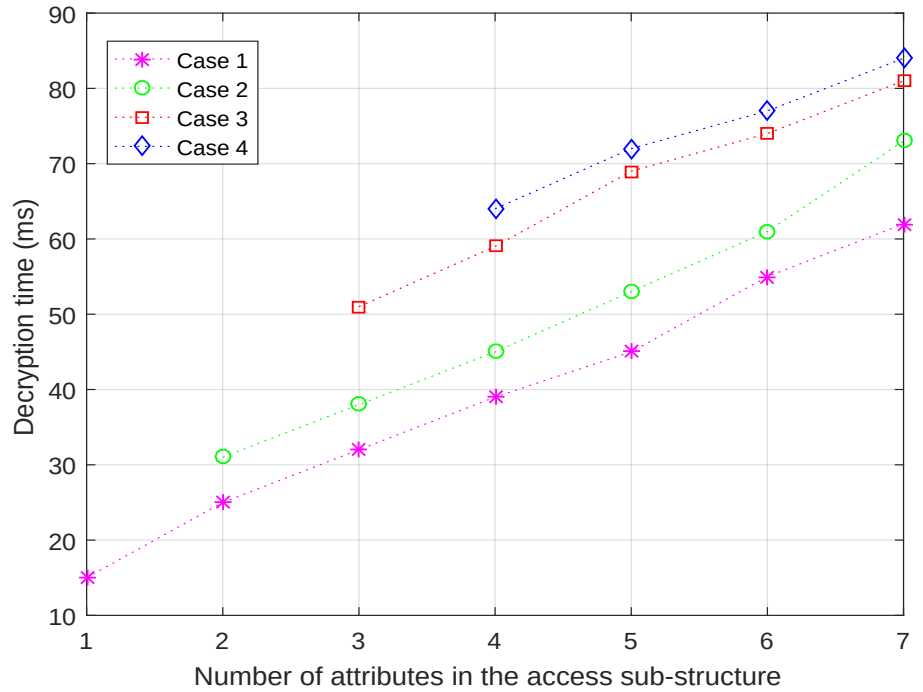



Figure 5.3: Decryption cost with the number of attributes in the access sub-structure

the results, it is obvious that both the encryption and the decryption cost increase with the number of attributes in $\mathcal{T}'$. Furthermore, we can also observe that the decryption cost is slightly lower than the encryption cost since the decryption pro-

cess requires less number of exponentiation operations compared to the process of encryption in the proposed MA-CP-ABE scheme. Note that we considered a maximum of 7 attributes in the access sub-structure $\mathcal{T}'$ since we may not come across sub-structures having more than 7 attributes in practice, especially considering our application of interest. However, given that the variation of computational cost is almost linear, it is fair to conclude that the proposed scheme is scalable and will function effectively under access sub-structures with a larger number of attributes as well.

Along with the computational cost, it is also important to analyze the size of an encrypted message when utilizing the proposed scheme under the considered parameter setting. Suppose, a message $M \in \mathbb{G}_1$ is encrypted with a sub-structure $\mathcal{T}'$ having $s$ attributes managed by $l$ AAs. Given that we use a 512-bit finite field to generate the cyclic groups, each ciphertext component $(C_0, C', C'')$ will be of 1024 bits. Hence, the size of the ciphertext is $1024(s + l + 1)$ bits. It is also important to note that, although we assumed $M \in \mathbb{G}_1$ (1024 bits), message sizes of health information could be much larger in practice, especially considering medical images. Thus, we can use an Advanced Encryption Standard (AES) symmetric key $K$ to encrypt the message $M$, and then encrypt $K$ with the proposed MA-CP-ABE scheme.

A comparison between the proposed Scheme 6 and the identified related works which propose similar health information sharing schemes are tabulated in Table 5.1. The health information sharing scheme in [111] uses the CP-ABE scheme in [105] as the underlying AC mechanism while the schemes proposed in [107, 112] use the CP-ABE scheme in [106] and [101] respectively. All of the above mentioned CP-ABE schemes use a centralized TA to manage and issue attributes to all the users in the system. Such a centralized approach is not suitable for EHR sharing application in consideration with the associated access requirements. For instance, an EHR may need to be shared among a set of users having attributes from more than one organizational entity (i.e. allowing access for any physician from hospital A or hospital B). In such a scenario, it is not realistic to assume that attributes specific for each organizational entity is issued by a centralized TA. In our solution, we adopt a distributed system architecture such that each entity has the capability of operating as an AA.

In contrast to the aforementioned solutions with a centralized TA, the scheme in [23] presents a health information sharing scheme supporting a distributed attribute architecture by utilizing the MA-ABE scheme in [100]. This MA-ABE scheme in [100] requires a user to have at least one attribute from each of the available AAs,

Table 5.1: Comparison of ABE based health information sharing schemes

| Scheme | AC mechanism | Attribute management | Comment |
|---|---|---|---|
| [107] | CP-ABE [106] | Centralized | Central TA to manage all attributes |
| [111] | CP-ABE [105] | Centralized | ” |
| [112] | CP-ABE [101] | Centralized | ” |
| [23] | MA-ABE [100] | Distributed | Not scalable |
| Scheme 6 | Proposed MA-CP-ABE | Distributed | - |

and therefore the health information sharing scheme in [23] will not be scalable. For instance, let us consider the following scenario. Assume that there are 100 AAs in the system and the data owner wants to encrypt an object $O$ with only one attribute which belongs to a specific AA. However, for the proper operation of the utilized MA-ABE scheme in [23], $O$ must be encrypted with at least one attribute from each AA (which can be achieved via dummy attributes). This applies to the decryption as well. Thus, the computation cost increases significantly with the number of AAs in the system, although the number of real attributes used for the encryption is significantly low. In our solution, we overcome this issue, and the data owner only needs the public keys of AAs corresponding to the attributes he uses to encrypt EHR data while decrypting user only needs secret keys corresponding to the attributes used during the encryption process.

## 5.5 Chapter Summary

In this chapter, we have presented Scheme 6 which proposes a secure and scalable ABE based AC mechanism (MA-CP-ABE) which is compatible with the access requirements associated with AS 2. Scheme 6 addresses the challenges brought by provisioning fine-grained access for encrypted EHRs to users while overcoming the practicality and scalability limitations associated with the existing ABE driven health information sharing schemes. Our MA-CP-ABE scheme is more scalable, since it facilitates the data owner to encrypt data with a set of attributes (from one or more AAs) in such a way that a user who possesses secret keys corresponding to the aforementioned attributes can successfully decrypt the data (i.e. the scheme does not require a user to have secret keys from all AAs). We have also shown that the proposed scheme is resistant against chosen plaintext attacks and attacks mounted via attribute collusion. Furthermore, the scheme can handle attribute revocation which helps in preventing illegal access via already revoked attributes.

# Chapter 6

# Delegatable ABAC Scheme for AS 2

*Enforcing controlled access delegatability along with enabling fine-grained access to remotely stored encrypted patient EHRs is a challenging research question. In Chapter 6, we intend to address this problem by proposing a delegatable ABAC scheme which is compatible with the health information sharing scenario specified in AS 2. The proposed scheme (Scheme 7) is an extension of the MA-CP-ABE construction in Scheme 6. We start the chapter by summarizing the influential research work in the direction of provisioning access delegatability over encrypted data. Then, Scheme 7 is presented in detail followed by some concluding remarks which conclude the chapter. The proposed Scheme 7 is based on our work in Paper 8.*

## 6.1 Related Work

As we have presented in Chapter 5, ABE is capable of provisioning secure and fine-grained access to remotely stored encrypted data. However, there have only been few related works which have explored the issue of access delegation on data encrypted with ABE schemes.

In [113, 114], the authors have proposed ABE based mechanisms to delegate access to encrypted data by allowing users to delegate their attributes. The solution in [113] allows a user to delegate a subset of attributes he owns to another user by issuing secret keys. The main advantage of this scheme is that it helps to relieve the computation and management overhead on the central TA, given that it only needs to manage higher level users since other users can obtain relevant keys from the higher level users. However, the main issue is that this scheme is not equipped with any

mechanism to control subsequent re-delegations made by the users. This means that a user $U_2$ who obtained attributes from another user $U_1$ will be able to re-delegate the delegated attributes without the consent of the initial delegator $U_1$. Furthermore, the intermediate delegators will not be able to revoke their descendants and revocation can only be enforced by the central TA. This issue is addressed in [114] with the help of a third-party mediator which facilitates the delegation process. During delegation of attributes, the delegator needs to inform the mediator about the impending delegation including which attributes are delegated. Then, the delegatee can obtain the relevant delegated attribute keys from the delegator and the mediator as partial keys. This induces some control over the process of delegation since the delegator's permission over re-delegation of attributes is enforced through the mediator. However, given the fact that the mediator has a full view of who delegates which attribute to whom, might affect the privacy of users.

Proxy re-encryption (PRE) schemes can also be used for achieving access delegation over encrypted data. In general, PRE allows a third-party entity (proxy) to translate a ciphertext encrypted under a specific entity's public key in such a way that the translated ciphertext can be decrypted by a different entity's private key. During the process of conversion, the proxy will not learn either of the decryption key or the plaintext. Although the traditional PRE schemes have been primarily developed for enforcing one-to-one access delegation [115–120], the integration of PRE with ABE has made it possible to achieve many-to-many access delegation [121–123]. Many-to-many access delegation permits a user who has the capability to decrypt a specific ciphertext to re-delegate the access to a set of users who do not have the capability to decrypt the ciphertext. However, the schemes presented in [121, 122] have no control over the re-delegation of access while the scheme in [123] has control over the re-delegation to a certain extent. This is due to the fact that the data owner who initially encrypts the data can allow or deny the ability of re-encryption of the ciphertext and thereby the further delegations. However, if the data encryptor has allowed the ciphertext to be delegated, further re-delegations by the delegatees will be uncontrolled. In addition, all the above mentioned solutions allow a delegatee to access all the data that can be accessed by the delegator with the delegated attributes. Hence, a delegator will not be able to selectively provide access only to a subset of data (selective delegatability) that can be accessed with the delegated attributes.

The above analysis shows that the existing schemes suffer from the inability of provisioning controlled access delegation over remotely stored encrypted data. As a solution, in this chapter, we propose Scheme 7 in which we extend the MA-

CP-ABE construction in Scheme 6 to facilitate flexible as well as controlled access delegation with respect to AS 2. This scheme is capable of provisioning multi-level, many-to-many controlled access delegation meaning that a user who is eligible to decrypt a specific ciphertext can delegate the access to that ciphertext based on attributes to a set of users. Furthermore, any user who is eligible to access the delegated ciphertext can re-delegate it further, if and only if the preceding delegator has given the permission to do so. Moreover, unlike the existing ABE schemes with delegatability, where a delegatee becomes eligible to access all data items associated with delegated attributes, Scheme 7 enables a user to delegate access to a subset of data that can be accessed with the delegated attributes (selective delegatability of data) which helps to induce more control over the process of access delegation.

## 6.2 Security and Privacy Requirements

Before presenting Scheme 7, we introduce the security and privacy requirements to be satisfied in the proposed delegatable ABAC scheme with respect to AS 2.

- The requirements: *confidentiality of EHR data*, *resistance against attribute collusion*, *attribute revocation* and *user privacy* are as introduced in Section 5.3.

- *Controlled access delegation:* Access delegation on patients' EHR data must be controlled meaning that further delegations by a delegatee are only feasible with the consent of the delegator.

- *Patient privacy:* We require the users who do not possess enough attributes to satisfy the access requirement to be prevented from gaining access to the stored EHR data of patients. The properties: EHR data confidentiality, resistance against attribute collusion as well as controlled access delegation contributes to strengthening the patient privacy.

## 6.3 Proposed Scheme 7

We present this scheme, starting with an overview and it is followed by the eight phases in the proposed scheme: system initialization, attribute key distribution, EHR encryption, EHR decryption without access delegation, EHR access delegation, EHR decryption under access delegation, how the scheme is extended to multi-level access delegation and the revocation of users from attributes. Similar to other

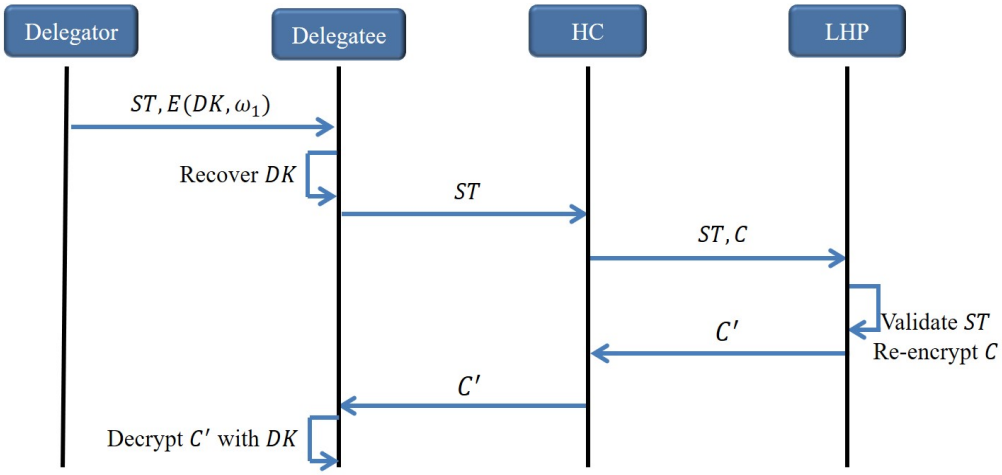



Figure 6.1: Flow diagram representing the overview of EHR access delegation in Scheme 7

schemes, we conclude this section with the security analysis and the performance evaluation of Scheme 7.

### 6.3.1 Overview of Scheme 7

Scheme 7 is based on the MA-CP-ABE construction proposed in Scheme 6. Hence, a user will be able to decrypt a specific encrypted $EHR_{obj}$ stored in the HC with the help of secret attribute keys corresponding to the attributes which satisfy the access sub-structure associated with the $EHR_{obj}$. The following describes how we extend Scheme 6 to provide access delegation in Scheme 7. Suppose, a user $U$ (delegator) wants to delegate the access to $EHR_{obj}$, $O$ ($U$ has necessary attributes $\omega$ to access $O$) to any user who is having the set of attributes $\omega_1$. First, $U$ generates a re-encryption key $RK_{\omega \to \omega_1}$ which is used to translate the ciphertext $C$ (of $O$ stored in HC) into a form that allows it to be decrypted with the attributes $\omega_1$. In addition, $U$ generates a decryption key $DK$, which facilitates the ciphertext to be decrypted after the re-encryption (using $RK_{\omega \to \omega_1}$). $U$ encrypts $RK_{\omega \to \omega_1}$ with the public key of LHP $PK_{LHP}$ (i.e. $E(RK_{\omega \to \omega_1}, PK_{LHP})$) and $DK$ with the public attribute keys corresponding to the attributes in $\omega_1$ (i.e. $E(DK, \omega_1)$). Furthermore, $U$ generates a token including the information about the access delegated $EHR_{obj}$, $E(RK_{\omega \to \omega_1}, PK_{LHP})$, some auxiliary information (details are given in Section 6.3.6) and signs it with the attribute secret keys associated with $\omega$. The resulting signed token ($ST$) is finally sent to the delegatees along with $E(DK, \omega_1)$.

When a delegatee requires accessing the $EHR_{obj} = O$, the delegatee should send the signed token $ST$ to the HC. This allows HC to determine that the requested access is subjected to a delegation; hence HC will forward $ST$ along with the relevant ciphertext $C$ to LHP. Then, the LHP will validate the token signature,

if validated successfully, the re-encryption key $RK_{\omega \to \omega_1}$ is recovered, and $C$ is re-encrypted with $RK_{\omega \to \omega_1}$. Then, the re-encrypted ciphertext $C'$ is forwarded to the delegatee through HC. Finally, the delegatee will be able to decrypt $C'$ with the help of the decryption key $DK$.

### 6.3.2 System Initialization

To initialize the system, first, a set of global public parameters is generated which is shared among all AAs. AAs agree on two multiplicative cyclic groups $\mathbb{G}_0$, $\mathbb{G}_1$ of prime order $p$ with $g_0, g_1$ being generators of $\mathbb{G}_0$ and a bilinear map $e : \mathbb{G}_0 \times \mathbb{G}_0 \to \mathbb{G}_1$. In addition, two secure hash functions $H_1, H_2$ are also agreed upon, where $H_1 : \{0,1\}^* \to \mathbb{Z}_p^*$ and $H_2 : \mathbb{G}_1 \to \mathbb{Z}_p^*$. AAs also agree on a shared secret $\xi \in \mathbb{Z}_p^*$ as was in Scheme 6. Then, AAs publish the set of global public parameters $(\mathbb{G}_0, \mathbb{G}_1, H_1, H_2, e, g_0, g_1, p)$. After the global initialization of AAs, each AA is locally initialized, and the initialization procedure is described below with respect to $AA_k$, assuming that the attribute set administered by $AA_k$ is denoted by $A^k$.

- $AA_k$ chooses two random exponents $\alpha_k, \beta_k \in \mathbb{Z}_p^*$ and computes $X_k = {g_0}^{\beta_k}$, $Y_k = {g_1}^{\beta_k}$, $Z_k = e(g_0, g_0)^{\alpha_k}$. Then, a unique random identifier $t_{k,i} \in \mathbb{Z}_p^*$ for each attribute $i$ in $A^k$ is selected. Furthermore, each attribute managed by $AA_k$ is also bound to public attribute keys $T_{k,i}, D_{k,i}$, where $T_{k,i} = {g_0}^{t_{k,i}}$, $D_{k,i} = {g_1}^{t_{k,i}}$.

- $AA_k$ will keep $\{\xi, \alpha_k, \beta_k, t_{k,i}\}_{i=1,2,\ldots,|A^k|}$ as the master secret set denoted by $MK_k$ and publish $\{X_k, Y_k, Z_k, T_{k,i}, D_{k,i}\}_{i=1,2,\ldots,|A^k|}$ as the authority's public tuple denoted by $PK_k$.

### 6.3.3 Attribute Key Distribution

Let us assume that, the user $U_m$ wants to acquire attribute keys for the set of attributes $A_m$ in which $A_m^k \subseteq A_m$ denotes the subset of attributes which should be acquired from $AA_k$. The construction of attribute keys for this scheme is identical to the construction of attribute keys for Scheme 6. Therefore, according to Section 5.4.3, if the attribute key set issued to $U_m$ by $AA_k$ is given by $SK_m^k$ then,

$$SK_m^k = \{sk_0^k, sk_i^k\}_{i=1,2,\ldots,|A_m^k|} \text{ in which,}$$

$$sk_0^k = {g_0}^{\frac{\alpha_k - r_m}{\beta_k}} \tag{6.1}$$

$$sk_i^k = g_0^{\frac{r_m}{t_{k,i}}}. \tag{6.2}$$

Note that $r_m = H_1(ID_m) + \xi$, where $ID_m$ denotes the identity of $U_m$. We refer the readers to Section 5.4.3 for more details on attribute key distribution.

### 6.3.4 EHR Encryption

Suppose, LHP wants to encrypt EHR data $M \in \mathbb{G}_1$, which corresponds to the object $O$ of a given patient's EHR $I$. LHP starts the process by generating the access structure $\mathcal{T}$ and deducing a set of access sub-structures $\{\mathcal{T}_k\}_{k=1,2,...,q}$ as explained in Section 2.1.3. Thus, the ciphertext of $M$ encrypted with $\mathcal{T}$ is given by,

$$E(M) = (\mathcal{T}, \{E_k\}_{k=1,2...,q})$$

where $E_k$ denotes the ciphertext of $M$ encrypted with the access sub-structure $\mathcal{T}_k$. The process of computing $E_k$ is as follows.

Let us assume that the sub-structure $\mathcal{T}_k$ contains $s$ attributes and they are managed by $l$ AAs such that, $l \leq N$, where $N$ is the total number of AAs in the system. Then, the ciphertext $E_k$ can be represented using the ciphertext components $C_0, \{C_i', AD_i'\}_{i=1,2,...,l}$ and $\{C_i'', AD_i''\}_{i=1,2,...,s}$, such that,

$$E_k = (\mathcal{T}_k, C_0, \{C_i', AD_i'\}_{i=1,2,...,l}, \{C_i'', AD_i''\}_{i=1,2,...,s}).$$

To compute the ciphertext components in $E_k$, LHP first generates a random exponent $h \in \mathbb{Z}_p^*$ and using the public keys of $l$ AAs, it computes the ciphertext components $C_0$ and $\{C_i', AD_i'\}_{i=1,2,...,l}$ such that,

$$C_0 = M \prod_{i=1}^{l} Z_i^h = M \cdot e(g_0, g_0)^{h \sum_{i=1}^{l} \alpha_i} \tag{6.3}$$

$$C_i' = X_i^h = g_0^{\beta_i h} \tag{6.4}$$

$$AD_i' = Y_i^h = g_1^{\beta_i h}. \tag{6.5}$$

To compute $\{C_i'', AD_i''\}_{i=1,2,...,s}$, a secret share of $h$ is assigned for each attribute in $\mathcal{T}_k$ as follows.

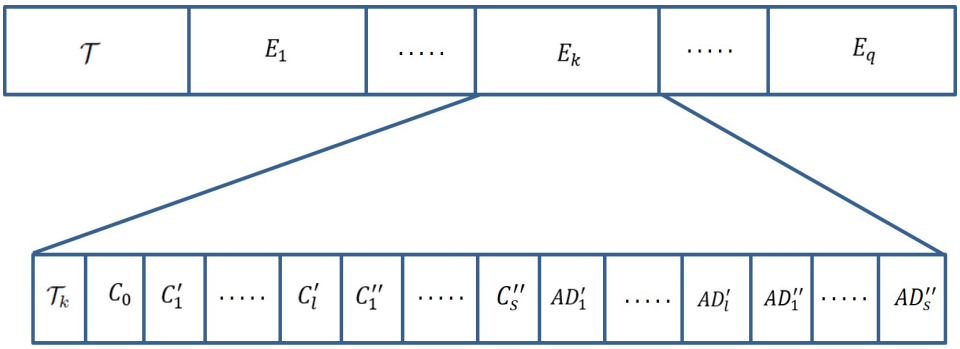



Figure 6.2: Structure of the ciphertext $E(M)$ in Scheme 7

- For each attribute in $\mathcal{T}_k$ except the last, a random exponent $h_i \in \mathbb{Z}_p^*$ is assigned while the last element is assigned the value equals to $l \cdot h - \sum_{i=1}^{s-1} h_i$.
- Then, the LHP computes $\{C''_i, AD''_i\}_{i=1,2,...,s}$ such that,

$$C''_i = T_i^{h_i} = {g_0}^{t_i h_i} \tag{6.6}$$

$$AD''_i = D_i^{h_i} = {g_1}^{t_i h_i} \tag{6.7}$$

where $T_i, D_i$ correspond to the public attribute keys of the $i^{th}$ attribute in $\mathcal{T}_k$.

Similarly, LHP generates the ciphertexts relevant for all the sub-structures of $\mathcal{T}$ to complete the encryption of the object $O$. We have illustrated the structure of the ciphertext $E(M)$ in Figure 6.2.

### 6.3.5 EHR Decryption without Access Delegation

Before we present the details on how the proposed Scheme 7 facilitates access delegation, it is essential to briefly present the EHR decryption process, without delegation.

Suppose, $U_m$ wants to access the $EHR_{obj}$, $O$ associated with the EHR, $I$ stored in the HC. $U_m$ initiates the process by sending an access request indicating the $EHR_{id}$ and $EHR_{obj}$ information corresponding to the access required EHR to HC. Then, HC fetches the corresponding $\mathcal{T}$ associated with $O$ and sends it back to $U_m$. This allows $U_m$ to determine the smallest subset of attributes $A'_m \subseteq A_m$ that satisfy the received $\mathcal{T}$ and generate the sub-structure $\mathcal{T}'$ based on the attributes in $A'_m$. Upon receiving $\mathcal{T}'$, HC fetches the corresponding EHR ciphertext $E'$ and sends them back to $U_m$ which enables him to decrypt the encrypted data using the relevant attribute secret keys.

To describe the decryption process, let us assume that the received ciphertext $E'$ is encrypted with $s$ attributes which are managed by $l$ AAs. Then,

$$E' = (\mathcal{T}', C_0, \{C_i'\}_{i=1,2,...,l}, \{C_i''\}_{i=1,2,...,s})$$

in which $C_0$, $C_i'$ and $C_i''$ are given in (6.3),(6.4) and (6.6) respectively. HC will not send $\{AD_i'\}_{i=1,2,...,l}, \{AD_i''\}_{i=1,2,...,s}$ ciphertext components to $U_m$, given that these components are only required during decryption subjected to access delegation. Hence, the decryption without access delegation in Scheme 7 resembles EHR decryption in Scheme 6. Therefore, $U_m$ can recover $M$ by computing,

$$\frac{C_0}{\prod_{i=1}^{s} e(C_i'', sk_i) \cdot \prod_{i=1}^{l} e(C_i', sk_0^i)} = \frac{M \cdot e(g_0, g_0)^{h \sum_{i=1}^{l} \alpha_i}}{e(g_0, g_0)^{h \sum_{i=1}^{l} \alpha_i}} = M.$$

Note that,

$$sk_i = {g_0}^{r_m/t_i} \text{ and } sk_0^i = {g_0}^{(\alpha_i - r_m)/\beta_i} \tag{6.8}$$

where $sk_i$ is the secret attribute key component that relates the identity of $U_m$ to the $i^{th}$ attribute in $A_m'$ whereas $sk_0^i$ is the secret key component that relates the identity of $U_m$ to the identity of the AA which issued the $i^{th}$ attribute in $A_m'$.

### 6.3.6 EHR Access Delegation

Suppose, now $U_m$ wants to delegate the access to the $EHR_{obj} = O$ which is associated with an access structure $\mathcal{T}$ to any user who is having attributes that satisfy an access sub-structure $\mathcal{T}''$ ($\mathcal{T}''$ is not a sub-structure of $\mathcal{T}$). The delegation procedure is divided into three parts as re-encryption key generation, decryption key generation and signed delegation token generation.

**Re-encryption key generation:** As we have shown in Section 6.3.5, it is possible for $U_m$ to decrypt the object $O$ using his secret keys associated with the sub-structure $\mathcal{T}'$. Hence, $U_m$ uses the attribute keys associated with the attributes in $\mathcal{T}'$ to generate re-encryption keys. The process of generating re-encryption keys is described below. Note that $A_m'$ denotes the attribute subset of $U_m$ associated with the sub-structure $\mathcal{T}'$ and $|A_m'| = s$. Furthermore, it is assumed that the $s$ attributes have come from $l$ AAs.

- $U_m$ selects a random exponent $d \in \mathbb{Z}_p^*$.

- If the set of re-encryption keys is denoted by $RK$ then,

$$RK = \{\{RK_i\}_{i=1,2...,s}, \{RK_0^i\}_{i=1,2,...,l}\} \tag{6.9}$$

$$RK_i = sk_i \cdot {g_1}^d = {g_0}^{\frac{r_m}{t_i}} \cdot {g_1}^d \tag{6.10}$$

$$RK_0^i = sk_0^i \cdot {g_1}^d = {g_0}^{\frac{\alpha_i - r_m}{\beta_i}} \cdot {g_1}^d \tag{6.11}$$

in which $sk_i$ and $sk_0^i$ represent the secret keys of $U_m$ associated with the $i^{th}$ attribute in $A'_m$. In order to allow the re-encryption key $RK$ to be only available to the LHP, elements in $RK$ are encrypted with the public key component ${g_0}^{\beta_1}$ of LHP ($AA_1$). To achieve that, $U_m$ selects a random exponent $a \in \mathbb{Z}_p^*$ and computes the encrypted re-encryption key,

$$ERK = \{\{RK_i \cdot {g_0}^{a\beta_1}\}_{i=1,2...,s}, \{RK_0^i \cdot {g_0}^{a\beta_1}\}_{i=1,2...,l}, {g_0}^a\}. \tag{6.12}$$

**Decryption key generation:** The decryption key allows a user with a set of attributes that satisfy the sub-structure $\mathcal{T}''$ to access the delegated $EHR_{obj}$ after it is re-encrypted with $RK$. If the decryption key is given by $DK$, then $DK = {g_0}^d$. To ensure $DK$ can only be accessed by a user having attributes that satisfy $\mathcal{T}''$, $DK$ is encrypted with the attributes in $\mathcal{T}''$ with the help of the encryption mechanism associated with our MA-CP-ABE scheme given in Section 5.4.4. We denote the encrypted decryption key as $EDK$.

**Signed delegation token generation:** The delegation token ($DT$) allows a delegatee to provide evidence for the LHP that he has the right to access the delegated $EHR_{obj}$. The token includes information on the access delegated $EHR_{obj}$, sub-structure associated with the delegator ($\mathcal{T}'$), sub-structure associated with the delegatees ($\mathcal{T}''$), encrypted re-encryption key ($ERK$), delegation permission (DP) index, and the token expiration information (EXP). The DP index (either $0$ or $1$) defines the permission for further delegations by the delegatees. DP index of $0$ meaning that the delegatees do not have the right to delegate further while an index of $1$ allows the delegatees to further delegate the access to the considered $EHR_{obj}$. The token expiration information determines the validity period of an issued token. Figure 6.3 illustrates the structure of the delegation token issued by $U_m$ to delegate the access to the $EHR_{obj} = O$ to delegatees having attributes that satisfy $\mathcal{T}''$.

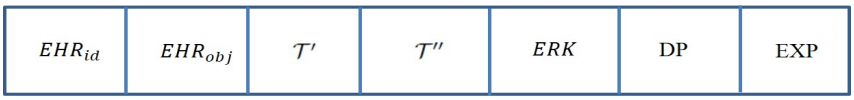



Figure 6.3: Structure of the delegation token issued by $U_m$

Before issuing the token to delegatees, $U_m$ must sign the token using the attribute keys associated with the attributes in $\mathcal{T}'$ which enables LHP (when the delegatee is claiming the access) to determine that the token is generated by an entity which has the right to decrypt the associated $EHR_{obj}$. The token signature generation is as follows. We considered earlier that $\mathcal{T}'$ contains $s$ attributes issued by $l$ AAs. The attribute secret keys corresponding to these attributes ($\{sk_i\}_{i=1,2,...,s}$, $\{sk_0^i\}_{i=1,2,...,l}$) are given in (6.8).

- $U_m$ chooses a random exponent $b \in \mathbb{Z}_p^*$ and a set of exponents $\{u_i\}_{i=1,2,...,l}$ where $u_i$ is the number of attributes in $\mathcal{T}'$ which belongs to $AA_i$.
- Then, $U_m$ generates $\delta_0 = \prod_{i=1}^{l} e(g_0, g_0)^{\alpha_i b u_i}$ and $(\sigma_1, \sigma_2)$ such that,

$$\sigma_1 = \{sk_i{}^b\}_{i=1,2,...,s} = \{g_0{}^{\frac{r_m b}{t_i}}\}_{i=1,2,...,s} \tag{6.13}$$

$$\sigma_2 = \{(sk_0^i)^{u_i b}\}_{i=1,2,...,l} = \{g_0{}^{\frac{(\alpha_i - r_m)u_i b}{\beta_i}}\}_{i=1,2,...,l}. \tag{6.14}$$

- Thereafter, $U_m$ generates $Q, R \in \mathbb{Z}_p^*$ such that $Q = H_1(DT)$ and $R = H_2(\delta_0)$. Using $Q, R$, $U_m$ computes,

$$\sigma_3 = g_0{}^{\frac{\beta_1}{Q+R}} \tag{6.15}$$

  where $g_0{}^{\beta_1}$ is a public key component of LHP ($AA_1$). Then, the signed token is given by $ST = DT||(\sigma_1, \sigma_2, \sigma_3)$.

To complete the delegation of access, finally, $U_m$ sends the signed token $ST$ along with the encrypted decryption key $EDK$ to delegatees.

### 6.3.7 EHR Decryption under Access Delegation

Now, we extend the considered scenario from the previous subsection to describe how a user who has the right to access a specific $EHR_{obj}$ through delegation can successfully access and decrypt the intended $EHR_{obj}$. Consider a user $U_n$, who is having attributes that satisfy the sub-structure $\mathcal{T}''$ and has received the signed delegation token $ST = DT||(\sigma_1, \sigma_2, \sigma_3)$ from $U_m$ to access the object $O$. To gain access, $U_n$, first of all, sends an access request to HC with the received $ST$. With

the reception of $ST$, HC determines that the sender is a delegatee and therefore forwards the signed token $ST$ to LHP along with the EHR ciphertext components associated with the object $O$. The ciphertext components sent by HC to LHP are given by $E' = (\mathcal{T}', \{C'_i\}_{i=1,2,...,l}, \{C''_i\}_{i=1,2,...,s})$, where $C'_i$ and $C''_i$ are as mentioned in (6.4) and (6.6). Upon receiving $ST$ and $E'$, LHP primarily has two tasks: verification of the signed token and the re-encryption of the ciphertext as described below.

**Verification of the signed token:** LHP first checks the expiration information of the token, and if it is not expired, the signature verification is carried out as follows. Note that the signature components $(\sigma_1, \sigma_2, \sigma_3)$ are as given in (6.13) - (6.15). Using $\sigma_1, \sigma_2$ along with the public attribute keys $\{T_i\}_{i=1,2,...,s}$ and the authority related public keys $\{X_i\}_{i=1,2,...,l}$ associated with the attributes in $\mathcal{T}'$ (delegator's substructure) LHP computes the helper string $\delta_1$ such that,

$$\begin{aligned}\delta_1 &= \prod_{i=1}^{s} e({sk_i}^b, T_i) \cdot \prod_{i=1}^{l} e(({sk_0^i})^{u_i b}, X_i) \\ &= \prod_{i=1}^{s} e({g_0}^{\frac{r_m b}{t_i}}, {g_0}^{t_i}) \cdot \prod_{i=1}^{l} e({g_0}^{\frac{(\alpha_i - r_m)u_i b}{\beta_i}}, {g_0}^{\beta_i}) = \prod_{i=1}^{l} e(g_0, g_0)^{\alpha_i b u_i}.\end{aligned}$$

Thereafter, LHP generates $Q', R' \in \mathbb{Z}_p^*$ such that $Q' = H_1(DT)$ and $R' = H_2(\delta_1)$. LHP determines the token signature is valid, given that the condition,

$$e(\sigma_3, {g_0}^{Q'+R'}) \stackrel{?}{=} e({g_0}^{\beta_1}, g_0) \quad \text{is held.}$$

**Re-encryption of the ciphertext:** After validating the token signature, LHP extracts the encrypted re-encryption key $ERK$ (given in (6.12)) and recovers the re-encryption key $RK$ with the help of $ERK$ and the master secret exponent of LHP $\beta_1$. Note that the expression for $RK$ and its elements $\{RK_i\}_{i=1,2,...,s}$ and $\{RK_0^i\}_{i=1,2,...,l}$ are given in (6.9) - (6.11). With the help of $RK$ and the ciphertext components $\{C'_i\}_{i=1,2,...,l}, \{C''_i\}_{i=1,2,...,s}$, LHP computes the re-encrypted cipher $RC$,

$$\begin{aligned}RC &= \prod_{i=1}^{s} e(RK_i, C''_i) \cdot \prod_{i=1}^{l} e(RK_0^i, C'_i) \\ &= \prod_{i=1}^{s} e({g_0}^{\frac{r_m}{t_i}} \cdot {g_1}^d, {g_0}^{t_i h_i}) \cdot \prod_{i=1}^{l} e({g_0}^{\frac{\alpha_i - r_m}{\beta_i}} \cdot {g_1}^d, {g_0}^{h\beta_i})\end{aligned}$$

$$= \prod_{i=1}^{l} e(g_0, g_0)^{\alpha_i h} \cdot e(g_0, g_1)^{d\beta_i h} \cdot \prod_{i=1}^{s} e(g_0, g_1)^{dt_i h_i}.$$

Finally, LHP will send $RC$ to HC to complete the re-encryption process. Then, HC will forward the re-encrypted ciphertext $RC$ along with the ciphertext components $C_0, \{AD_i'\}_{i=1,2,...,l}, \{AD_i''\}_{i=1,2,...,s}$ to $U_n$. With the reception of the aforementioned ciphertext components from HC, $U_n$ can carry out the decryption as follows.

- First, $U_n$ recovers $DK = {g_0}^d$ from the encrypted decryption key $EDK$ using the attribute secret keys associated with the attributes in $\mathcal{T}''$. This decryption is achieved by following the mechanism given in Section 6.3.5.

- Then, $U_n$ will be able to obtain $M$ with the help of $DK, C_0, \{AD_i'\}_{i=1,2,...,l}$, $\{AD_i''\}_{i=1,2,...,s}$ as follows.

$$M' = \frac{C_0 \cdot \prod_{i=1}^{s} e(DK, {AD_i}'') \cdot \prod_{i=1}^{l} e(DK, {AD_i}')}{RC}$$
$$= \frac{C_0 \cdot \prod_{i=1}^{s} e({g_0}^d, {g_1}^{t_i h_i}) \cdot \prod_{i=1}^{l} e({g_0}^d, {g_1}^{\beta_i h})}{\prod_{i=1}^{l} e(g_0, g_0)^{\alpha_i h} \cdot e(g_0, g_1)^{d\beta_i h} \cdot \prod_{i=1}^{s} e(g_0, g_1)^{dt_i h_i}} = M.$$

### 6.3.8 Extending to Multi-level Access Delegation

In Section 6.3.6, we presented how the proposed scheme enables first level access delegation (i.e. $U_m$ who is eligible to access the object $O$ delegates the access to any user having attributes that satisfy the sub-structure $\mathcal{T}''$). Now, let us see, how we can enable multi-level delegation. Suppose, $U_n$ (who received access from the first level delegation) wants to delegate further to any user who is having attributes that satisfy the access sub-structure $\mathcal{T}'''$. $U_n$ proceeds as follows.

- First, $U_n$ encrypts the decryption key $DK = {g_0}^d$ (received from $U_m$) using attributes in $\mathcal{T}'''$ and generates the re-encrypted decryption key $EDK_1$.

- Then, $U_n$ generates a new delegation token $DT_1$ including the delegator's sub-structure ($\mathcal{T}''$), delegatee's sub-structure ($\mathcal{T}'''$), delegation permission (DP) index and the token expiration information (EXP). Note that this new token does not require to have the information on the delegated $EHR_{obj}$ and the encrypted re-encryption key due to the fact that this token is used together with the delegation token associated with first level delegation as an aggregation.

- Suppose, the signed token received by $U_n$ from the first level delegation is given by $ST = \{DT, [DT]_{\mathcal{T}'}\}$. The notation $[DT]_{\mathcal{T}'}$ denotes the signature of

$DT$ made using the attributes in the sub-structure $\mathcal{T}'$. Then, $U_n$ generates the aggregated token ($AGT$) using the received $ST$ and the new delegation token $DT_1$ such that,

$$AGT = \{DT, [DT]_{\mathcal{T}'}\}||\{DT_1, [DT||DT_1]_{\mathcal{T}''}\}.$$

- Finally, $U_n$ forwards $AGT$ along with $EDK_1$ to the delegatees.

Suppose, now a user $U_r$ (who has attributes that satisfy $\mathcal{T}'''$) wants to access the object $O$ with the aggregated token $AGT$. Similar to the first level delegation scenario, $U_r$ should forward $AGT$ to $LHP$ through $HC$. The procedure utilized for re-encryption (at LHP) and decryption at the user's end is as same as in the first level delegation scenario. However, the only difference is the protocol that LHP adopts to validate the aggregated token. For an aggregated token (with two delegation tokens) to be valid, the following conditions must be maintained.

- The first token ($DT$) must not be expired, must have a $DP$ index of 1 and the associated signature ($[DT]_{\mathcal{T}'}$) should be verified.
- The delegatee's sub-structure in the first token must be the same as the delegator's sub-structure in the next token ($DT_1$).
- $DT_1$ should not be expired and its signature $\{DT_1, [DT||DT_1]_{\mathcal{T}''}\}$ must be valid.

Similarly, by aggregating the tokens appropriately as mentioned above, it is possible to achieve higher-order delegations (third level delegation and higher) effectively.

### 6.3.9 Attribute Revocation Mechanism

In this scheme, revocation of attributes from the users must be handled on both attribute level and token level (to revoke delegatees). Attribute level revocation ensures that a user will not be able to use the secret keys related to the revoked attribute in any further transactions. In the proposed scheme, the attribute level revocation is handled by the AA which is responsible for the attribute to be revoked and the procedure is the same as the one used in Scheme 6. Hence, we will not repeat the procedure in this subsection. Scheme 7 adopts two mechanisms to revoke delegatees via revoking their associated delegation tokens. Each token includes its validity period and when issuing a delegation token, issuing entity can choose the validity period appropriately based on the delegatee. This provides revocation

through token expiration. Suppose, a delegator wants to revoke a specific token before it expires. Then, the delegator can enforce it by generating a unique token identifier using $H_1$ and pass it to the LHP. LHP will add the token identifier to its blacklist, which prevents delegatees from using the associated token.

### 6.3.10 Security Analysis

In this subsection, our intention is to show that Scheme 7 exhibits IND-CPA security while being able to resist the attacks mounted via attribute collusion.

**Resistance against chosen plaintext attacks:** Scheme 7 is an extension of the MA-CP-ABE scheme proposed in Scheme 6 to facilitate access delegation. In Scheme 7, data and the decryption key (which facilitates the delegatees to decrypt the re-encrypted data) are encrypted using the data encryption mechanism of Scheme 6. We have shown that the MA-CP-ABE construction in Scheme 6 is IND-CPA secure under the DBDH assumption in Section 5.4.7. Therefore, we can show that Scheme 7 also exhibits IND-CPA security under the DBDH assumption in the same manner.

**Resistance against attribute collusion:** We prevent the possibility of attribute collusion via infusing the user identity to each of the attribute related secret key. This ensures that the secret keys issued to two users concerning the same attribute will be different from each other. Hence, when a message $M$ is blinded with the factor of $e(g_0, g_0)^{h\sum_{i=1}^{l}\alpha_i}$ during encryption, any decrypting user will not be able to successfully reconstruct the blinding factor $e(g_0, g_0)^{h\sum_{i=1}^{l}\alpha_i}$ using the colluded attributes to recover the message $M$. This is evident from the analysis given in Section 5.4.7 for Scheme 6, and it is identical to Scheme 7 under no access delegations.

During an access delegation, the delegator generates the decryption key $DK$ and encrypts it with the attributes in the delegatee's sub-structure. Therefore, to access the delegated resource, the delegatee should first recover $DK$ using the secret keys associated with the attributes in the delegatee's sub-structure. Given that the blinding mechanism used to encrypt $DK$ is similar to the encryption of message $M$ (discussed above), the collusion of attributes by delegatees will not allow a successful recovery of $DK$. Thereby, the possibility of collusion attacks by delegatees is also prevented.

Table 6.1: Comparison of end-user computational complexity of Scheme 7 with existing ABE based delegatable AC schemes

| Scheme | Decryption | | Access delegation | | Decryption under delegation | | Selective delegatability |
|---|---|---|---|---|---|---|---|
| | $N_{exp}$ | $N_e$ | $N_{exp}$ | $N_e$ | $N_{exp}$ | $N_e$ | |
| Scheme 7 | 0 | $\|A_1\|+1$ | $\|A_1\|+\|A_2\|+5$ | 0 | 0 | $\|A_1\|+\|A_2\|+2$ | Yes |
| [123] | 0 | $2\|A_A\|+1$ | $\|A_A\|+5$ | 0 | 1 | $2\|A_A\|+2$ | No |
| [114] | 0 | $2\|A_1\|+1$ | $\|A_2\|+1$ | 0 | 0 | $2\|A_2\|+1$ | No |
| [122] | 0 | $\|A_1\|+1$ | $2\|A_1\|+\|A_2\|+4$ | 0 | 1 | $\|A_2\|+4$ | No |

### 6.3.11 Performance Evaluation

In this section, we provide evidence for the performance of Scheme 7 in terms of the end-user computational complexity as well as comparing its delegatability characteristics with the existing solutions. We have evaluated the end-user computational complexity based on the exponentiation and pairing count for the three processes: decryption without delegation, access delegation and decryption under delegation at the user's end.

In Table 6.1, end-user computational complexity and the delegation capability of the proposed scheme are compared with the most relevant schemes in literature. The notations $|A_1|$, $|A_2|$ and $|A_A|$, denote the number of attributes in the delegator's sub-structure, delegatee's sub-structure and the number of attributes managed by the centralized AA respectively. Note that, all existing solutions have a centralized AA, whereas our scheme supports multiple AAs; hence to make the comparison feasible, we set the number of AAs to 1 in the analysis. We have used $N_{exp}$ and $N_e$ to denote the number of exponentiations and the number of pairing operations respectively.

In [123], it is necessary for a user to have secret keys for all attributes in $A_A$ (negative secret keys for attributes not belonging to the user). Hence, when $|A_A| >> |A_1|$, the scheme in [123] does not scale and will not function efficiently. It is also observable that [114] has a slightly lower end-user computational complexity compared to our scheme, but it uses a third-party mediator to assist the delegation which affects the privacy of users as explained in Section 6.1. Furthermore, our scheme capable of providing better control over delegation while facilitating delegation of access to a subset of data that can be accessed with the delegated attributes (selective delegatability), a characteristic which is not available in other schemes.

## 6.4 Chapter Summary

In this chapter, we extended the MA-CP-ABE scheme proposed in Scheme 6, to facilitate flexible, controlled access delegation on data outsourced to a cloud platform as specified in AS 2. The proposed Scheme 7 has two novel properties. The scheme

is capable of provisioning multi-level access delegation in which subsequent delegation of access by a delegatee is only allowed if the corresponding delegator has given the permission to do so. Furthermore, a user is allowed to selectively delegate data, meaning that delegatees will not be able to access other resources which are associated with the same access sub-structure as the resource for which the access is granted. We have also provided evidence that scheme 7 is IND-CPA secure and exhibits resistance against attacks mounted via attribute collusion as well as providing superior delegating capabilities compared to similar existing solutions.

# Chapter 7

# Conclusions and Future Work

*This chapter is organized into two sections. In the first section, we summarize the dissertation work while pointing out the contributions we made through the proposed ABAC schemes. The second section describes a few potential future research directions in conjunction with our work presented in this dissertation.*

## 7.1 Conclusions

This dissertation addresses several key challenges associated with constructing secure as well as privacy preserving, collaborative e-health environments which are capable of enabling flexible and timely sharing of EHRs of patients among the intended recipients. We considered two application scenarios which we denoted as AS 1 and AS 2. In AS 1, we assumed that the LHP locally stores the EHRs of patients whereas, in AS 2, it was assumed that the LHP stores the patient EHRs in a third-party cloud platform. In relation to the scenarios mentioned above, our objective was to construct secure, privacy preserving and efficient EHR sharing schemes which can provide access anonymization and access delegatability in a controlled manner using ABAC as the underlying AC mechanism. To achieve this goal, in this dissertation, we have proposed seven different ABAC constructions presented in Chapter 3 to Chapter 6.

In Chapter 3, we proposed two anonymous, ABAC schemes: Scheme 1 and Scheme 2 which are compatible with the access requirements associated with AS 1. Scheme 1 allows a user to access an EHR of a patient by constructing a zero-knowledge proof using the secret attribute keys which provide evidence for the LHP that the user owns a valid set of attributes that satisfy the associated access policy.

This scheme also allows the users to be unlinkable over multiple sessions in addition to being able to access the resources anonymously. Furthermore, the Scheme 1 allows a user to selectively disclose the attributes which contribute to user privacy as well as preventing the attacks mounted via attribute collusion and attribute forgery which contributes to strengthening the patient privacy. As we have described in Section 3.6, one of the negatives associated with Scheme 1 is that it induces higher key management overhead to the end-user. This is due to the fact that, a user will have to manage $(d + 1)$ secret keys when obtaining $d$ attributes from a given AA. The proposed Scheme 2 primarily addresses the aforementioned issue. In Scheme 2, we proposed a multi-show unlinkable attribute credential scheme and with the help of this construction an ABAC scheme was proposed which is conducive for the scenario specified in AS 1. In the credential scheme, when a user obtains $d$ attributes from an AA, the resulting credential issued by the AA embeds all the requested attributes in a single credential. Therefore, the user will have to manage only a single secret key; hence the user will have a substantially lower end-user key management overhead in comparison to Scheme 1. We have also shown that the proposed credential scheme outperforms the existing multi-show unlinkable credential schemes in terms of the associated end-user computational complexity providing evidence for its effectiveness. Similar to the Scheme 1, Scheme 2 also exhibits the property of selective disclosure of attributes and resistance against attribute forgery. However, as we have pointed out in Section 3.6, there is a possibility of colluding attributes in credentials owned by different users which makes the collusion of attributes a possibility. This is something that we plan to address in our future work.

The ability of a user to delegate access to another user in a controlled manner significantly improves the access flexibility in a health information sharing environment. To achieve the necessary control over access delegation, we require the three characteristics: a delegator should be able to delegate a subset of the attributes that he owns, the delegator should be able to control further re-delegations by the delegatees and control over the length of a particular chain of delegations to be maintained as stipulated in Section 4.1. From the related work given in Section 4.2, it is evident that controlled access delegation is an area that has not received enough attention, let alone achieving access delegatability in e-health systems. As a solution, we have proposed three ABAC constructions (Scheme 3, Scheme 4 and Scheme 5) with controlled access delegatability in Chapter 4 which are compatible with the health information sharing scenario specified in AS 1. In Scheme 3, we use an attribute based signature scheme to sign attribute tokens which provide the information about the ownership of assigned attributes and delegated attributes. Hence, users will be

able to commit one or more signed tokens to the LHP to provide evidence over the ownership of a set of attributes to gain access to stored EHRs. As shown in Section 4.4.7, Scheme 3 is resistant against attribute forgery, attribute collusion as well as replay attacks. However, Scheme 3 does not provide access anonymity given that each token embeds the PKI certificates of the issuer and the receiver to define the issuer and the receiver of the token uniquely. Furthermore, this scheme induces higher computational overhead when revoking a signed token from a user before its expiry, since this process requires issuing of new tokens to all the users who own the revoking attribute except the user to be revoked. We have addressed the above stated drawback in Scheme 4. In Scheme 4, we use a set of public blockchains to record attribute assignments, delegations as well as revocations. Hence, a revocation can simply be enforced via including a revocation transaction block in the blockchain pointing to the attribute assignment or the delegation which needs to be revoked. Furthermore, in Scheme 4, users are denoted with pseudo-identities, and therefore the scheme provides pseudo-anonymity guarantees to users in comparison to Scheme 3 which does not provide any level of access anonymity. Moreover, the Scheme 4 is also resistant against attribute forgery and attribute collusion given that a user will only be able to provide evidence over the ownership of attributes associated with his pseudo-identity. Our motivation for proposing Scheme 5 is to provide better anonymity guarantees to users in comparison to Scheme 3 and Scheme 4. To achieve this, we extended the anonymous credential scheme proposed in Scheme 2 to facilitate controlled access delegation, and we have shown in Section 4.8.9 that the proposed delegatable credential scheme has superior performance when disclosing delegated credentials in comparison to the only existing delegatable attribute based credential scheme proposed in [84].

Chapters 5 and Chapter 6 were dedicated to addressing the challenges associated with access anonymization and enforcing controlled access delegation when the EHRs of patients are remotely stored in an HC as specified in AS 2. In Chapter 5, we proposed the Scheme 6, in which we have utilized our IND-CPA secure, collusion-resistant CP-ABE construction denoted as the MA-CP-ABE scheme to realize a fine-grained health information sharing scheme conducive for a collaborative e-health environment. Given that this scheme uses an ABE scheme to encrypt data, users will be able to anonymously decrypt the data using the appropriate attribute secret keys upon downloading them from the HC. In comparison to the existing ABE based health information sharing schemes, Scheme 6 is more scalable considering the fact that it allows the LHP to encrypt EHR data with a set of attributes (from one or more AAs) in such a way that a user who possesses secret keys corresponding

to the aforementioned attributes can successfully decrypt the data without requiring to have secret keys from all the AAs. In Chapter 6, we have extended the MA-CP-ABE construction to provide controlled access delegation and presented as our seventh ABAC construction, Scheme 7. Besides for being able to provide controlled access delegation, the Scheme 7 also provides selective delegatability which means that the delegatees will not be able to access other resources which are associated with the same access sub-structure as the resource for which the access is granted via delegation. In comparison to the existing delegatable ABE schemes, Scheme 7 exhibits superior delegation capabilities given that the property of selective delegatability is not supported in the existing schemes. Furthermore, all the existing schemes use a centralized TA to manage and issue attributes whereas the Scheme 7 uses a distributed multi-authority architecture to issue attributes which make our scheme more practical and conducive for a collaborative e-health environment.

## 7.2 Future Work

In this section, we briefly present a few potential research directions that have emerged in relation to the contributions made in this dissertation.

- In Scheme 2, we proposed a multi-show unlinkable credential scheme, and we have extended it to support controlled access delegation in Scheme 5. Although it is efficient in comparison to existing multi-show unlinkable credential schemes, we have shown that there is a possibility of colluding attributes embedded in credentials owned by two or more users since the verifier will not be able to recognize on its own that these credentials are not owned by the same entity. But, in Section 3.6, we have pointed out that it is possible for the verifier to communicate with the AAs which issued the credentials to get an acknowledgment whether the credentials are owned by the same entity. However, this is not an efficient mechanism, since the verifier needs to communicate with AAs during every access session. Therefore, incorporating an efficient mechanism to resist the possibility of collusion attacks appears to be an interesting direction.

- When multi-level attribute delegations are permitted, the trustworthiness of the delegated attribute is affected by the individual trustworthiness of all the entities in the associated delegation chain of the considered attribute. This suggests that the integration of a suitable trust model to evaluate the trust level of the attributes will certainly add more control over the process of ac-

cess delegation as well as enhancing the security of the underlying AC mechanism. Therefore, how the delegatable ABAC schemes proposed in Scheme 3, Scheme 4, Scheme 5 and Scheme 7 can be integrated with appropriate trust models appears as another important research topic.

- In Scheme 4, we proposed a delegatable ABAC scheme using blockchain technology. Although this induces the lowest end-user computational overhead in comparison to the other two proposed delegatable ABAC schemes (Scheme 3 and Scheme 5), it can only provide pseudo-anonymous guarantees to the users given that all transactions associated with a particular user's pseudo-identity can be traced and linked together in the blockchains. For instance, when a user is involved in a first level delegation of an attribute, the related delegation transaction block will have a hash pointer to the corresponding assignment block in the blockchain. However, if we can utilize the idea used to construct the fully decentralized Zerocash payment scheme in [90], it might be possible for us to remove the hash pointers and thereby achieve full anonymity. Investigating the possibility of utilizing this approach to Scheme 4 and evaluating its feasibility with end-user computational complexity could also be an interesting research direction.

- In the delegatable ABE construction proposed in Scheme 7, we require the support of the LHP to facilitate the delegation process via re-encrypting the ciphertext which allows the delegatees to decrypt the re-encrypted ciphertext. If we can also move the re-encryption process to HC without jeopardizing the security of the system, it will be possible for us to relieve the LHP from the delegation process completely. This is also a task that we plan to address in our future work.

# REFERENCES

[1] N. Dong, H. Jonker, and J. Pang, "Challenges in eHealth: From Enabling to Enforcing Privacy," in *Foundations of Health Informatics Engineering and Systems*, ser. Lecture Notes in Computer Science (LNCS). Springer Berlin Heidelberg, 2012, vol. 7151, pp. 195 – 206.

[2] M. Kay, M. O.-G. van Andel, K. Klint, and C. Tristram, *Building Foundations for eHealth: Report of the WHO Global Observatory for eHealth*. Geneva: WHO Press, 2006.

[3] I. Carrin, J. Fernndez-Alemn, and A. Toval, "Usable Privacy and Security in Personal Health Records," in *Human-Computer Interaction - INTERACT 2011*, ser. Lecture Notes in Computer Science (LNCS). Springer Berlin Heidelberg, 2011, vol. 6949, pp. 36 – 43.

[4] H. J. Cheong, N. Y. Shin, and Y. B. Joeng, "Improving Korean Service Delivery System in Health Care: Focusing on National E-health System," in *Proceedings of the International Conference on eHealth, Telemedicine, and Social Medicine*. IEEE, Feb. 2009, pp. 263 – 268.

[5] A. Lang and A. Mertes, "E-Health Policy and Deployment Activities in Europe," *Telemedicine and e-Health*, vol. 17, no. 4, pp. 262 – 268, 2011.

[6] A. Sheikh, T. Cornford, N. Barber, A. Avery, A. Takian, V. Lichtner, D. Petrakaki, S. Crowe, K. Marsden, A. Robertson, Z. Morrison, E. Klecun, R. Prescott, C. Quinn, Y. Jani, M. Ficociello, K. Voutsina, J. Paton, B. Fernando, A. Jacklin, and K. Cresswell, "Implementation and Adoption of Nationwide Electronic Health Records in Secondary Care in England: Final Qualitative Results from Prospective National Evaluation in "Early Adopter" Hospitals," *British Medical Journal*, vol. 343, 2011.

[7] I. Pagkalos, S. Kortesis, G. Pangalos, and M. Hassapidou, "Deployment of the NETC@RDS service: A New Step Towards a European E-health Space," in *Proceedings of the 3rd International Conference on Biomedical Engineering and Informatics*. IEEE, Oct. 2010, pp. 2511 – 2515.

[8] A. A. O'Kane, H. M. Mentis, and E. Thereska, "Non-static Nature of Patient Consent: Shifting Privacy Perspectives in Health Information Sharing," in *Proceedings of the Conference on Computer Supported Cooperative Work*. ACM, Feb. 2013, pp. 553 – 562.

[9] "Secondary User Service (SUS) - Health and Social Care Information Centre," 2015, Available: http://www.hscic.gov.uk/sus, Accessed 12.11.2015.

[10] "Newly available Health Data will support Medical Research and Patient Empowerment," 2011, Available: http://www.gov.uk/government/news/newly-available-health-data-will-support-medical-research-and-patient-empowerment, Accessed: 28.10.2015.

[11] A. Ghazvini and Z. Shukur, "Security Challenges and Success Factors of Electronic Healthcare System," *Procedia Technology*, vol. 11, pp. 212 – 219, 2013.

[12] US Department of Health and Human Services, "Standards for Privacy of Individually Identifiable Health Information; Final Rule," *45 Code of Federal Regulations*, vol. 78, no. 17, pp. 5566 – 5702, 2013.

[13] "Health Insurance Portability and Accountability Act," 2015, Available: http://www.hhs.gov/hipaa/for-professionals/index.html, Accessed: 02.10.2015.

[14] M. Scholl, K. Stine, J. Hash, P. Bowen, A. Johnson, C. D. Smith, and D. I. Steinberg, "An Introductory Resource Guide for Implementing the Health Insurance Portability and Accountability Act (HIPAA) Security Rule," *NIST Special Publication Revision 1*, vol. 800-66, 2008.

[15] C.-M. Yang, H.-C. Lin, P. Chang, and W.-S. Jian, "Taiwan's Perspective on Electronic Medical Records' Security and Privacy Protection: Lessons learned from HIPAA," *Computer Methods and Programs in Biomedicine*, vol. 82, no. 3, pp. 277 – 282, 2006.

[16] J. Mirkovic, E. Skipenes, E. Christiansen, and H. Bryhni, "Security and Privacy Legislation Guidelines for Developing Personal Health Records," in *Proceedings of the 2nd International Conference on eDemocracy eGovernment*. IEEE, Apr. 2015, pp. 77 – 84.

[17] A. Calder, *Implementing Information Security Based on ISO 27001/ISO 27002*. Van Haren Publishing, 2009.

[18] O. Par and E. Soysal, "Security Standards for Electronic Health Records," in *Proceedings of the IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*. IEEE, Aug. 2012, pp. 815 – 817.

[19] "Security Breaches 2005 - Present," 2005, Available: http://www.hhs.gov/hipaa/for-professionals/index.html, Accessed: 16.10.2015.

[20] Y. Chen and H. Xu, "Privacy Management in Dynamic Groups: Understanding Information Privacy in Medical Practices," in *Proceedings of the Conference on Computer Supported Cooperative Work*. ACM, Feb. 2013, pp. 541 – 552.

[21] H. S. G. Pussewalage and V. A. Oleshchuk, "An Attribute Based Access Control Scheme for Secure Sharing of Electronic Health Records," in *Proceedings of the 18th IEEE International Conference on E-health, Networking, Application and Services (IEEE HEALTHCOM'16)*. IEEE, Sep. 2016, pp. 526 – 531.

[22] ——, "An Efficient Multi-Show Unlinkable Attribute Based Credential Scheme for a Collaborative E-health Environment," in *Proceedings of the 3rd IEEE International Conference on Collaboration and Internet Computing (IEEE CIC'17)*. IEEE, Oct. 2017, pp. 421 – 428.

[23] M. Li, S. Yu, Y. Zheng, K. Ren, and W. Lou, "Scalable and Secure Sharing of Personal Health Records in Cloud Computing Using Attribute-Based Encryption," *IEEE Transactions on Parallel and Distributed Systems*, vol. 24, no. 1, pp. 131 – 143, 2013.

[24] H. S. G. Pussewalage and V. A. Oleshchuk, "Attribute Based Access Control Scheme With Controlled Access Delegation for Collaborative E-health Environments," *Journal of Information Security and Applications*, vol. 37, pp. 50 – 64, 2017.

[25] ——, "A Patient-Centric Attribute Based Access Control Scheme for Secure Sharing of Personal Health Records Using Cloud Computing," in *Proceedings of the 2nd IEEE International Conference on Collaboration and Internet Computing (IEEE CIC'16)*. IEEE, Nov. 2016, pp. 46 – 53.

[26] S. Bleikertz, M. Schunter, C. W. Probst, D. Pendarakis, and K. Eriksson, "Security Audits of Multi-tier Virtual Infrastructures in Public Infrastructure Clouds," in *Proceedings of the ACM Workshop on Cloud Computing Security Workshop*. ACM, Oct. 2010, pp. 93 – 102.

[27] B. Grobauer, T. Walloschek, and E. Stocker, "Understanding Cloud Computing Vulnerabilities," *IEEE Security and Privacy*, vol. 9, no. 2, pp. 50 – 57, 2011.

[28] S. Ruj, "Attribute Based Access Control in Clouds: A Survey," in *Proceedings of the International Conference on Signal Processing and Communications*. IEEE, Jul. 2014, pp. 1 – 6.

[29] R. S. Sandhu and P. Samarati, "Access Control: Principle and Practice," *IEEE Communications Magazine*, vol. 32, no. 9, pp. 40 – 48, 1994.

[30] H. Yang, "Cryptographic Enforcement of Attribute Based Authentication," Ph.D. dissertation, University of Agder, Jun. 2016, Available: http://hdl.handle.net/11250/2391521.

[31] R. J. Anderson, "A Security Policy Model for Clinical Information Systems," in *Proceedings of IEEE Symposium on Security and Privacy (IEEE S & P'96)*. IEEE, May 1996, pp. 30 – 43.

[32] H. C. Cankaya, "Access Control Lists," in *Encyclopedia of Cryptography and Security*. Springer US, 2011, pp. 9 – 12.

[33] R. S. Sandhu, E. J. Coyne, H. L. Feinstein, and C. E. Youman, "Role-Based Access Control Models," *IEEE Computer*, vol. 29, no. 2, pp. 38 – 47, 1996.

[34] R. S. Sandhu, "Role-Based Access Control," *Advances in Computers*, vol. 46, pp. 237 – 286, 1998.

[35] D. F. Ferraiolo, R. Sandhu, S. Gavrila, D. R. Kuhn, and R. Chandramouli, "Proposed NIST Standard for Role-Based Access Control," *ACM Transactions of Information and System Security*, vol. 4, no. 3, pp. 224 – 274, 2001.

[36] E. Bertino, P. A. Bonatti, and E. Ferrari, "TRBAC: A Temporal Role-Based Access Control Model," *ACM Transactions of Information and System Security*, vol. 4, no. 3, pp. 191 – 233, 2001.

[37] D. F. Ferraiolo, J. F. Barkley, and D. R. Kuhn, "A Role-Based Access Control Model and Reference Implementation Within a Corporate Intranet," *ACM Transactions of Information and System Security*, vol. 2, no. 1, pp. 34 – 64, 1999.

[38] H. Zhang, Y. He, and Z. Shi, "Spatial Context in Role-Based Access Control," in *Information Security and Cryptology - ICISC 2006*, ser. Lecture Notes in Computer Science (LNCS). Springer Berlin Heidelberg, Nov. 2006, vol. 4296, pp. 166 – 178.

[39] V. C. Hu, D. R. Kuhn, and D. F. Ferraiolo, "Attribute-Based Access Control," *IEEE Computer*, vol. 48, no. 2, pp. 85 – 88, 2015.

[40] X. Jin, R. Krishnan, and R. Sandhu, "A Unified Attribute-Based Access Control Model Covering DAC, MAC and RBAC," in *Proceedings of Data and Applications Security and Privacy XXVI*. Springer Berlin Heidelberg, Jul. 2012, pp. 41 – 55.

[41] V. Hu, D. F. Ferraiolo, D. R. Kuhn, R. N. Kacker, and Y. Lei, "Implementing and Managing Policy Rules in Attribute Based Access Control," in *Proceedings of the IEEE International Conference on Information Reuse and Integration*. IEEE, Aug. 2015, pp. 518 – 525.

[42] D. Servos and S. L. Osborn, "Current Research and Open Problems in Attribute-Based Access Control," *ACM Computer Surveys*, vol. 49, no. 4, pp. 1 – 45, 2017.

[43] H. S. G. Pussewalage and V. A. Oleshchuk, "Privacy Preserving Mechanisms for Enforcing Security and Privacy Requirements in E-health Solutions," *International Journal of Information Management*, vol. 36, no. 6, Part B, pp. 1161 – 1173, 2016.

[44] V. Hu, D. Ferraiolo, R. Kuhn, A. Schnitzer, K. Sandlin, R. Miller, and K. Scarfone, "Guide to Attribute Based Access Control (ABAC) Definition and Considerations," *National Institute of Standards Technology Special Publication*, vol. 800-162, 2014.

[45] A. Sahai and B. Waters, "Fuzzy Identity-Based Encryption," in *Advances in Cryptology - EUROCRYPT 2005*, ser. Lecture Notes in Computer Science (LNCS). Springer Berlin Heidelberg, 2005, vol. 3494, pp. 457 – 473.

[46] 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 (ACM CCS'06)*. ACM, Nov. 2006, pp. 89 – 98.

[47] A. Lewko and B. Waters, "New Proof Methods for Attribute-Based Encryption: Achieving Full Security through Selective Techniques," in *Advances in Cryptology - CRYPTO 2012*, ser. Lecture Notes in Computer Science (LNCS). Springer Berlin Heidelberg, 2012, vol. 7417, pp. 180 – 198.

[48] S. Rass and D. Slamanig, *Cryptography for Security and Privacy in Cloud Computing*. Norwood, MA, USA: Artech House, Inc., 2013.

[49] Department of Health and Human Services, "Security and Electronic Signature Standards," *Federal Register*, vol. 63, no. 155, pp. 43 241–43 280, 1998.

[50] G. H. M. B. Motta and S. S. Furuie, "A Contextual Role-Based Access Control Authorization Model for Electronic Patient Record," *IEEE Transactions on Information Technology in Biomedicine*, vol. 7, no. 3, pp. 202 – 207, 2003.

[51] D. M. Eyers, J. Bacon, and K. Moody, "OASIS Role-Based Access Control for Electronic Health Records," *IEE Proceedings - Software*, vol. 153, no. 1, pp. 16 – 23, 2006.

[52] M. Khan and K. Sakamura, "Toward a Synergy among Discretionary, Role-Based and Context-Aware Access Control Models in Healthcare Information Technology," in *Proceedings of the World Congress on Internet Security*. IEEE, Jun. 2012, pp. 66 – 70.

[53] J. Jin, G.-J. Ahn, M. J. Covington, and X. Zhang, "Access Control Model for Sharing Composite Electronic Health Records," in *Collaborative Computing: Networking, Applications and Worksharing*, ser. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering. Springer Berlin Heidelberg, 2009, vol. 10, pp. 340 – 354.

[54] R. Bhatti, K. Moidu, and A. Ghafoor, "Policy-Based Security Management for Federated Healthcare Databases (or RHIOs)," in *Proceedings of the International Workshop on Healthcare Information and Knowledge Management*. ACM, Nov. 2006, pp. 41 – 48.

[55] L. D. Martino, Q. Ni, D. Lin, and E. Bertino, "Multi-Domain and Privacy-Aware Role Based Access Control in eHealth," in *Proceedings of the 2nd International Conference on Pervasive Computing Technologies for Healthcare*. IEEE, Jul. 2008, pp. 131 – 134.

[56] R. Zhang, J. Liu, Z. Han, and L. Liu, "RBTBAC: Secure Access and Management of EHR Data," in *Proceedings of the International Conference on Information Society*. IEEE, Jun. 2011, pp. 494 – 499.

[57] F. Hansen and V. Oleshchuk, "Location-Based Security Framework for use of Handheld Devices in Medical Information Systems," in *Proceedings of the*

*4th IEEE International Conference on Pervasive Computing and Communications Workshops (IEEE PERCOMW'06)*. IEEE, Mar. 2006, pp. 565 – 569.

[58] T. Okamoto, "Cryptography Based on Bilinear Maps," in *Applied Algebra, Algebraic Algorithms and Error-Correcting Codes*, ser. Lecture Notes in Computer Science (LNCS). Springer Berlin Heidelberg, 2006, vol. 3857, pp. 35 – 50.

[59] R. A. Sahu and S. Padhye, "Efficient ID-Based Signature Scheme from Bilinear Map," in *Advances in Parallel Distributed Computing*, ser. Communications in Computer and Information Science (CCIS), vol. 203. Springer Berlin Heidelberg, 2011, pp. 301 – 306.

[60] H. Cohen, G. Frey, R. Avanzi, C. Doche, T. Lange, K. Nguyen, and F. Vercauteren, *Handbook of Elliptic and Hyperelliptic Curve Cryptography*. Boca Raton, FL: Chapman & Hall/CRC, 2005.

[61] K. S. McCurley, "The Discrete Logarithm Problem," in *Proceedings of the Symposia in Applied Mathematics*, Aug. 1989, pp. 49 – 74.

[62] I. F. Blake, X. Gao, R. C. Mullin, S. A. Vanstone, and T. Yaghoobian, "The Discrete Logarithm Problem," in *Applications of Finite Fields*, ser. The Springer International Series in Engineering and Computer Science (Communications and Information Theory). Springer US, 1993, vol. 199.

[63] P. Vullers, "Efficient Implementations of Attribute-Based Credentials on Smart Cards," Ph.D. dissertation, Radboud University Nijmegen, Nov. 2014, Available: http://www.cs.ru.nl/P.Vullers/publications/2014_phd_thesis.pdf.

[64] N. Guo, Y. Jin, and K. Yim, "Anonymous Credential-Based Privacy-Preserving Identity Verification for Business Processes," in *Proceedings of the 8th International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing*. IEEE, Jul. 2014, pp. 554 – 559.

[65] J. Camenisch and E. Van Herreweghen, "Design and Implementation of the Idemix Anonymous Credential System," in *Proceedings of the 9th ACM Conference on Computer and Communications Security (ACM CCS'02)*. ACM, Nov. 2002, pp. 21 – 30.

[66] S. A. Brands, *Rethinking Public Key Infrastructures and Digital Certificates: Building in Privacy*. Cambridge, MA, USA: MIT Press, 2000.

[67] C. Paquin and G. Zaverucha, "U-prove cryptographic specification v1.1 (revision 3)," Dec. 2013, Available: https://www.microsoft.com/en-us/research/publication/u-prove-cryptographic-specification-v1-1-revision-3/, Accessed: 01.03.2017.

[68] W. Mostowski and P. Vullers, "Efficient U-Prove Implementation for Anonymous Credentials on Smart Cards," in *Security and Privacy in Communication Networks*, ser. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering. Springer Berlin Heidelberg, 2012, vol. 96, pp. 243 – 260.

[69] J. Camenisch and A. Lysyanskaya, "A Signature Scheme with Efficient Protocols," in *Security in Communication Networks - SCN 2002*, ser. Lecture Notes in Computer Science (LNCS). Springer Berlin Heidelberg, 2003, vol. 2576, pp. 268 – 289.

[70] S. A. Pourbakhsh and R. Katti, "Efficient Attributes in Secure Credentials," in *Proceedings of the IEEE Pacific Rim Conference on Communications, Computers and Signal Processing*. IEEE, Aug. 2015, pp. 107 – 112.

[71] G. Fuchsbauer, C. Hanser, and D. Slamanig, "Structure-Preserving Signatures on Equivalence Classes and Constant-Size Anonymous Credentials," Cryptology ePrint Archive, Report 2014/944, 2014, Available: http://eprint.iacr.org/2014/944.

[72] J. Camenisch and A. Lysyanskaya, "Signature Schemes and Anonymous Credentials from Bilinear Maps," in *Advances in Cryptology - CRYPTO 2004*, ser. Lecture Notes in Computer Science (LNCS). Springer Berlin Heidelberg, 2004, vol. 3152, pp. 56 – 72.

[73] S. Ringers, E. Verheul, and J.-H. Hoepman, "An Efficient Self-Blindable Attribute-Based Credential Scheme," Cryptology ePrint Archive, Report 2017/115, 2017, Available: http://eprint.iacr.org/2017/115.

[74] D. Boneh, "The Decision Diffie-Hellman Problem," in *Algorithmic Number Theory*, ser. Lecture Notes in Computer Science (LNCS). Springer Berlin Heidelberg, 1998, vol. 1423, pp. 48 – 63.

[75] E. Barker, W. Barker, W. Burr, W. Polk, and M. Smid, "Recommendation for Key Management Part 1: General (Revision 3)," *National Institute of Standards and Technology*, vol. 800-57, 2012.

[76] H. S. G. Pussewalage and V. Oleshchuk, "Blockchain Based Delegatable Access Control Scheme for a Collaborative E-health Environment," in *Proceedings of the 1st IEEE International Conference on Blockchain (IEEE Blockchain'18)*. IEEE, Jul. 2018, pp. 1204 – 1211.

[77] E. Barka, R. Sandhu, and R. S, "A Role-Based Delegation Model and Some Extensions," in *Proceedings of the 23rd National Information Systems Security Conference*, 2000, pp. 101 – 114.

[78] L. Zhang, G.-J. Ahn, and B.-T. Chu, "A Rule-Based Framework for Role-Based Delegation and Revocation," *ACM Transactions on Information and System Security*, vol. 6, no. 3, pp. 404 – 441, 2003.

[79] X. Zhang, S. Oh, and R. Sandhu, "PBDM: A Flexible Delegation Model in RBAC," in *Proceedings of the 8th ACM Symposium on Access Control Models and Technologies (ACM SACMAT'03)*. ACM, Jun. 2003, pp. 149 – 157.

[80] E. Freudenthal, T. Pesin, L. Port, E. Keenan, and V. Karamcheti, "dRBAC: Distributed Role-Based Access Control for Dynamic Coalition Environments," in *Proceedings of the 22nd International Conference on Distributed Computing Systems*. IEEE, Jul. 2002, pp. 411 – 420.

[81] L. Zhang, G.-J. Ahn, and B.-T. Chu, "A Role-Based Delegation Framework for Healthcare Information Systems," in *Proceedings of the 7th ACM Symposium on Access Control Models and Technologies (ACM SACMAT'02)*. ACM, Jun. 2002, pp. 125 – 134.

[82] M. F. F. Khan and K. Sakamura, "A Smartcard-Based Framework for Delegation Management in Healthcare Access Control Systems," in *Proceedings of the IEEE Region 10 Conference*. IEEE, Nov. 2016, pp. 2739 – 2742.

[83] ——, "A Secure and Flexible E-Health Access Control System with Provisions for Emergency Access Overrides and Delegation of Access Privileges," in *Proceedings of the 18th International Conference on Advanced Communication Technology*. IEEE, Jan. 2016, pp. 541 – 546.

[84] J. Camenisch, M. Drijvers, and M. Dubovitskaya, "Practical UC-Secure Delegatable Credentials with Attributes and Their Application to Blockchain," in *Proceedings of the ACM Conference on Computer and Communications Security (ACM CCS'17)*. ACM, Nov. 2017, pp. 683 – 699.

[85] M. Barua, R. Lu, and X. Shen, "SPS: Secure Personal Health Information Sharing with Patient-Centric Access Control in Cloud Computing," in *Proceedings of the IEEE Global Communications Conference (IEEE GLOBECOM'13)*. IEEE, Dec. 2013, pp. 647 – 652.

[86] A. Narayanan, J. Bonneau, E. Felten, A. Miller, and S. Goldfeder, *Bitcoin and Crytocurrency Technologies*. Princeton, NJ: Princeton University Press, 2016.

[87] A. A. Antonopoulos, *Mastering Bitcoin*. Sebastopol, CA: O'Reilly Media Inc., 2014.

[88] F. Tian, "An Agri-food Supply Chain Traceability System for China based on RFID Blockchain Technology," in *Proceedings of the 13th International Conference on Service Systems and Service Management*. IEEE, Jun. 2016, pp. 1 – 6.

[89] S. Nakamoto, "Bitcoin: A Peer-to-peer Electronic Cash System," Available: https://bitcoin.org/bitcoin.pdf, Accessed: 15.01.2018.

[90] E. B. Sasson, A. Chiesa, C. Garman, M. Green, I. Miers, E. Tromer, and M. Virza, "Zerocash: Decentralized Anonymous Payments from Bitcoin," in *Proceedings of the IEEE Symposium on Security and Privacy (IEEE S & P'14)*. IEEE, May 2014, pp. 459 – 474.

[91] Y. Gao, X. Chen, Y. Sun, X. Niu, and Y. Yang, "A Secure Cryptocurrency Scheme based on Post-Quantum Blockchain," *IEEE Access*, 2018, Early Access.

[92] D. Vujii, D. Jagodi, and S. Rani, "Blockchain Technology, Bitcoin, and Ethereum: A Brief Overview," in *Proceedings of the 17th International Symposium INFOTEH-JAHORINA*. IEEE, Mar. 2018, pp. 1 – 6.

[93] S. Hohenberger and B. Waters, "Short and Stateless Signatures from the RSA Assumption," in *Advances in Cryptology - CRYPTO 2009*, ser. Lecture Notes in Computer Science (LNCS). Springer Berlin Heidelberg, 2009, vol. 5677, pp. 654 – 670.

[94] R. L. Rivest, A. Shamir, and L. Adleman, "A Method for Obtaining Digital Signatures and Public-key Cryptosystems," *Communications of the ACM*, vol. 21, no. 2, pp. 120 – 126, 1978.

[95] R. Cramer and V. Shoup, "Signature Schemes Based on the Strong RSA Assumption," *ACM Transactions on Information and System Security*, vol. 3, no. 3, pp. 161 – 185, 2000.

[96] M. Chase and A. Lysyanskaya, "On Signatures of Knowledge," in *Advances in Cryptology - CRYPTO 2006*, ser. Lecture Notes in Computer Science (LNCS). Springer, Berlin, Heidelberg, 2006, vol. 4117, pp. 78 – 96.

[97] M. Belenkiy, J. Camenisch, M. Chase, M. Kohlweiss, A. Lysyanskaya, and H. Shacham, "Randomizable Proofs and Delegatable Anonymous Credentials," in *Advances in Cryptology - CRYPTO 2009*, ser. Lecture Notes in Computer Science (LNCS). Springer, Berlin, Heidelberg, 2009, vol. 5677, pp. 108 – 125.

[98] G. Fuchsbauer, "Commuting Signatures and Verifiable Encryption," in *Advances in Cryptology - EUROCRYPT 2011*, ser. Lecture Notes in Computer Science (LNCS), K. G. Paterson, Ed. Springer, Berlin, Heidelberg, 2011, vol. 6632, pp. 224 – 245.

[99] H. Lin, Z. Cao, X. Liang, and J. Shao, "Secure Threshold Multi Authority Attribute Based Encryption without a Central Authority," in *Progress in Cryptology - INDOCRYPT 2008*, ser. Lecture Notes in Computer Science (LNCS). Springer, Berlin, Heidelberg, 2008, vol. 5365, pp. 426 – 436.

[100] M. Chase and S. S. M. Chow, "Improving Privacy and Security in Multi-Authority Attribute-Based Encryption," in *Proceedings of the 16th ACM Conference on Computer and Communications Security (ACM CCS'09)*. ACM, Nov. 2009, pp. 121 – 130.

[101] B. Waters, "Ciphertext-Policy Attribute-Based Encryption: An Expressive, Efficient, and Provably Secure Realization," in *Public Key Cryptography - PKC 2011*, ser. Lecture Notes in Computer Science (LNCS). Springer, Berlin, Heidelberg, 2011, vol. 6571, pp. 53 – 70.

[102] R. Ostrovsky, A. Sahai, and B. Waters, "Attribute-Based Encryption with Non-monotonic Access Structures," in *Proceedings of the 14th ACM Conference on Computer and Communications Security (ACM CCS'07)*. ACM, Oct. 2007, pp. 195 – 203.

[103] H. S. G. Pussewalage and V. A. Oleshchuk, "A Distributed Multi-Authority Attribute Based Encryption Scheme for Secure Sharing of Personal Health

Records," in *Proceedings of the 22nd ACM Symposium on Access control Models and Technologies (ACM SACMAT'17)*. ACM, Jun. 2017, pp. 255 – 262.

[104] S. S. Chow, "A Framework of Multi-Authority Attribute-Based Encryption with Outsourcing and Revocation," in *Proceedings of the 21st ACM Symposium on Access control Models and Technologies (ACM SACMAT'16)*. ACM, Jun. 2016, pp. 215 – 226.

[105] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-Policy Attribute-Based Encryption," in *Proceedings of the IEEE Symposium on Security and Privacy (IEEE S & P'07)*. IEEE, May 2007, pp. 321 – 334.

[106] L. Ibraimi, Q. Tang, P. Hartel, and W. Jonker, "Efficient and Provable Secure Ciphertext-Policy Attribute-Based Encryption Schemes," in *Information Security Practice and Experience*, ser. Lecture Notes in Computer Science (LNCS). Springer Berlin Heidelberg, 2009, vol. 5451, pp. 1 – 12.

[107] L. Ibraimi, M. Asim, and M. Petkovic, "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*. IEEE, Jun. 2009, pp. 71 – 74.

[108] M. Barua, X. Liang, R. Lu, and X. Shen, "ESPAC: Enabling Security and Patient-Centric Access Control for eHealth in Cloud Computing," *International Journal of Security and Networks*, vol. 6, no. 2/3, pp. 67 – 76, 2011.

[109] ——, "PEACE: An Efficient and Secure Patient-Centric Access Control Scheme for eHealth Care System," in *Proceedings of the IEEE Conference on Computer Communications Workshops (IEEE INFOCOM WKSHPS'11)*. IEEE, Apr. 2011, pp. 970 – 975.

[110] C. Danwei, C. Linling, F. Xiaowei, H. Liwen, P. Su, and H. Ruoxiang, "Securing Patient-Centric Personal Health Records Sharing System in Cloud Computing," *China Communications*, vol. 11, no. 13, pp. 121 – 127, 2014.

[111] S. Alshehri, S. P. Radziszowski, and R. K. Raj, "Secure Access for Healthcare Data in the Cloud Using Ciphertext-Policy Attribute-Based Encryption," in *Proceedings of the 28th IEEE International Conference on Data Engineering Workshops*. IEEE, Apr. 2012, pp. 143 – 146.

[112] C. J. Wang, X. L. Xu, D. Y. Shi, and W. L. Lin, "An Efficient Cloud-Based Personal Health Records System Using Attribute-Based Encryption and Anonymous Multi-Receiver Identity-Based Encryption," in *Proceedings of the 9th International Conference on P2P, Parallel, Grid, Cloud and Internet Computing*. IEEE, Nov. 2014, pp. 74 – 81.

[113] S. Ding, Y. Zhao, and H. Zhu, "Extending Fuzzy Identity-Based Encryption with Delegating Capabilities," in *Proceedings of the 6th IEEE Joint International Information Technology and Artificial Intelligence Conference*. IEEE, Aug. 2011, pp. 19 – 23.

[114] L. Ibraimi, M. Petkovic, S. Nikova, P. Hartel, and W. Jonker, "Ciphertext-Policy Attribute-Based Threshold Decryption with Flexible Delegation and Revocation of User Attributes (Extended Version)," 2009, Available: http://doc.utwente.nl/65471/, Accessed: 17.05.2016.

[115] M. Blaze, G. Bleumer, and M. Strauss, "Divertible Protocols and Atomic Proxy Cryptography," in *Advances in Cryptology - EUROCRYPT 1998*, ser. Lecture Notes in Computer Science (LNCS). Springer Berlin Heidelberg, 1998, vol. 1403, pp. 127 – 144.

[116] M. Mambo and E. Okamoto, "Proxy Cryptosystems: Delegation of the Power to Decrypt Ciphertexts," *IEICE Transactions on Fundamentals of Electronics Communications and Computer Sciences*, vol. 80, no. 1, pp. 54 – 63, 1997.

[117] 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 (ACM TISSEC)*, vol. 9, no. 1, pp. 1 – 30, 2006.

[118] C.-K. Chu and W.-G. Tzeng, "Identity-Based Proxy Re-encryption Without Random Oracles," in *Information Security - ISC 2007*, ser. Lecture Notes in Computer Science (LNCS). Springer Berlin Heidelberg, 2007, vol. 4779, pp. 189 – 202.

[119] M. Green and G. Ateniese, "Identity-Based Proxy Re-encryption," in *Applied Cryptography and Network Security - ACNS 2007*, ser. Lecture Notes in Computer Science (LNCS). Springer Berlin Heidelberg, 2007, vol. 4521, pp. 288 – 306.

[120] T. Matsuo, "Proxy Re-encryption Systems for Identity-Based Encryption," in *Pairing-Based Cryptography - PBC 2007*, ser. Lecture Notes in Computer Science (LNCS). Springer Berlin Heidelberg, 2007, vol. 4575, pp. 247 – 267.

[121] S. Guo, Y. Zeng, J. Wei, and Q. Xu, "Attribute-Based Re-encryption Scheme in the Standard Model," *Wuhan University Journal of Natural Sciences*, vol. 13, no. 5, pp. 621 – 625, 2008.

[122] X. Liang, Z. Cao, H. Lin, and J. Shao, "Attribute Based Proxy Re-encryption with Delegating Capabilities," in *Proceedings of the 4th ACM Symposium on Information, Computer and Communications Security (ACM ASIACCS'09)*. ACM, Mar. 2009, pp. 276 – 286.

[123] S. Luo, J. Hu, and Z. Chen, "Ciphertext Policy Attribute-Based Proxy Re-encryption," in *Information and Communications Security- ICICS 2010*, ser. Lecture Notes in Computer Science (LNCS). Springer Berlin Heidelberg, 2010, vol. 6476, pp. 401 – 415.