Enhanced Security and Privacy for Blockchain-enabled Electronic Medical Records in eHealth

by

Xiaoshuai Zhang

A thesis submitted to the University of London for the degree of
Doctor of Philosophy

School of Electronic Engineering and Computer Science
Queen Mary University of London
United Kingdom

November 2020

TO MY FAMILY

Abstract

Electronic medical records (EMRs) as part of an eHealth system are vital assets centrally managed by medical institutions and used to maintain up to date patients' medical histories. Such centralised management of EMRs may result in an increased risk of EMR damage or loss to medical institutions. In addition, it is difficult to monitor and control who can access their EMRs and for what reasons as eHealth may increasingly involve the use of IoT devices such as eHealth wearables and distributed networks. Blockchain is proposed as a promising method applied to support distributed data storage to maintain and share EMRs using its inherent immutability (forgery resistance). However, the original blockchain design cannot restrict unauthenticated or unauthorised data access for use as part of EMR management.

Therefore, two novel authorisation schemes to enhance the security and privacy of blockchain use for EMRs are proposed in this work. The first one can omit the agent layer (gateway) to authorise users' access to blockchain-enabled EMRs with block level granularity, whilst maintaining compatibility with the underlying Blockchain data structure. Then, an improved scheme is proposed to implement multiple levels of granularity authorisation, whilst supporting flexible data queries. This scheme dispenses with the need to use a public key infrastructure (PKI) in authorisation and hence reduces the resource cost of computation and communication. Furthermore, to realise privacy preservation during authorisation, a challenge-response anonymous authorisation is proposed that avoids the disclosure of users' credentials when authorising data access requests.

Compared with the baseline schemes, the proposed authorisation schemes can decrease the time consumption of computation and data transmission and reduce the transmitted data size so that they can be used in low-resource IoT devices applied to blockchain-enabled EMRs as demonstrated in performance experiments. In addition, theoretical

validations of correctness demonstrate that the proposed authorisation schemes work correctly.

Acknowledgments

First and foremost, I would like to express my deepest gratitude to my primary supervisor, Dr. Stefan Poslad for opening me a door to the research field of blockchain and providing me with the invaluable guidance as well as the persistent encouragement, helping me go through all difficulties I confronted during my study. He has given me not only many constructive suggestions directed and inspired me in the research, but also innumerable wise opinions for my daily life with his patient and colossal knowledge. Without his constant support, my research could not be possible to be completed.

I would like to express my special appreciation to my second supervisor Professor Pasquale Malacaria who taught me a lot about organisation and highlights for presentations. His attitude towards research and broad research interests always boosts me to pursue higher standard work. I am also thankful to my independent assessor Dr. Fabrizio Smeraldi for offering numerous concrete comments to my reports. They all grounded my further advance; I could not ask for a better supervision panel. Meanwhile, I would like to express my profound gratitude to Dr. Kok Keong (Michael) Chai for continuously supporting me in papers and projects.

I would also like to thank all my friends and colleagues: Dr. Qiao Cheng, Dr. Yuhang Dai, Dr. Tianhao Guo, Dr. Xiangyu He, Dr. Eng Tseng Lau, Dr. Chao Liu, Dr. Yuanwei Liu, Dr. Shan Luo, Dr. Zixiang Ma, Dr. Rana Massoud, Dr. Wenxuan Mou, Dr. Zhaoyang Xu, Dr. Jinze Yang, Dr. Xingjian Zhang, Dr. Jingjing Zhao, Fan Fu, Tianwei Hou, Raphael Kim, Anna Li, Wanlin Li, Zhaoliang Luan, Haoran Qi, Bizhu Wang, Bang Wu, Tingting Xie, Meng Xu, Guangyuan Zhang and so on, for the academic discussions and for all the fun we have had in the last four years. There are many other people that I am not mentioning here stayed besides me in some ways, I love them all.

Last but not least, I would like to say big thank you to my beloved fiancée Jing Sun

and both our families. You are the most important people in my life and thanks for supporting me and always accepting me whenever I feel depressed or frustrated. It is you give me the confidence and courage to go through the up and downs. Without your love and support over the years, none of this would have been possible.

N.B. This work was funded in part by a PhD scholarship funded jointly by the China Scholarship Council (CSC) and Queen Mary University of London (QMUL), and I'm very grateful for both their support throughout to have enabled me to take this major step in my major career.

Table of Contents

List of Figures

List of Tables

List of Publications

1. **X. Zhang**[1], C. Liu, K. K. Chai, and S. Poslad, "A Challenge-Response Assisted Authorisation Scheme for Data Access in Permissioned Blockchains," in *Sensors*, vol. 20, no. 17, 4681, 2020, doi:10.3390/s20174681.

2. **X. Zhang**, C. Liu, S. Poslad and K. K. Chai, "A Provable Semi-Outsourcing Privacy Preserving Scheme for Data Transmission from IoT Devices," in *IEEE Access*, vol. 7, pp. 87169-87177, 2019, doi: 10.1109/ACCESS.2019.2925403.

3. **X. Zhang** and S. Poslad, "Blockchain Support for Flexible Queries with Granular Access Control to Electronic Medical Records (EMR)," *2018 IEEE International Conference on Communications (ICC)*, Kansas City, MO, USA, 2018, pp. 1-6, doi: 10.1109/ICC.2018.8422883.

4. **X. Zhang**, S. Poslad and Z. Ma, "Block-based Access Control for Blockchain-based Electronic Medical Records (EMRs) Query in eHealth," *2018 IEEE Global Communications Conference (GLOBECOM)*, Abu Dhabi, United Arab Emirates, 2018, pp. 1-7, doi: 10.1109/GLOCOM.2018.8647433.

[1]Google Scholar profile: `https://scholar.google.com/citations?user=Oo46BqAAAAAJ`

List of Abbreviations

ABAC	Attribute-based Access Control
ABE	Attribute-based Encryption
ABS	Attribute-based Signature
ACL	Access Control List
Adv	Advantage
AES	Advanced Encryption Standard
AI	Artificial Intelligence
BDH	Bilinear Diffie-Hellman
BFT	Byzantine Fault Tolerance
BP	Bilinear Pairing
BLS	Boneh–Lynn–Shacham
BTC	Bitcoin
CDH	Computational Diffie-Hellman
CSP	Communicating Sequential Processes
CT	Computed Tomography
DAO	Decentralised Autonomous Organization
DDH	Decisional Diffie–Hellman
DDoS	Distributed Denial-of-service
DES	Data Encryption Standard
DH	Diffie-Hellman

DL	Discrete Logarithm
DLP	Discrete Logarithm Problem
DoS	Denial-of-service
DSA	Digital Signature Algorithm
ECC	Elliptic Curve Cryptography
ECCDH	Elliptic-curve Computational Diffie-Hellman
ECDH	Elliptic-curve Diffie-Hellman
ECDLP	Elliptic-curve Discrete Logarithm Problem
ECDSA	Elliptic Curve Digital Signature Algorithm
EMR	Electronic Medical Record
EMRs	Electronic Medical Records
ETH	Ethereum
EUF-CMA	Existential Unforgeability under Chosen Message Attack
EVM	Ethereum Virtual Machine
FDR	Failures Divergences Refinement
gcd	greatest common divisor
IBE	Identity-based Encryption
IBM	International Business Machines (Corporation)
IBS	Identity-based Signature
ICT	Information and Communications Technologies
ID	Identity
IEEE	Institute of Electrical and Electronics Engineers
iff	if and only if
IND-CCA	Indistinguishability under Chosen-Ciphertext Attack
IoT	Internet of Things
IP	Internet Protocol
IPFS	InterPlanetary File System
IPsec	Internet Protocol Security
JAAS	Java Authentication and Authorisation Service

KGC	Key Generation Centre
MIRACL	Multiprecision Integer and Rational Arithmetic Cryptographic Library
MITM	Man-in-the-middle
MPC	Multi-party Computation
MRI	Magnetic Resonance Imaging
NIST	National Institute of Standards and Technology
OW-CCA	One-Way against Chosen Ciphertext Attack
PBFT	Practical Byzantine Fault Tolerance
PC	Personal Computer
PET	Positron Emission Tomography
PGP	Pretty Good Privacy
PKC	Public-key Cryptography
PKG	Private Key Generator
PKI	Public Key Infrastructure
PoET	Proof of Elapsed Time
PoL	Proof of Luck
PoS	Proof of Stake
PoW	Proof of Work
Pr	Probability
QMUL	Queen Mary, University of London
RBAC	Role-based Access Control
RSA	Rivest-Shamir-Adleman cryptosystem
SOPs	Standard Operation Procedures
SPOF	Single point of failure
SSH	Secure Shell
SSL	Secure Sockets Layer
SSS	Shamir's Secret Sharing
TTP	Trusted Third Party
TCP	Transmission Control Protocol

TLS	Transport Layer Security
TPM	Trusted Platform Module
XFT	Cross Fault Tolerance
XOR	Exclusive OR
ZKP	Zero-Knowledge Proof

Chapter 1

Introduction

The value of the global eHealth market is estimated to grow to USD 39 billion by 2025 [1]. Electronic medical records (EMRs) are a key element of this. In eHealth, EMRs are a fundamental component to store patients' medical history containing not only diagnoses and prescriptions but also medical check results, a person's health status, and other medical data [2]. Therefore, as an important private asset of patients, EMRs need to be kept safe when used by centralised authorities such as hospitals, clinics, healthcare centres and so on. Under this centralised management pattern of electronic medical record (EMR) use, many different stakeholders such as patients, doctors, pathologists, administrators and other medical staff may need to access EMR data.

More recently, facilitated by the advances in Internet of Things (IoT), which can offer seamless platforms for people and objects to connect with each other much easier, there are more and more IoT devices such as medical alert bracelets, wireless pacemakers and implanted insulin pumps that act as sources as eHealth data and are becoming integrated into eHealth applications to realise numerous beneficial medical functions (e.g., health index monitoring and continuous medicine supplement) [3]. Such devices can be used outside healthcare centres, enabling not just health providers, but also health users, to monitor their own health status anywhere and anytime [4]. However, when IoT is

exploited to advance eHealth, EMRs tend to be highly distributed in terms of who (local doctor, hospital doctor, administrator etc.) has modified what and where this is done (e.g., in hospital, doctor surgery, care homes, private homes etc.) [5]. Therefore, management of EMRs including secure storage and access control, is a crucial requirement for eHealth, yet this is very challenging to achieve because of the highly distributed and fragmented nature of EMRs and the range of providers and users who are authorised to access them [2]. In addition, such changes to EMRs can overburden such centralised EMR management in terms of computation and communication costs in managing such distributed changes to EMRs. Hence, a shift from centralised EMR management to a more decentralised one to reduce the workload of central authorities and to allow patients to control their own private data more transparently, is being investigated [6].

1.1 Research motivation

A promising approach to achieve decentralised EMR management is an innovation from the cryptocurrency field - blockchain. Blockchain is a distributed and immutable ledger that enables transparent transactions originating from a cryptocurrency Bitcoin proposed by Satoshi Nakamoto [7]. The immutable feature means a blockchain is resistant to tampering with its block data because once recorded, the data in any given block cannot be modified retroactively without amending all subsequent blocks (i.e., forgery resistance). A key benefit of blockchain is the decentralised consensus mechanism to realise anonymous, secure and accountable transactions. Therefore, blockchain-enabled EMRs can be a desirable solution to store EMRs in clouds or by patients themselves distributedly so that medical institutions can reduce the cost of EMR management i.e., it is not required to construct and maintain large-scale data centres to store all patients' EMRs [8, 9]. Meanwhile, the consensus mechanism can be utilised as an access control method so that every patient can participate to determine whether a user can access to its requested EMRs or not instead of thoroughly centralised access control by an authority [10].

Although blockchain has its unique merits to implement distributed EMR management, there are still many challenges that should be considered in depth by stakeholders such as hospitals, healthcare centres and medical research institutions to realise access control for blockchain-enabled EMRs. If conventional EMRs evolve into blockchain-enabled EMRs, the designed access control schemes should adapt to the blockchain structure and support the participation of patients. To be specific, if the access control is designed to be more precise to control the access (query) to the data of specific attributes in a block, the data structure of blockchain should be discussed because it is different compared with the conventional data structure in databases. Meanwhile, since different stakeholders such as different medical staff with different roles may need different types of access control to different parts of EMRs, fine-grained and flexible granularity control should be considered in access control schemes. Fine-grained means the granularity control can support patients to authorise data access to not only certain blocks but also the specific attributes in different blocks in the blockchain-enabled EMRs (see Section 4.2.3). Flexible implies that the granularity control can allow users to query the data of many blocks, the data of the specific attributes in different blocks and a mixture of the former both. For example, the attending physician of a patient may wish to view all medical records (i.e., a block or many blocks) of this patient but a nurse should only handle his (or her) prescriptions, injection doses and allergy information (i.e., specific attributes in some blocks). For epidemiologists and other medical analysts, the information they utilise for epidemiological and pathological analyses should be restricted i.e., as they only require medical information (e.g., diagnoses, ages, genders, medical images and used drugs) but other personal information such as names, social numbers and addresses are not needed.

Furthermore, eHealth may involve IoT devices that with limited or low resource, for computation, communication and storage resources, and power, hence, cryptographic methods applied in the design of access control to EMRs involving such IoT devices should be designed to be similarly low-resource. Hence, the motivation to research

and innovate access control for IoT-driven blockchain-enabled EMRs in this work is summarised as follows.

- As an aspect of access control, authorisation with fine-grained and flexible granularity control has not been discussed sufficiently and concretely to allow patients to precisely protect their private data in blockchain-enabled EMRs (see Section 2.2.3 and 2.3).
- Key existing access control schemes are not compatible with the blockchain structure as they only consider the data structure of traditional EMRs (see Section 2.2.1.1 and 2.2.3).
- In current solutions for such access control, the adopted cryptographic components are not lightweight enough (see Section 2.1.1.4 and 2.3) to ensure that the resource-constrained IoT devices applied in eHealth can execute these solutions smoothly constrained by low consumption of computation, communication and power.

1.2 Research aim and objectives

The research aim of this work is to enhance the security and privacy of blockchain-enabled EMRs by designing access control to them to realise authenticated and authorised EMR access. Meanwhile, the access control needs to fit the blockchain structure in blockchain-enabled EMR queries. In addition, the designed access control schemes should be lightweight to be used by resource-constrained IoT devices in eHealth.

There are three objectives to be achieved in this work. The major objective is to enhance the security and privacy of blockchain-enabled EMRs by cryptographic access control schemes. The access control schemes designed in this work utilise lightweight cryptographic approaches as the second associated objective to support the use of resource-constrained IoT devices applied in eHealth. The third associated objective focuses on privacy preservation to limit the exposure of private information during the access control to blockchain-enabled EMRs.

1.3 Research scope and contributions

The research scope of this work is novel access control schemes for blockchain-enabled EMRs. The blockchain-enabled EMRs considered in this research consist of many participants (nodes), a distributed ledger and a consensus mechanism to support distributed EMR recording and storage, and to be immune to forgery to ensure the data integrity of EMRs. In this work, there are two patterns for the proposed access control schemes to interact with the blockchain-enabled EMRs. Firstly, the access control schemes are implemented in an independent entity out of the blockchain-enabled EMRs i.e., users should pass the access control provided by this entity before requesting data from the blockchain-enabled EMRs (see the proposed schemes in Section 4). Secondly, the access control schemes are executed by certain participants in the blockchain-enabled EMRs (see the proposed schemes in Sections 3 and 5). In this way, the result (allow or deny) of access control can be determined by the consensus mechanism used in the blockchain-enabled EMRs.

The novelty of the contributions of this work is three-fold. Implementing access control with fine-grained and flexible granularity control for blockchain-enabled EMRs is the first novel contribution. The proposed access control schemes can authorise fine-grained and flexible data access to share blockchain-enabled EMRs compatible with mainstream blockchain data structures. This implies that the achieved granularity control can support users to request not only data from several entire blocks (i.e., block-level granularity, see Section 3), but also the data pertaining to specific attributes within blocks such as a patient's gender, age and prescriptions or to support both of these (see Section 4). Secondly, to improve the performance of the proposed access control schemes, the cryptographic components utilised in the proposed access control schemes are designed to be suitable for low-resource IoT devices. Therefore, in the validated performance experiments with a Raspberry Pi 2 as the low-resource IoT device, the resource consumption for computation and communication is reduced to be suitable for resource-constrained IoT devices applied in eHealth. The third novel contribution is to restrict patients' pri-

vate information exposed in the nodes in the blockchain network during the authorisation because these nodes may be compromised and become untrusted (see Section 5).

1.4 Thesis organisation

The overview of the work in this thesis is summarised in Figure 1.1. To enhance the security and privacy for blockchain-enabled EMRs, access control (including authentication and authorisation) and privacy preservation are discussed in this thesis. The content of each chapter is summarised as follows.

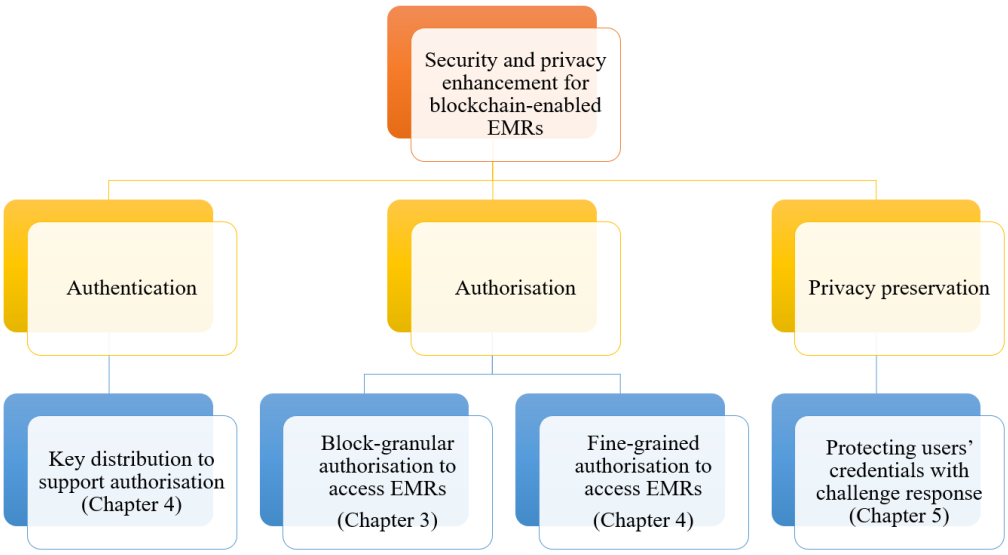

Figure 1.1: The overview of the novel work in the rest of this thesis.

Chapter 2 covers first a broad overview of eHealth, blockchain and blockchain-enabled EMRs concepts. Then, a comprehensive investigation of the state-of-the-art security and privacy concerns for eHealth, blockchain and the access control for blockchain-enabled EMRs is carried out. Furthermore, three open challenges of access control in blockchain-enabled EMRs are highlighted to clarify the research motivation by analysing the limitations of the state-of-the-art access control solutions for blockchain-enabled

EMRs.

Chapter 3 proposes an authorisation scheme for block-level queries for use in blockchain-enabled EMRs. This scheme can authorise users' access to different blocks in blockchain-enabled EMRs based upon ECC (elliptic curve cryptography) as a lightweight cryptographic foundation. The results of the experiments show that the proposed scheme can reduce the time cost of computation and communication when compared with baseline schemes.

Chapter 4 addresses a novel design for fine-grained and flexible authorisation to access blockchain-enabled EMRs with a cloud-based key distribution scheme. The proposed authorisation scheme can support three types of fine-grained data queries including blocks (block-level query), certain attributes in different blocks (attribute-level query), and a mixture of the former two query types (mixed query). The key distribution scheme supports the authorisation to authenticate the identities of the participants (e.g., IoT devices) and distribute the tokens used in their authorisation. Meanwhile, Shamir secret sharing (SSS) is utilised to enable encryption and decryption so that the proposed scheme dispenses with the need to use a public key infrastructure (PKI) for the key exchange of encryption and hence decreases the communication overhead. Furthermore, unlike many common key distribution schemes heavily relying on trusted clouds, the proposed key distribution scheme can adapt to untrusted clouds to distribute the keys to the participants and ensure the confidentiality and integrity of the keys. As the underlying applied cryptographic principles, ECC and SSS, can cut down the data size for transmission, and decrease the time cost of computation and communication in comparative experiments.

Chapter 5 considers the reliability of the pre-selected (consensus) nodes in the permissioned blockchain network for EMR sharing to suggest an anonymous authorisation scheme using a challenge-response strategy without exposing users' identities. The idea stems from using a zero-knowledge proof (ZKP) to avoid exposing real users' credentials to the pre-selected nodes based upon the use of hash, SSS and ECC. Data access authorisation does not require sharing users' actual credentials with pre-selected nodes

in the permissioned blockchain network, avoiding such sensitive information leakage. If the pre-selected nodes are compromised by attacks, thus becoming untrusted, real users' credentials cannot be leaked, preserving users' private information. The experimental results demonstrate that the proposed design can reduce not only the time consumption of computation for authorisation, encryption and decryption, and communication but also the transaction cost of authorisation when compared with baseline schemes.

Chapter 6 presents the conclusion and some thoughts for future work.

Chapter 2

Fundamental Concepts and a Critical Analysis of the State of the Art

This chapter first provides the introduction to some fundamental concepts and several prevalent access control methods in public-key cryptography for understanding the work next. Meanwhile, the background of eHealth, blockchain and blockchain-enabled eHealth is presented to provide a broad vision of the related research scope. Then, a comprehensive investigation of the state of the art related to the security concerns in eHealth and blockchain and blockchain-enabled eHealth is illustrated. Then, for the specific research field of this work, a comparison and analysis of access control for blockchain-enabled EMRs is undertaken and the challenges for current state of the art are undertaken.

2.1 Preliminaries and background

2.1.1 Public-key cryptography

Public-key cryptography (PKC), or asymmetric cryptography, is a cryptosystem (cryptographic system) that uses two pairs of keys: public keys, which may be disseminated widely, and private keys, which are known only to the owner to safeguard the owner's privacy. The generation of such keys depends upon cryptographic algorithms based upon mathematical difficult problems that produce one-way functions. Realising effective security only requires keeping the private key secret. Furthermore, the public key can be openly distributed without compromising encryption security [11].

Such a cryptosystem can be utilised to encrypt or sign messages. To be specific, any person can encrypt a message using the receiver's public key, but the encrypted message can be decrypted only with the receiver's private key. On the other hand, a message's owner can sign this message with the owner's private key to create a digital signature of the message, and anyone can confirm the integrity of the signed message through verifying the signature with the owner's public key [12]. Nowadays, public-key algorithms have become a fundamental security ingredient in modern cryptosystems, applications and protocols such as Digital Signature Algorithm (DSA), Elliptic Curve Digital Signature Algorithm (ECDSA), Transport Layer Security (TLS), Internet Protocol Security (IPsec), Secure Shell (SSH), Open Secure Sockets Layer (OpenSSL) and Pretty Good Privacy (PGP), assuring the confidentiality, authenticity and non-reputability (integrity) of electronic communications and data storage [13].

In this section, the fundamental public-key cryptosystems are introduced and compared including Rivest–Shamir–Adleman (RSA), Diffie-Hellman (DH) and ECC based upon two different mathematical difficult problems. While RSA cryptosystem relies on the mathematical difficult problem of big integer factorisation [14], DH and ECC cryptosystems are implemented based upon a discrete logarithm problem (DLP) [15–17].

2.1.1.1 RSA

RSA is a public-key cryptosystem that relies on the computational difficulty of factoring the product of two large enough prime integers. Since solving the problem of big integer factorisation is still an open question, there are no published methods to compromise an RSA cryptosystem if a large enough key is applied.

In the textbook style, an RSA cryptosystem consists of two phases, key generation and encryption/decryption (or sign/verify), which are described as follows.

Algorithm 1 Key generation

Require: two large primes p, q
$n = p \cdot q$
$\phi(n) = (p-1) \cdot (q-1)$
choose an integer e, $1 < e < \phi(n)$ and $gcd(e, \phi(n)) = 1$, (i.e., e and $\phi(n)$ are coprime)
determine $d \equiv e^{-1} (mod\ \phi(n))$
return (n, e, d)

In Algorithm 1, *gcd* means greatest common divisor, the symbol $\equiv$ denotes modular congruence and $\phi(n)$ represents Euler's totient function [18]. Meanwhile, d is the modular multiplicative inverse of e modulo $\phi(n)$. After (n, e, d) are generated, (n, e) can be published as the public keys whilst d is the private key that should be kept secretly by the user who generates these keys with Algorithm 1. Then, the encryption and decryption algorithms are described in Algorithm 2 and 3 with the keys (n, e) and d respectively.

Algorithm 2 Encryption

Require: a plain integer message m $(0 < m < n)$, public keys (n, e)
$c = m^e\ (mod\ n)$
return c

Algorithm 3 Decryption

Require: a ciphertext c $(c < n)$, private key d and public key n
$m = c^d\ (mod\ n)$
return m

The correctness of RSA encryption/decryption can be briefly illustrated by $c^d = m^{ed} = m^{k\phi(n)+1} = m(m^{k\phi(n)}) = m(m^{\phi(n)})^k \equiv m(1)^k \equiv m\ (mod\ n)$ based upon Euler's

theorem [19], where m is an integer ($0 < m < n$) and $k \in \mathbb{Z}^*$. Note that to implement a digital signature, a user can use d (instead of e) to sign a message m to generate a signature c by following Algorithm 2. Then, one user can verify the signature c by computing c^e (instead of c^d in Algorithm 3) and compare its result with the received message m.

As for the performance, since RSA is a relatively high computation algorithm, it is not commonly used to directly encrypt bulk data [20]. More often, RSA is utilised to generate shared keys for symmetric key cryptography such as AES (Advanced Encryption Standard) that is then used to encrypt (or decrypt) bulk data.

2.1.1.2 Diffie–Hellman

Diffie-Hellman (or Diffie–Hellman key exchange) is one of the earliest practical protocols of public key exchange over a public channel implemented upon the discrete logarithm problem in a finite cyclic group. A general description of the DH protocol is presented as follows with a schematic diagram shown in Figure 2.1.

1. Alice and Bob agree on a multiplicatively finite cyclic group G with a prime order n and a generator $g \in G$.

2. Alice randomly selects an integer a $(1 < a < n)$, and then computes and sends $A = g^a \ (mod\ n)$ to Bob.

3. Bob selects a random integer number b $(1 < b < n)$, and then computes and sends $B = g^b \ (mod\ n)$ to Alice.

4. Alice computes $B^a = (g^b)^a = g^{ab} \ (mod\ n)$.

5. Bob computes $A^b = (g^a)^b = g^{ab} \ (mod\ n)$.

6. Both Alice and Bob now possess the common element $K = g^{ab}$ which can be used as the shared secret key.

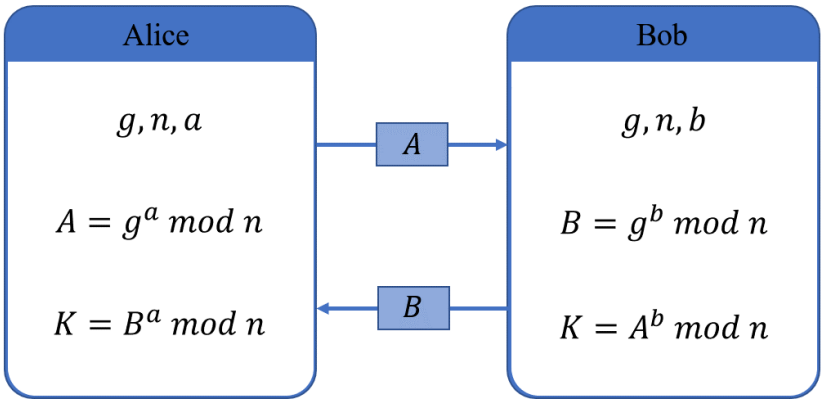

Figure 2.1: The schematic diagram of DH protocol.

The security behind DH protocol depends on the difficulty (hardness) of calculating g^{ab} with using g, g^a, g^b, which is called the computational Diffie-Hellman (CDH) problem. Note that there are another two hardness problems for the discrete logarithm (DL) called the DL problem (DLP) and the decisional Diffie–Hellman problem (DDH). DLP represents the hardness of determining a by given g, g^a. DDH means the hardness of determining if $g^c = g^{ab} (i.e.,\ c = ab)$ holds by given g, g^a, g^b, g^c. Since the CDH problem (as well as DLP and DDH) has not been proven to be completely unsolvable in discrete mathematics [17], the formal CDH assumption should be stated as given below when the CDH problem is applied in designing cryptographic schemes.

CDH Assumption. Let G be a multiplicatively finite cyclic group with a prime order n and a generator $g \in G$. For any two random integers a, b $(1 < a, b < n)$, any probabilistic polynomial-time algorithm $\mathcal{A}$ computes g^{ab} with its advantage (Adv):

$$Adv_{\mathcal{A}}^{CDH} = Pr[c = g^{ab} | c = \mathcal{A}(g, g^a, g^b)],$$

where Pr represents probability. The CDH assumption can hold if for any probabilistic polynomial-time algorithm $\mathcal{A}$, when its advantage $Adv_{\mathcal{A}}^{CDH}$ is negligible.

2.1.1.3 Elliptic curve cryptography

Elliptic curve cryptography (ECC) is an approach to public-key cryptography based upon elliptic curves over finite fields. Compared with RSA and DH, ECC allows smaller keys to provide equivalent security [21] that will be further discussed in the next section (see Section 2.1.1.4). In cryptography, ECC can be utilised to implement key exchange, digital signatures, and many other tasks.

Currently, there are two major cryptographic operations widely used in cryptography based upon two different types of elliptic curves. One is scalar multiplication operation used on non-singular elliptic curves and the other one is bilinear pairing (BP) operation used on supersingular elliptic curves. Next, the two types of elliptic curves and their corresponding cryptographic operations are introduced briefly.

- **Scalar multiplication**

Scalar multiplication is the most common operation in ECC, which relies on non-singular elliptic curves over finite fields. Such a non-singular elliptic curve can be defined as:

$$E_p(a,b) : y^2 \equiv x^3 + ax + b \ (mod\ p),$$

where a, b are two integer coefficients and p represents the finite field $\mathbb{F}_p$, for the elliptic curve $E_p(a,b)$ that the coefficients are in (i.e., $mod\ p$). All the integer points on $E_p(a,b)$ over $\mathbb{F}_p$ can be organised as an additive group G with an operator $+$, an identity (point at infinity) O, and an inverse $-$. Note that $\forall P \in G,\ P + O = O + P = P$; for a $P = (x,y) \in G$, $-P = (x,-y)$ and $P + (-P) = (x,y) + (x,-y) = O$.

According to the above definitions, $\forall P(x_1,y_1), Q(x_2,y_2) \in G$, calculating $P + Q$ can be discussed by three conditions:

1. $x_1 \neq x_2$, draw a straight line through P, Q and get another point of intersection $-R$ on E, the inverse of $-R$ is the result of $P + Q$ (i.e., $R = P + Q$) shown in Figure 2.2.(1);

2. $x_1 = x_2$ and $y_1 = -y_2$, $P + Q = O$ based upon the definition of the inverse O is shown in Figure 2.2.(2);

3. $x_1 = x_2$ and $y_1 = y_2$, draws a tangent line over the point P and gets a point of intersection $-R$ on E, the inverse of $-R$ is the result of $P + Q$ (i.e., $R = P + Q = P + P = 2P$) as shown in Figure 2.2.(3).

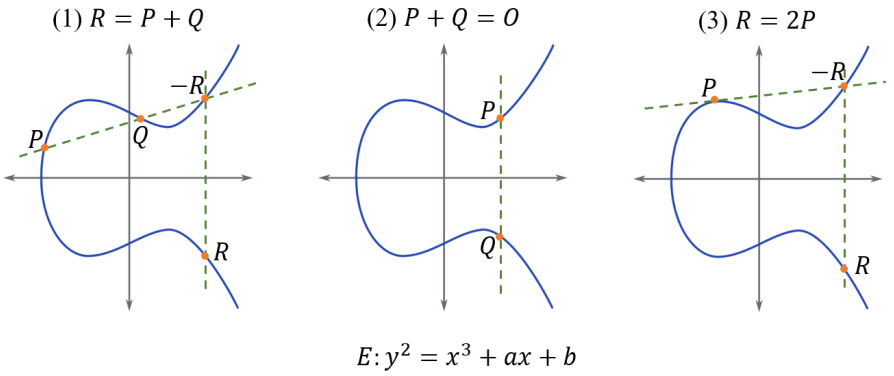

Figure 2.2: The examples of addition $(+)$ on the elliptic curve E.

Hence, the scalar multiplication of a point $P \in G$ can be expressed as:

$$\underbrace{P + P + ... + P}_{n} = nP,$$

where P is added n times on the curve E. Based upon scalar multiplication, the DH key exchange protocol can be implemented on an elliptic curve to form the elliptic-curve Diffie-Hellman (ECDH) protocol. To be specific, Alice computes and sends $A = aP$ to Bob, and Bob computes and sends $B = bP$ to Alice. Then, Both Alice and Bob can obtain the shared key $K = abP$ by computing aB (Alice) or bA (Bob).

Furthermore, since the group G is in the finite field $\mathbb{F}_p$, the CDH and DDH problems, and DLP can be extended to the group G on the elliptic curve E with the operation of scalar multiplication. Similar to the CDH assumption, the elliptic-curve computational Diffie-Hellman assumption can be described as follows.

Elliptic-curve Computational Diffie-Hellman (ECCDH). Let $E_p(a, b)$ be a

cryptographic secure elliptic curve with the prime field $\mathbb{F}_p$. For any point $P \in E_p(a,b)$ and two random integers $u, v \in_R \mathbb{Z}_p^*$, any probabilistic polynomial-time algorithm $\mathcal{A}$ computes uvP with its advantage:

$$Adv^{ECCDH}_{\mathcal{A},E_p(a,b)} = Pr[c = uvP | u \in_R \mathbb{Z}_p^*, c = \mathcal{A}(P, uP, vP)].$$

The ECCDH assumption can hold if for any probabilistic polynomial-time algorithm $\mathcal{A}$, its advantage $Adv^{ECCDH}_{\mathcal{A},E_p(a,b)}$ is negligible.

Under the base of the ECCDH assumption, numerous cryptographic algorithms, schemes and protocols have been proposed to secure data, communications, identities and so on in modern cryptography. One of the most famous cryptographic algorithms utilising the ECCDH assumption is the Elliptic Curve Digital Signature Algorithm (ECDSA) [22]. ECDSA allows users to generate and verify the signature of specific data (message) to realise the protection of data integrity. There are two phases in ECDSA including signature generation and signature verification briefly described in Algorithm 4 and Algorithm 5.

Algorithm 4 Signature generation

Require: a secure elliptic curve E with a prime order n and a base point G
Require: a message m to be sent
Require: a secure hash function $H : \{0,1\}^* \rightarrow \{0,1\}^{|n|}$ ($|n|$ is the bit length of the prime order n)

1. generate a key pair (d, P), where $d \in [1, n-1]$ is randomly selected as the private key and $P = dG$ is the public key
2. compute $e = H(m)$
3. select a random integer $k \in [1, n-1]$
4. compute the point $(x, y) = kG$ on E, where G is the base point
5. compute $r = x \ mod \ n$

if $r = 0$ **then** GOTO 3

6. compute $s = k^{-1}(e + rd) \ (mod \ n)$

if $s = 0$ **then** GOTO 3

return (r, s) as the signature of m

Algorithm 5 Signature verification

Require: a secure elliptic curve E with a prime order n and a base point G
Require: the received public key P, message m and signature (r, s)
Require: a secure hash function $H : \{0,1\}^* \rightarrow \{0,1\}^{|n|}$ ($|n|$ is the bit length of the order n)

if $P = O$ **or** $P \notin E$ **or** $nP \neq O$ **then**
  return *error*
end if
if $r \notin [1, n-1]$ **or** $s \notin [1, n-1]$ **then**
  return *error*
end if
1. compute $e = H(m)$
2. compute $u_1 = es^{-1}\ mod\ n$ and $u_2 = rs^{-1}\ mod\ n$
3. compute the point $(x_1, y_2) = u_1G + u_2P$ on E
if $r \equiv x_1\ mod\ n$ **then**
  return *valid*
else
  return *invalid*
end if

The correctness of ECDSA is implied in computing

$$\begin{aligned} & u_1G + u_2P \\ & = u_1G + u_2dG \\ & = (u_1 + u_2d)G \\ & = (es^{-1} + rs^{-1}d)G \end{aligned}$$

$$
\begin{aligned}
&= (e + rd)s^{-1}G \\
&= (e + rd)[k^{-1}(e + rd)]^{-1}G \\
&= (e + rd)(e + rd)^{-1}(k^{-1})^{-1}G \\
&= kG.
\end{aligned}
$$

Therefore, the signature (r, s) is valid because $(x_1, y_1) = u_1G + u_2P = kG = (x, y)$ can result in $x_1 \equiv x \equiv r \ (mod\ n)$.

Even though the hardness problems and assumptions in ECC are equivalent to that in DL, ECC has its unique advantages when compared with RSA (based upon the hardness of big integer factorisation) and DH (based upon the hardness of DL). The key length used in ECC is much shorter than that used in RSA and DH under the equivalent security level [21]. Furthermore, ECC has a better computational efficiency because the time consumption of scalar multiplication operation is much less than that of general modular exponentiation used in RSA and DH [23]. Therefore, for the eHealth scenario involving any resource-constrained IoT devices deployed, ECC is a more suitable approach for cryptography to be adopted for designing security schemes to safeguard eHealth services efficiently.

- **Bilinear pairing**

Bilinear pairing (BP) is another prevalent operation on elliptic curves that is widely applied to realise different security purposes (e.g., group signature [24] and data aggregation [25]) in modern cryptography.

In most BP-related studies, the BP operation can be sculpted with three cyclic groups G_1, G_2 and G_T. Let G_1, G_2 be two additive cyclic groups of the prime order n, and G_T another multiplicative cyclic group of the same prime order n. A map: $e : G_1 \times G_2 \rightarrow G_T$ can be an admissible bilinear pairing (BP operation) iff (if and only if) the following properties are satisfied.

- **Bilinearity:** $\forall a, b \in \mathbb{Z}_n^*,\ \forall P, R \in G_1$, and $Q, S \in G_2$,
  $e(P, Q+S) = e(P,Q)e(P,S)$,
  $e(P+R, Q) = e(P,Q)e(R,Q)$,
  $e(aP, bQ) = e(P,Q)^{ab}$.

- **Non-degeneracy:** There exists $P \in G_1$ and $Q \in G_2$ such that $e(P,Q) \neq 1$.

- **Computability:** $\forall P \in G_1,\ Q \in G_2$, there exists an efficient algorithm to compute $e(P,Q)$.

Note that there are two types of BP distinguished by the elliptic curves that the groups G_1 and G_2 are generated from: symmetric BP on supersingular elliptic curves and asymmetric BP on ordinary elliptic curves. In the implementation of symmetric BP, G_1, G_2 are the same additive cyclic group on a supersingular elliptic curve i.e., $G_1 = G_2$, but in the implementation of asymmetric BP, G_1, G_2 are two different additive cyclic groups on two different ordinary elliptic curves with the same order. Meanwhile, G_T is the base field of the applied elliptic curves.

A simple example of utilising BP to construct cryptographic schemes is a signature scheme called Boneh–Lynn–Shacham (BLS) signature scheme [26], which can be briefly presented as the following four phases.

1. **Setup**

   Let $e : G \times G \to G_T$ be a non-degenerate, efficiently computable, bilinear pairing, where G is an additive cyclic group on an elliptic curve E and G_T is the group of E's base field. Meanwhile, G and G_T has the same prime order r and g is the generator of G.

2. **Key generation**

   The private key is denoted by x, a random integer selected from the interval $[0, r-1]$. The holder of the private key publishes the public key, g^x.

3. **Sign**

Given the private key x and a plain message m, the signature sent to the receiver is $\sigma = h^x$, where h is the hash value of the bitstring m as $h = H(m)$.

4. **Verify**

   Given the received message m, its signature σ and the public key g^x of the m's sender, the signature verification is to check if $e(\sigma, g) = e(H(m), g^x)$ holds.

Note that the security of the BLS scheme relies on the CDH problem. If σ is real signature of the message m, the correctness of the BLS scheme can be stated by $e(\sigma, g) = e(h^x, g) = e(H(m)^x, g) = e(H(m), g)^x = e(H(m), g^x)$. In terms of the three hardness problems in DL, DLP and CDH problems still hold under the BP operation. Furthermore, there is another hardness problem can hold under the BP operation i.e., bilinear Diffie-Hellman (BDH) problem. The BDH problem can be stated as: given $e : G_1 \times G_2 \rightarrow G_T$ and $\{P, aP, bP, cP\} \in G_1$, where $a, b, c \in \mathbb{Z}_n^*$, computing $e(P, P)^{abc}$ is unsolvable in polynomial time. However, the DDH problem becomes solvable by comparing $e(aP, bP)$ with $e(cP, P)$ with given e and $\{P, aP, bP, cP\} \in G_1$. If $c = ab$ holds, $e(aP, bP) = e(cP, P)$ should hold because $e(cP, P) = e(P, P)^c = e(P, P)^{ab} = e(aP, bP)$. This new feature has led to BP being one of the most popular topics discussed and utilised to construct numerous cryptographic schemes to serve various fields such as bill aggregation in smart grids, group/ring signature for integrity verification, and identity-based/attribute-based encryption (IBE/ABE) in multi-party computation (MPC).

However, an obvious drawback of the BP operation is computational efficiency. Many evaluations related to the time consumption of BP indicate that the BP operation is much more time-consuming and inefficient to be computed [27, 28]. Furthermore, BP requires higher energy consumption when compared with other public key operations in computation [23, 29]. On average, the time and energy cost of one BP operation is about five to ten times that of one scalar multiplication operation on the same elliptic curves [20, 30]. Therefore, the application of BP should be limited or replaced with other energy-efficient operations (e.g., scalar multiplication) in designing the cryptographic schemes for resource-constrained devices (e.g., certain IoT devices) to help reduce resource con-

sumption and save energy.

2.1.1.4 Computational performance comparison

In this part, I compare the actual time cost of three mainstream cryptographic operations including modular exponentiation, scalar multiplication and bilinear pairing when they are computed by a Raspberry Pi 2[1] (ARM Cortex-A7 processor running at 900MHz) as a real resource-constrained IoT device. Note that modular exponentiation is used by both RSA and DH whilst scalar multiplication and bilinear pairing are used in ECC. The experiments are implemented with a cryptographic toolkit called MIRACL (Multiprecision Integer and Rational Arithmetic Cryptographic Library) [31] under the equivalent security level, 128-bit security (see Table 2.1), recommended by National Institute of Standards and Technology (NIST) [21].

Table 2.1: Security level and equivalent key size (bits).

Security level	Cryptographic operation		
	Modular exponentiation	Scalar Multiplication	Bilinear pairing
80	1024	160	160
112	2048	224	224
128	**3072**	**256**	**256**
192	7680	384	384
256	15360	512	512

According to the selected 128-bit security level, the modulo used in modular exponentiation is 3072 bits. Meanwhile, the secure elliptic curves selected for scalar multiplication and bilinear pairing are secp256k1[2] and BN12[3], respectively. Furthermore, a personal computer (PC) with Intel i7 processor running at 4.2 GHz is used to conduct the same experiments as the reference. The three operations, modular exponentiation, scalar multiplication, and bilinear pairing are denoted by $\mathcal{O}_{exp}$, $\mathcal{O}_{mul}$ and $\mathcal{O}_{pair}$, respectively. The results of the time cost (millisecond) on Raspberry Pi 2 and PC are shown in Table 2.2.

It is clear that the Raspberry Pi 2 requires much more time to finish the computa-

[1] https://www.raspberrypi.org/products/raspberry-pi-2-model-b/
[2] https://safecurves.cr.yp.to/equation.html
[3] https://github.com/miracl/MIRACL/blob/master/source/curve/pairing/bn_pair.cpp

Table 2.2: Time cost (millisecond) of the three cryptographic operations on two platforms.

Platform	$\mathcal{O}_{mul}$	$\mathcal{O}_{exp}$	$\mathcal{O}_{pair}$
Raspberry Pi 2	7.771	403.536	72.613
PC	0.417	12.808	3.261

tion of the same cryptographic operation when compared with the baseline PC system because the computational resource of the Raspberry Pi 2 is quite limited (i.e., resource-constrained). Moreover, in Raspberry Pi 2, the time cost of the operations $\mathcal{O}_{pair}$ and $\mathcal{O}_{exp}$ is approximately 10 times and 50 times that of the operation $\mathcal{O}_{mul}$ demonstrating that the computational efficiency of modular exponentiation and bilinear pairing is much lower than that of scalar multiplication under the same security level and running conditions. These results and the conclusion have been confirmed in many other studies [20, 23, 27–30, 32]. Therefore, utilising ECC with the operation of scalar multiplication is a better choice in cryptography in order to design time-saving (lightweight) security schemes for eHealth services when using ICT (information communication technologies) resource-constrained devices.

2.1.2 Advanced access control primitives

To understand and discuss the various access control implementations, I introduce several cryptographic access control methods for PKC in this section.

2.1.2.1 Identity-based encryption

Identity-based encryption (IBE), is an important structure to underpin identity-based cryptography [33]. Identity-based cryptography allows any participant to generate a public key from certain known identity information such as an e-mail address or a social number. A schematic description of identity-based cryptography is demonstrated with the following three phases.

1. A trusted third party, called the private key generator (PKG), first generates the master private key and then publishes a master public key. Meanwhile, the master

private key should be kept secretly by the private key generator.

2. Given the master public key, any participant can use the identity to compute a public key by combining the master public key with the identity.

3. To obtain a corresponding private key to the public key, the owner of the identity involved in the public key can give the identity to the private key generator, which uses the master private key to generate the corresponding private key for the identity.

Under the described background of identity-based cryptography, IBE means that a sender can encrypt a message with the receiver's public key generated from the receiver's identity information and the master public key. To decrypt the encrypted message from the sender, the receiver first submits its identity information to the private key generator and retrieves its private key (decryption key) from the trusted private key generator. Then, the receiver can perform the decryption with the obtained private key. One of the earliest IBE implementations is Boneh–Franklin scheme [34], which is constructed based upon BP and BDH problem. This IBE scheme can be illustrated as follows.

- **Setup**

  1. Let $e : G_1 \times G_1 \rightarrow G_2$ be a non-degenerate, efficiently computable, bilinear pairing, where G_1 is an additive cyclic group on an elliptic curve E and G_2 is the group of E's base field. Meanwhile, G_1 and G_2 has the same prime order q and P is a base point of G_1.

  2. The private key generator chooses a random master private key $mk = s \in \mathbb{Z}_q^*$ then computes the master public key $K_{pub} = sP$.

  3. Two secure hash functions $H_1 : \{0,1\}^* \rightarrow G_1^*$ and $H_2 : G_2 \rightarrow \{0,1\}^n$ are published to all the participants, where n is a fixed bit length.

  4. The plaintext space and the ciphertext space are defined as $\mathcal{M} = \{0,1\}^n$ and $\mathcal{C} = G_1^* \times \{0,1\}^n$, respectively.

- **Extract**

  To generate the public key PK_{ID} for the identity $ID \in \{0,1\}^*$, the private key generator computes $PK_{ID} = H_1(ID)$. When the authentic owner of the identity ID requests the private key, the private key generator computes the private key $d_{ID} = s \cdot PK_{ID}$ then sends d_{ID} back to the identity owner.

- **Encrypt**

  Give a message $m \in \mathcal{M}$, the sender can generate the ciphertext c by following steps:

  1. compute $PK_{ID} = H_1(ID)$;
  2. select a random integer $r \in \mathbb{Z}_q^*$;
  3. compute $Q_{ID} = e(PK_{ID}, K_{pub})$;
  4. compute $c = (rP, m \oplus H_2(Q_{ID}^r))$.

- **Decrypt**

  Given the ciphertext $c = (u, v) \in \mathcal{C}$, the receiver can retrieve the plaintext m by computing $m = v \oplus H_2(e(d_{ID}, u))$ with its private key d_{ID}.

The correctness of the decryption in Boneh–Franklin scheme can be deduced by

$$\begin{aligned} v \oplus H_2(e(d_{ID}, u)) &= v \oplus H_2(e(sPK_{ID}, rP)) \\ &= v \oplus H_2(e(PK_{ID}, P)^{sr}) \\ &= v \oplus H_2(e(PK_{ID}, sP)^{r}) \\ &= v \oplus H_2(e(PK_{ID}, K_{pub})^{r}) \\ &= m \oplus H_2(Q_{ID}^r) \oplus H_2(Q_{ID}^r) \\ &= m. \end{aligned}$$

When observing the encrypting phase in the Boneh–Franklin scheme, there appears

to be an advantage for IBE i.e., every sender can compute the receiver's public key by itself with the identity information given by the receiver. Therefore, IBE can eliminate the infrastructure needed for public key distribution to reduce the communication cost. For example, in the Boneh–Franklin scheme, the private key generator does not need to publish the receiver's public key to all the senders. The authenticity of the receiver's public key is guaranteed implicitly as long as the transmission of the generated private key to the corresponding sender is kept secure. Note that the private key generator should be a trusted entity to check users' identities and to ensure users' public keys computed by it are authentic. Furthermore, if there is a finite number of users in an IBE cryptosystem, the private key generator can delete the master private key after every user obtains its private key.

However, this design still introduces several drawbacks. If the master private key is deleted, all the current users' public and private keys are always valid i.e., it is impossible to revoke or update the users' keys. On the other hand, there is an implicit key escrow problem in IBE cryptosystems which implies that the private key generator knows all the users' private keys, but in general public-key cryptosystems, the private keys are usually generated by the users themselves. Therefore, if the private key generator is compromised, all the public and private key pairs and all the encrypted messages are also compromised because the master private key is exposed to an attacker. The last point is that using BP operations, this can increase the computational cost of the users' devices especially for resource-constrained devices (see Section 2.1.1.3 and 2.1.1.4).

2.1.2.2 Attribute-based encryption

Attribute-based encryption (ABE) is a variation of IBE to control the encryption with attributes and policies instead of identity information in IBE [35]. Some policies are used to describe the relations between different attributes. In ABE, policies to access medical records can be sculpted using binary decision trees [36]. There are mainly two types of ABE schemes, key-policy attribute-based encryption (KP-ABE) [37] and

ciphertext-policy attribute-based encryption (CP-ABE) [38, 39]. The difference between KP-ABE and CP-ABE lies on how they utilise attributes and policies in private keys and ciphertext.

In KP-ABE, the ciphertext is encrypted with an attribute i.e., each attribute corresponds to a set of different ciphertext. Meanwhile, the private keys are determined by the defined policies. If a user has enough attributes to satisfy the policies (access trees) when it requests to access the ciphertext related to the specific attribute, the private key generator can provide the user with the corresponding private key (for the specific attribute used to encrypt the ciphertext) to allow it to decrypt the requested ciphertext. Therefore, if a user wants to decrypt the ciphertext encrypted with three different attributes, it needs to satisfy the policies with its possessed attributes to obtain three different private keys to decrypt the corresponding ciphertext. The application scenarios of KP-ABE involve log management (e.g., different users can access different parts of the system log), paid content access control, etc., [40, 41].

In contrast, in CP-ABE, the ciphertext is encrypted with a policy i.e., each policy corresponds to a set of different ciphertext, but the private keys are generated based upon the attributes. If a user has sufficient attributes to satisfy a policy, the private key generator can produce a private key generated by the attributes the user owns to allow the user to decrypt the ciphertext encrypted with the policy. Compared with KP-ABE, CP-ABE is more flexible because the policy can be customised dynamically with respect to different requirements. Furthermore, a user can use only one private key to decrypt all the ciphertext encrypted with the same policy. This primitive can be used for encryption sharing [42] and access control [43] to decrease the number of the private keys [44].

However, since ABE stems from IBE, ABE also inherits IBE's drawbacks. Therefore, in order to apply IBE and ABE to cryptosystems, the private key generator should be protected carefully, to avoid it becoming compromised. Meanwhile, the key escrow problem also needs to be addressed to ensure the confidentiality and integrity of the users' private keys used in generation and transmission, through using other cryptographic

components such as certificate and secret sharing.

2.1.2.3 Trapdoor

A trapdoor is a classic design component for authorisation in cryptography. Basically, the secret's owner can use the trapdoor to authorise the decryption or other operations to the ciphertext such as outsourcing computing in clouds. Furthermore, different trapdoors can restrict different operations to the ciphertext to achieve more fine-grained authorisation [45].

For example, the general blockchain structure is shown in Figure 2.3, where ts represents timestamp, h is the hash value, and the integrity is implemented by hashing as

$$\begin{cases} h_1 = Hash(data_1, ts_1), \\ h_i = Hash(data_i, ts_i, h_{i-1}),\ i = 2...n. \end{cases}$$

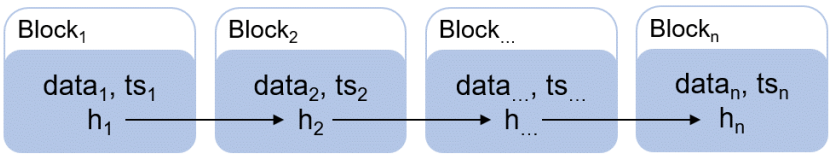

Figure 2.3: The general structure of blockchain.

A simple trapdoor design for the general blockchain structure can be implemented by adding a secret value α in computing hash values such as

$$\begin{cases} h_1 = Hash(data_1, ts_1, \alpha) \\ h_i = Hash(data_i, ts_i, h_{i-1} \oplus \alpha),\ i = 2...n. \end{cases}$$

If someone tries to verify $Block_i$, it should obtain the secret value α from the block's owner, i.e., the verification should be authorised by the block's owner. Otherwise, the hash value h_i cannot be verified if only the public information $data_i, ts_i$, and h_i are known but the secret value α is unknown.

2.1.2.4 Shamir secret sharing

Shamir's secret sharing (SSS) [46] is a common method to realise access control [47, 48]. In SSS, a secret y is divided into n shares and shared among n shareholders. If any t or more than t shares are given, it is able to reconstruct the secret y, but with fewer than t shares, it cannot reconstruct the secret.

Generally, SSS scheme is constructed based upon Lagrange interpolating polynomials. There are n shareholders $\mathcal{U} = \{U_1, ..., U_n\}$ and a large random prime q. The scheme consists of the following two algorithms *SSS.Generation* and *SSS.Reconstruction*:

• SSS.Generation(q, y)

This algorithm takes the prime q and the secret $y \in \mathbb{Z}_q$ and does the following:

1. Pick a polynomial $f(x)$ of degree $t-1$ randomly: $f(x) = a_0 + a_1 x + ... + a_{t-1}x_{t-1}$ $(mod\ q)$, where the secret $y = a_0 = f(0)$ and all coefficients $a_0, a_1, ..., a_{t-1}$ are random in $\mathbb{Z}_q$.

2. Compute all shares: $y_i = f(x_i)$ $(mod\ q)$ for $i = 1, ..., n$, where $x_i \in \mathbb{Z}_q$ are picked randomly.

3. Output a list of n points $\{(x_1, y_1), (x_2, y_2), ..., (x_n, y_n)\}$. Each share y_i is distributed to the corresponding shareholder U_i privately. Note that x_i need not be kept secretly.

• SSS.Reconstruction$(q, (x_{i_1}, y_{i_1}), ..., (x_{i_t}, y_{i_t}))$

This algorithm takes the prime q and any t points $\{(x_{i_1}, y_{i_1}), ..., (x_{i_t}, y_{i_t})\}$ as inputs, it reconstructs and outputs the secret y as

$$y = f(0) = \sum_{i \in A} \Delta_i y_i \ (mod\ q),$$

where

$$\Delta_i = \prod_{j \in A/\{i\}} \frac{x_j}{x_j - x_i} \ (mod\ q)$$

are the Lagrange interpolation coefficients and $A = \{i_1, ..., i_t\} \subseteq \{1, ..., n\}$.

According to the description of SSS, the major operation used in SSS is modular exponentiation in a finite field. However, the security of SSS is established based upon the hardness of reconstructing the correct Lagrange interpolating polynomial with the shares fewer than t [46], which differs from the hardness problems underlying RSA and DH. It has been proven that the size of the used finite field is equivalent to the security level of SSS i.e., the security level is 128 bits if the size of q is 128 bits [49, 50]. Therefore, the modular exponentiation of SSS is quite efficient in terms of computation because of its much smaller modulo space (finite field) based upon Table 2.1 and Table 2.2. This advantage implies that SSS can be a promising method to be utilised to realise access control in the eHealth scenario that includes low-resource devices.

2.1.2.5 Zero-knowledge proof

Zero-knowledge proof (ZKP) was first proposed by Shafi Goldwasser, Silvio Micali, and Charles Rackoff in 1989 [51]. ZKP is an interactive method for one party to prove that a statement is true. The highlight of ZKP is to realise the proof without revealing anything other than the statement is true (i.e., no leakage of the statement-related information). In cryptography, a zero-knowledge proof system should satisfy the following three properties [51, 52]:

- **Completeness**

  If the statement is true, the honest verifier (i.e., one following the protocol properly) will be convinced of this fact by the honest prover;

- **Soundness**

  If the statement is false, no cheating prover can convince the honest verifier that it is true;

- **Zero-knowledge**

  If the statement is true, no cheating verifier learns anything other than the fact

that the statement is true. In other words, just knowing the statement (not the secret) is sufficient to imagine a scenario showing that the prover knows the secret. This is formalised by showing that every cheating verifier, given only the statement to be proved (and no access to the prover), can produce a transcript that "looks like" an interaction between the honest prover and the cheating verifier.

An abstract example of ZKP showed in Figure 2.4[4]. is a game between Victor (green) and Peggy (purple). In this example, Peggy has an uncovered the secret word (key) used to open a magic door in a cave. The cave is shaped like a ring with the entrance on one side and the magic door blocking the opposite side. Victor wants to know whether Peggy knows the secret word; but Peggy, being a very private person, does not want to reveal her knowledge (the secret word) to Victor or to reveal the fact of her knowledge to the world in general. They play the following game.

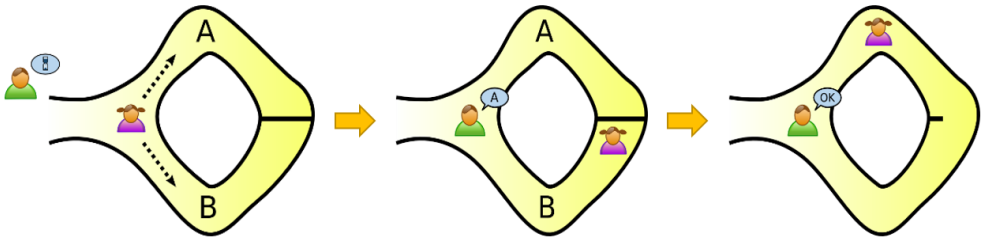

Figure 2.4: An abstract example for ZKP.

First, Victor waits outside the cave and Peggy goes in. Peggy picks either path A or B, but Victor is not allowed to see which path she takes at this time. Then, Victor enters the cave and shouts a random name of the path he wants her to use to return, either A or B.

Proving Peggy really knows the magic word is easy: she opens the door and returns along the selected path by Victor. However, suppose Peggy did not know the word. Then, she would only be able to return by the named path if Victor gave the name of the same path by which she had entered. Since Victor would choose A or B at random, Peggy would have a 50% chance of guessing correctly. If they repeated this trick many times,

[4]Figure source: https://en.wikipedia.org/wiki/Zero-knowledge_proof

say 50 times in a row, Peggy's chance of successfully anticipating all of Victor's requests would become significantly low (or negligible). Thus, if Peggy repeatedly appears at the exit Victor names, he can conclude that it is probable that Peggy does in fact know the secret word.

An example of the game's implementation with RSA is called Fiat-Shamir protocol [53] presented as follows.

- **Setup**

  Let T be a trusted third party (TTP). T selects and publishes an RSA modulus $n = p \times q$, but keeps p and q secret. Peggy first selects a secret $s \in [1, n-1]$ coprime to n. Then, Peggy computes $v = s^2 \ mod \ n$. Finally, Peggy registers v with T as her public key.

- **Construct**

  Each of the t rounds of the game has three messages with the following forms:

  1. Peggy chooses a random number $r \in [1, n-1]$ and sends $x = r^2 \ mod \ n$ to Victor;

  2. Victor randomly selects a challenge bit $e = 0$ or $e = 1$, and sends e to Peggy;

  3. Peggy computes and sends $y = r \cdot s^e \ mod \ n$ to Victor. Note that either $y = r$ holds if $e = 0$, or $y = r \cdot s$ holds if $e = 1$.

- **Verify**

  Victor rejects the proof if $x = 0$ or $y = 0$, or accepts upon verifying $y^2 \equiv x \cdot v^e \pmod{n}$.

These steps are iterated t rounds sequentially and independently. Victor only accepts the proof that Peggy knows s if all t rounds return success. As for the correctness, if $e = 0$, $y^2 \equiv r^2 \cdot s^0 \equiv r^2 \equiv x \ (mod \ n)$; if $e = 1$, $y^2 \equiv r^2 \cdot s^2 \equiv x \cdot v \equiv x \cdot v^e \ (mod \ n)$. Even though ZKP is a powerful component to realise access control such as an identity check

for authentication and certificate verification for both authentication and authorisation, it obviously increases the size of the exchanged messages in the interaction (t rounds) so that the communication overhead can be much higher when ZKP is applied. Therefore, utilising ZKP to realise security aims for the resource-constrained IoT devices in eHealth should be considered cautiously to avoid a higher energy cost caused by the heavy communications for the ZKP interaction. This is because the larger size of the exchanged messages used in ZKP interactions take a longer time for data transmission (e.g., use Wi-Fi to transmit data). It means the microprocessor used for data transmission needs to run for a longer time in normal working status and the running frequency is much higher than that in idle status or power-saving mode leading to a higher energy cost.

2.1.2.6 Equality test

Equality test, introduced by Yang et al. in 2010 [54], is a special kind of public-key component which allows one to check whether two ciphertexts encrypted under different public keys yet contain the same message. Since this method compares the ciphertexts to determine the corresponding plaintexts are identical or not, it can be a possible method to implement the authorisation through an untrusted third party. Since most of current equality test schemes are constructed based upon BP with CDH problem, I introduce a simplified example of BP-based equality test according to the design of Yang et al. [54].

- **Setup**

  1. Let G_1, G_2 denote two groups of the prime order q, g a generator of G_1, $e : G_1 \times G_1 \rightarrow G_2$ a bilinear pairing map, and H a secure hash function.

  2. Select $x \in \mathbb{Z}_q^*$ and compute $y = g^x$. The public key $pk = y$ and the private key $sk = x$.

- **Encrypt**

  To encrypt a plaintext $m \in G_1$, select a random integer $r \in \mathbb{Z}_q^*$ then compute $U = g^r, V = m^r, W = H(U, V, y^r) \oplus (m||r)$. The ciphertext is $C = (U, V, W)$.

- **Equality test**

Given two ciphertexts $C_1 = (U_1, V_1, W_1)$ and $C_2 = (U_2, V_2, W_2)$ of two messages m_1, m_2 with two key pairs (x_1, y_1) for C_1 and (x_2, y_2) for C_2, if $e(U_1, V_2) = e(U_2, V_1)$ holds, $m_1 = m_2$; otherwise, $m_1 \neq m_2$.

- **Decrypt**

  To decrypt a ciphertext $C = (U, V, W)$, compute $m||r = H(U, V, U^x) \oplus W$. If $m \in G_1 \wedge r \in \mathbb{Z}_q^* \wedge U = g^r \wedge V = m^r$, this algorithm outputs m; otherwise, outputs error.

If $m_1 = m_2$ holds, the correctness of the equality test lies on $e(U_1, V2) = e(g^r, m_2^r) = e(g, m_2)^{r^2} = e(g, m_1)^{r^2} = e(g^r, m_1^r) = e(U_2, V_1)$. Besides, the correctness of the decryption is that $H(U, V, U^x) \oplus W = H(U, V, (g^r)^x) \oplus H(U, V, y^r) \oplus (m||r) = H(U, V, (g^r)^x) \oplus H(U, V, (g^x)^r) \oplus (m||r) = H(U, V, g^{rx}) \oplus H(U, V, g^{xr}) \oplus (m||r) = m||r$. Note that the computational efficiency of the equality test is relatively low to limit its applications where the performance and energy efficiency should be considered as a priority because of the analysis of BP operation given in Section 2.1.1.3.

In Section 2.1.2, I introduce several advanced methods for access control prevalently applied in designing modern cryptographic schemes. It is clear that each method has its own proprieties to serve various security purposes in access control. However, the feasibility of utilising these methods as the cryptographic components to construct security schemes in a specific scenario should be considered thoroughly, and by carefully considering different factors involved, such as computational complexity, time consumption, communication throughput, and energy efficiency.

2.1.3 eHealth

eHealth is a recent healthcare practice supported by ICT to cover broad electronic and digital processes in health including electronic medical records (EMRs), electronic prescriptions (ePrescribing), telemedicine (i.e., physical and psychological diagnosis and treatments at a distance), telesurgery and so on [55]. The global eHealth market size was valued at about USD 13 billion in 2020 and is estimated to grow to USD 39 billion

by 2025 [1]. Meanwhile, facilitated by advances in Internet of Things (IoT), there is a strong tendency that more and more IoT devices (e.g., wearable medical alert bracelets, healthcare sensors, wireless pacemakers and implanted insulin pumps) are involved in eHealth to protect people [56]. Generally, there are three components consisting of an eHealth application: data, devices and services [57] as shown in Figure 2.5.

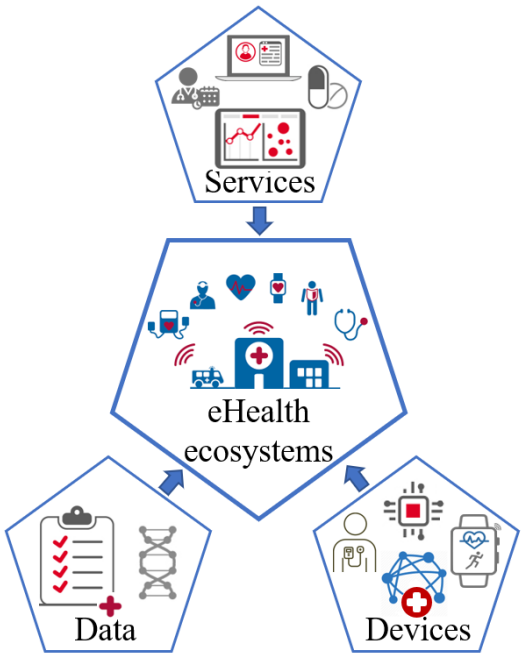

Figure 2.5: A high-level schematic general architecture of eHealth.

In this architecture, eHealth data mainly represents electronic medical records (EMRs[5]) for personal healthcare including health status data, examinations, diagnosis history and results, prescriptions and so on [59]. The eHealth data is usually saved and managed by the providers of eHealth services such as hospitals and clinics. However, driven by the use of Internet of Things (IoT), many wearable eHealth devices are allowed to be used outside healthcare centres, enabling healthcare users (e.g., patients) to upload their health data, and healthcare providers (e.g., hospitals and doctors) and users to monitor users' health status anywhere and anytime [3]. Therefore, there is an obvious trend to

[5]There are many other notations similar to EMR and EMRs but they have different definitions in different papers [58]. In this thesis, EMRs is defined to involve both private (identity) and medical information of patients.

realise the distributed management of electronic medical records (EMRs) because EMRs tend to be highly distributed in terms of who (doctor, nurse, administrator etc.) has modified what and when and where this is done [10].

Meanwhile, eHealth involves different kinds of sensors and portable devices to monitor heartbeat (for arrhythmia detection), blood pressure, breathing rate, electrocardiogram (EEG), peripheral oxygen saturation, blood glucose and many other physical health indexes [59]. Furthermore, there are other functional wearable and implanted devices that have been applied for different medical purposes such as allergy alert bracelets, wireless pacemakers and implanted insulin pumps. Since the medical data generated from such devices needs to be transported to medical institutions or emergency departments for concrete analysis and further actions by medical staff, distributed networks are deployed to achieve the connectivity between the devices and the participants in eHealth [60]. Supported by the eHealth data and devices, numerous different eHealth services can be realised to offer broad medical functions and networks for personal healthcare remotely including personal health status monitor, emergency medicine, periodic drug supplement, allergy and arrhythmia alarm, fast (early) diagnosis and so on. In other words, people can benefit from eHealth to know their health status anywhere and anytime, detect physical abnormalities and to contact with emergency (i.e., first aid) health service provider in a timely manner, and to obtain diagnoses and prescriptions without going to local medical institutions.

More recently, eHealth is being transformed from a centralised architecture to a more distributed shape because of the advances of IoT offering seamless platforms for people and objects to connect with each other, much easier [10]. Therefore, benefiting from IoT, eHealth devices can support people to view their real-time health indexes and ensure these devices to connect to professional medical institutions in a timely manner for rapid medical analysis and response to a dynamic manner [61]. Meanwhile, this transformation (i.e., more distributed eHealth) may require the support of distributed data management approaches such as a blockchain to allow every participant to manage

access to its own medical data with using distributed storage and consensus network.

2.1.4 Blockchain

Blockchain, stemming from the first decentralised digital cryptocurrency proposed by Satoshi Nakamoto [7], is a distributed and immutable ledger that enables transparent transactions. A general blockchain structure is depicted in Figure. 2.3 in Section 2.1.2.3. Blockchain technology is a combination of modern cryptography, peer-to-peer network communication, distributed system consistency, and smart contracts that implements data exchange, process, and storage [62]. A key benefit of blockchain is the decentralised consensus mechanism to realise anonymous, secure and accountable transactions. In general, a blockchain can be categorised into three types which are public blockchain, private blockchain and consortium blockchain. In a public blockchain, every peer is able to access the blockchain and participate in the consensus process according to the applied consensus mechanism. In contrast, the block generation and consensus process are fully controlled by a single organisation in a private blockchain. The consortium blockchain nominates a set of pre-selected nodes to control the consensus process. Real-world examples of a public blockchain include Ethereum and IOTA [63], whereas Hyperledger Fabric [64] enables competing businesses and groups to maintain the privacy and confidentiality of their transactions in a consortium or private blockchain [65]. Note that there is another taxonomy of blockchain that a blockchain can be permissioned or permissionless. A permissioned blockchain needs prior approval before joining the system or accessing the blockchain data whereas a permissionless blockchain allows anyone to participate in the system [66]. A brief comparison of public blockchain, consortium blockchain and private blockchain is shown in Table 2.3.

As a key component of blockchain, consensus mechanisms are responsible for maintaining the integrity of the data that safeguards the immutability and auditability under any circumstances. One traditional consensus mechanism "Proof of Work" (PoW), which is firstly introduced in Bitcoin, defines the difficulty of finding new valid blocks and

Table 2.3: The comparison of public blockchain, consortium blockchain and private blockchain.

Property	Public	Consortium	Private
Participating node	Permissionless	Permissioned	Permissioned
Consensus pattern	Decentralised	Partially centralised	centralised
Data transparency	Public	Private	Private
Trust model	Untrusted	Semi-trusted	Trusted

appends the found blocks to the chain [7]. It requires all participants to dedicate computation resources towards the "mining" work, where miners are required to solve a hash puzzle[6] to obtain a new valid block then add it to the global chain [67]. Nowadays, Bitcoin has become the mainstream base currency in the virtual cryptocurrency market[7]. However, it wastes a huge amount of computation (energy) and requires a constant global effort, which makes it impossible for some low-resource scenarios such as IoT-driven eHealth applications.

In order to mitigate the high computational consumption of mining, Proof of Stake (PoS) is adopted on the future Serenity platform (Ethereum 2.0) which partly replaces the mining operation with an alternative approach determining each participant's bonus by its stake of the virtual currency [68]. Apart from cryptocurrency, PoS has been applied in logistics, IoT, artificial intelligence (AI), and forecasting (e.g., with respect to voting and the stock market) and so on [66].

Practical Byzantine Fault Tolerance algorithm (PBFT) uses the concept of a replicated state machine for transactions which can resist the "34% attack" [69]. As the prevalent implementation of PBFT, Hyperledger Fabric is utilised to construct various applications in different domains including government, medical data management, smart grids and so on [70]. However, this algorithm requires the network to have the global knowledge of the participants, and there must be at least $3f + 1$ nodes in order to reach a consensus for the non-fault nodes if f nodes are fault. Therefore, it does

[6] An example is a miner should determine a message $m \in \{0,1\}^*$ to satisfy $H^n(m) = H(H(...H(m)...)) = 0$ to obtain one bitcoin (BTC), where H is a hash function to be iterated n times.

[7] Data source: https://coinmarketcap.com/

not scale well with the number of participants, i.e., numerous participants can obviously decrease the consensus performance to restrict the scalability of PBFT and other similar BFT-based (Byzantine Fault Tolerance based) consensus mechanisms [71–73].

There are other alternative consensus mechanisms that are proposed to replace the mining operation, such as XFT (Cross Fault Tolerance) with a low confirmation waiting time [74], Proof of Luck (PoL) with high transaction rate [75], and PoET (Proof of Elapsed Time) with a low computational consumption [76]. These consensus mechanisms are designed to provide different trade-offs between transaction rate, cost of participation, and efficiency of communication. While blockchain has independently emerged as a powerful technology, the consensus mechanism evolved independently as well dictated by the requirements from various distributed systems. Hence, choosing consensus mechanisms for different blockchain applications depends highly on an application scenario's requirements, where the trade-offs between latency, transactions rate, energy expenditure, security and so on should be considered. It is important to ensure that the selected consensus mechanism is capable of implementing all required functions correctly in not only the normal scenario but also certain adversarial conditions.

2.1.5 Blockchain-enabled EMRs

In conventional eHealth, EMRs are stored by different medical institutions (e.g., hospitals and healthcare centres) independently in their own databases. Obviously, every medical institution can benefit from this architecture to achieve efficient data security for its EMRs. However, the adverse side of a centralised architecture is to restrict EMR data sharing and interaction with other medical institutions because EMR data access from other medical institutions should be authorised by the EMR data owner that can degrade the time efficiency and flexibility when handling EMRs. Meanwhile, different medical institutions deploy different hardware and software to implement their data management systems of EMRs leading to incompatibilities for EMR data sharing amongst them [77].

Furthermore, driven by the use of Internet of Things (IoT), wearable eHealth devices

can be used outside healthcare centres, enabling not just health providers, but also health users, to monitor their own health status anywhere and anytime [78, 79]. Therefore, EMRs tend to be highly distributed in terms of who (local doctor, hospital doctor, administrator etc.) has modified what and when and where (e.g., in hospital, doctor surgery, care homes, private homes etc.) this is done. Management of EMRs including secure storage and access control, is a crucial requirement for eHealth [80], yet it is very challenging to achieve this because of the highly distributed and fragmented nature of EMRs and the wide range of users who are authorised to access EMRs [6].

The combination of blockchain and eHealth is a promising revolution to transform the current centralised management controlled by medical institutions in a more decentralise manner that allows every patient to manage its own EMRs [81]. According to the technical architectures of blockchain sculpted by Satoshi Nakamoto [7], blockchain technology and blockchain-enabled applications have attracted a high attention from different fields, including but not limited to, finance, governments and academies around the world in recent years, because of the features of blockchain: decentralisation, tampering immunity and transparency [82]. In eHealth ecosystems, the use of a blockchain model is currently being investigated as a highly distributed data structure for EMR transactions (to store, query and share) that enables them to be verified and recorded through a consensus of all the parties involved [6]. By utilising blockchain technology, every patient can possess and manage its own EMRs and share its EMRs with medical staff from different medical institutions. Compared with centralised EMR managements, blockchain-enabled EMR management allows EMR sharing and patients' supervision without the barriers between different medical institutions to shape the patient-centric (decentralised) EMR management [83].

Even though patients can benefit from this evolution to control their EMRs and preserve their privacy in a more flexible way, there are still many challenges that should be considered when implementing blockchains. In terms of security and privacy, one of the key challenges here when using blockchain in EMRs is that the inherent focus

of blockchain technology is not to limit unauthorised data access to avoid the leakage of specific confidential parts in EMRs [84]. For example, if an epidemiologist tries to determine the incidence of a disease based upon the blockchain-enabled EMRs, the data obtained by the epidemiologist should only reveal the disease name, the gender and age of each patient. However, in the blocks of the EMRs, other data such as social security numbers and home addresses may be presented, which are not related to the work of the epidemiologist, yet can be disclosed.

2.2 Literature review

2.2.1 eHealth security and privacy concerns

In this part, I analyse the threats and safeguards to security and privacy in eHealth in terms of the three components shown in Figure 2.5, i.e., data, devices and services.

2.2.1.1 Data

EHealth data such as EMRs involve not only medical diagnoses, prescriptions, medical images such as computed tomography(CT), magnetic resonance imaging (MRI), positron emission tomography (PET), ultrasound and pathology images, and other records, but also private information. Therefore, the major security and privacy concerns for eHealth data are access control (authentication and authorisation) and privacy preservation to restrict illegal data access since the private information of a patient should only be known by the related medical staff [85]. Furthermore, if the medical records are manipulated by the attacker, it may lead to false alarms, wasting medical resources and could even lead to patients being put into a critical condition (if the emergency situations are hidden in the EMRs).

To control the access to EMRs in eHealth, there are numerous authentication and authorisation schemes implemented based upon various cryptographic methods to verify the data requesters' identities and to allow the specific requesters to access different parts in EMRs [86]. For example, Van der Haak et al. [87] utilise symmetric encryption to

ensure the confidentiality of the shared EMRs and use digital signatures to protect EMRs integrity in sharing them across different medical institutions. As for authorisation, Lin et al. [59] and Tang et al. [88] propose two schemes of authorisation and encryption to transmit and share health monitoring data using cloud computing. The authorisation and encryption are implemented by BP and IBE to exploit cloud computing to help users reduce a heavy computation load without leaking their private information to the clouds. Meanwhile, the authors [89–91] take the advantage of ABE to realise fine-grained authorisation for EMRs access to allow different participants (distinguished by their attributes) to view different parts of the EMRs.

In privacy preservation, one main purpose and challenge is to separate the sensitive private information and the insensitive medical records in EMRs and keep users' anonymity when their EMRs are used for medical analysis and diagnosis [92]. A simple method proposed by Ateniese et al. [93] is to use pseudonyms to preserve patient anonymity i.e., the designed scheme can transform a prescription for the pseudonym used with his doctor to a prescription for the pseudonym used with the pharmacy. Furthermore, Dubovitskaya et al. [94] use the k-anonymity algorithm [95] to hide the patients' identity information in an EMR data set when the EMRs are exploited for pathological analysis by medical analysts. Besides, there is another privacy concern pertaining to the signature used to ensure EMR integrity since using signatures may expose the real identity of the EMR owner. Liu et al. [96] propose to utilise attribute-based signature (ABS) and a trapdoor to sign EMRs anonymously. Meanwhile, the signature verification is delegated to a TTP in the proposed system so that only the TTP and the EMR owner can possess the corresponding real identity of this owner.

2.2.1.2 Devices

EHealth devices consist of not only portable healthcare devices, but also large-scale (distributed) networks constructed by numerous network fundamental devices such as IoT-driven eHealth [3]. Thence, the security and privacy concerns to be discussed are

two-fold.

On the one side, how to secure the eHealth devices themselves should be considered because some devices are critical to sustain individual life such as cardiac pacemakers. If such devices work abnormally or are manipulated by malicious attackers, the users of these eHealth devices may have a huge risk to be operated in a life-threatening manner. For example, Li et al. [97] demonstrate a successful man-in-the-middle (MITM) attack to control an insulin pump through wireless communications. To mitigate this attack, the authors propose to use rolling code (pseudorandom number) as the access token to prevent replay attacks from eavesdroppers. Furthermore, to avoid hijacks and injections of malicious codes, Lu et al. [4] make use of a trusted platform module (TPM) and a process of authentication to secure the system booting and the kernel of the eHealth device.

On the other side, since EMRs and other private or medical information need to be transported by communications through eHealth networks, the communication security should also be discussed to prevent data leakage and tampering from eavesdropping and replay to ensure the confidentiality and integrity of the transmitted eHealth data. There are many cryptographic countermeasures proposed to address the issues of eavesdropping and replay based upon encryption and signature [3, 98, 99]. In the study of Lin et al. [100], IBE and IBS (identity-based signature) are constructed using the BP and DES (Data Encryption Standard) then utilised to encrypt and sign the transmitted EMRs to prevent eavesdropping. Meanwhile, the timestamp method is applied in their scheme to avoid replay attacks. Similarly, in [4], IBE is applied for data encryption to avoid eavesdropping in communications.

2.2.1.3 Services

As for eHealth services, the major security concern is the reliability to ensure that a user can access eHealth services stably especially when he or she is facing an emergency situation [101]. Denial-of-service (DoS) and distributed denial-of-service (DDoS) attacks

are a severe threat to be addressed to keep eHealth services running stably since this kind of attack aims to block communications and waste devices' computational resource to make eHealth services unavailable [102]. There are several common countermeasures to mitigate Dos and DDos attacks by analysing certain features in network packets or the patterns of malicious connections with policy filtering [103, 104], machine learning [105–107], game theory and many other methods in the network layer [108, 109]. In the application layer, authentication and signature are two prevalent approaches to form access policies to block the connections whose identity cannot be verified [110–113].

2.2.2 Blockchain threats and safeguards

Since blockchain and different consensus mechanisms were proposed, the security discussions about the blockchain context and consensus mechanisms have never stopped. Meanwhile, as the carrier of the trade strategies and applied consensus mechanisms, smart contracts may lead to transaction chaos or even real-world economic loss if their security vulnerabilities are exploited by malicious attackers. Furthermore, more and more smart contracts are designed to describe different consensus mechanisms, thus, the security of the executing entities (e.g., Ethereum virtual machine, EVM in Ethereum) is of concern to ensure that smart contracts can be executed correctly. Therefore, the threats and safeguards of blockchain are illustrated in terms of three aspects: consensus mechanism, smart contract and executing entity in this part. Note that I choose EVM as a practical case of executing entity to analyse because EVM (from Ethereum) is a major operational environment for smart contracts utilised by many blockchain applications.

2.2.2.1 Consensus mechanism

Collusion attack. The collision attack is the most common method, which can be used by attackers to attack different consensus mechanisms. To be specific, if the attacker has more than 50% of the computing power in a PoW-based context, the attacker can manipulate all the results of the consensus requests, thereby causing fatal problems to the PoW network (e.g., selfish mining, cancelled transactions and double spending) [114].

On the other hand, if the attacker can control over 50% of the validation nodes selected by the consensus leader, the collusion attack may occur in the PBFT-related context. Compared with the PoW-based context, the PBFT-based context can be more easily affected by a collusion attack, because the quantity of the used validation nodes defined by different consensus mechanisms that process each consensus request is much less than 50% of all the nodes in the PBFT-based context.

Sybil attack. The Sybil attack means that one attacker claims a large number of fake identities (nodes), and then attempts to influence the voting result of the consensus mechanism in the consensus network. If the identity authentication is not robust enough, the Sybil attack can become widespread in peer-to-peer networks. For a PoW consensus mechanism operating in an anonymous network, the method to ensure each node is valid is to check whether the node owns a considerable amount of computing power. Meanwhile, PoW consensus mechanism provides miners with an incentive to work honestly, but not to work incorrectly to avoid the Sybil attack. However, if enough fake nodes are selected in the validator group, the PBFT-related consensus mechanisms may be affected by the Sybil attack, resulting in a higher probability of an incorrect consensus. Note that the validating leader can request to check the voting result then punish the fake nodes in the PBFT-based context, but the latency is much higher than the PoW consensus mechanism [115].

Eclipse attack. In an eclipse attack, the attacker monopolises all incoming and outgoing connections of the victim, thus isolating the victim from the rest of its peers in the network [116]. To be specific, a node depends on n number of nodes selected by the peer selection strategy to view its distributed ledger in a decentralised network. However, if an attacker can force this victim node to choose all the n number of nodes from the malicious nodes manipulated by him, the attacker can eclipse the original ledger of the victim node and replace the original ledger with a tampered ledger. Figure 2.6 shows an example of the eclipse attack, where the victim node cannot send/receive correct ledgers in the decentralised network, since the victim can only choose the malicious nodes as

the peers. Compared with the Sybil attack that affects the entire blockchain network, the eclipse attack only attacks certain nodes more precisely, so the attack cost of the eclipse attack is much lower. It indicates that the eclipse attack is much easier to be performed in real-world blockchain-enable systems [117, 118]. In order to detect the

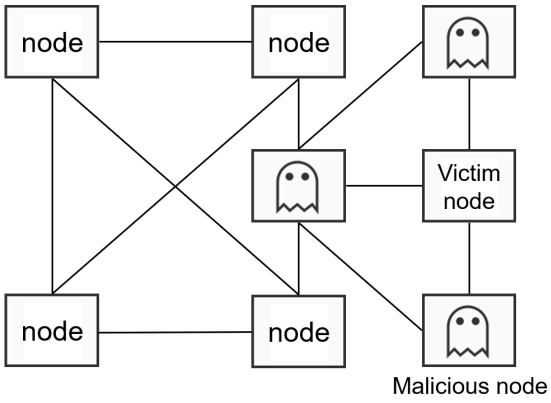

Figure 2.6: An example of an eclipse attack.

eclipse attack, Xu et al. [119] proposed to utilise random forest classification algorithm to separate the attack data packets in terms of certain packet features (e.g., packet size, access frequency, access time and so on). Furthermore, to mitigate the eclipse attack, some countermeasures are proposed by adding extra network links between different nodes. The countermeasure proposed by Walck et al. [120] is partial randomness, giving priority to the old nodes with fresh outgoing connections in the peer selection. Another countermeasure is to establish some known and verified nodes outgoing connections to test the neighbour nodes before they are selected as the peers [116].

2.2.2.2 Ethereum virtual machine (EVM)

EVM is a transaction-based state machine that runs on a 256-bit stack to execute all the functions in the smart contract, and then implements the consensus mechanism [121]. Compared with the VMs used for general computation such as Java Virtual Machine (JVM), Dalvik and ART in Android, the complexity of an EVM is relatively low because it only needs to execute smart contracts deterministically and supports certain cryptographic primitives [122]. Nevertheless, security is still a primary concern

in EVM, since it is the last barrier to prevent malicious smart contracts and flawed consensus mechanism. On the other hand, Ethereum is the most mainstream platform (or framework) serving for numerous cryptocurrency and non-cryptocurrency applications. If EVM itself has severe vulnerabilities, attackers may endanger all Ethereum-based platforms to cause irreversible financial loss. Meanwhile, there are four versions of EVM in use based upon different program languages: py-evm (Python), js-evm (JavaScript), geth (Golang) and aleth (C++). This diversity increases the potential attack range for attackers and the workload of security analysis for security researchers. In current research, the major methods for discovering EVM vulnerabilities are symbolic execution and fuzzing [123], and the explored vulnerabilities in EVM are mainly related to the memory management (e.g., stack overflow and illegal memory access) and opcode [121, 124]. Four vulnerable cases discovered in EVM are summarised as follows.

- CVE-2016-2085 [125]. This vulnerability exists because geth (EVM of Go Ethereum) invokes an insecure function *memcmp()*, which can lead to a timing side-channel attack [126]. Specifically, the forgery complexity of message authentication code (MAC) drops from 2^{128} to 2^{12}.

- CVE-2018-18920 [127]. This vulnerability can cause smart contracts to be executed indefinitely without paying for gas, since Py-EVM (EVM of Python Ethereum) does not check the invalid opcode, resulting in stack corruption.

- CVE-2018-19184 [128]. This vulnerability allows attackers to cause a denial of service (segmentation violation) via crafted bytecode (nil pointer dereference) in *geth*.

- CVE-2018-20421 [129]. This vulnerability allows attackers to waste a large amount of memory to lead to a denial of service by rewriting the length of a dynamic array, and then writing data to a location with a large index in the memory.

Apart from detecting vulnerabilities in EVM, some researchers try to reinforce EVM via bytecodes verification and semantics analysis [130–133]. The target of these two

methods is to eliminate unsafe bytecodes generated upon different smart contracts. Since EVM is continuously now maintained and updated, new vulnerabilities in EVM may threaten all consensus mechanisms and smart contracts implemented on Ethereum. Therefore, the blockchain community should pay close attention to EVM security.

2.2.2.3 Smart contracts

A smart contract is an entity to carry the implemented consensus mechanism and transaction strategy. Therefore, the security of consensus mechanism is tightly linked to the security of smart contracts. In order to explore the potential vulnerabilities in smart contracts, several fuzzing tools have been implemented based upon heuristic search, symbolic execution, control flow graph and data stream analysis [134–138]. The common vulnerabilities found real-world smart contracts are summarised as follows.

- **Leaking and suicidal.** A smart contract is considered to have a leak vulnerability if it leaks Ethereum (ETH) to attackers. Similarly, a smart contract is considered suicidal if it can be killed by attackers [139]. Both of these two vulnerabilities are caused by inappropriate permission settings (especially in some smart contracts' tests), which allow attackers to invoke *send()* or *selfdestruct()* functions without any restrictions.

- **Block status dependency.** If the transaction of sending ETH (or other critical operations) relies on certain block status variables (e.g., timestamp, difficulty, gas limit and so on), then these smart contracts can be vulnerable since an attacker can construct transactions to achieve malicious behaviours by analysing the block status. The timestamp dependency is an example. Every block has a timestamp in the blockchain to record the time of the transaction. When the trigger conditions of some critical operations in a smart contract depend on timestamps, the timestamp can be exploited as a vulnerability. If an attacker can manipulate the timestamps (e.g., change the local system time), the timestamp-dependent smart contracts may be vulnerable.

- **Exception disorder.** The reason for the exception disorder is the inconsistency during the exception handling. When a smart contract A tries to invoke a function f in another smart contract B, the function call may fail and then generate different exceptions. Normally, all the transactions will be reverted in terms of the chain of nested calls for $f \in B$. However, if there is at least one low-level function call (e.g., *address.call()* and *address.send()*) in the chain, the transaction rollback will be terminated at the last low-level function call. Therefore, the rest of the transactions cannot be reverted, and the exceptions cannot be propagated to the caller A.

- **Reentrancy.** In general, the status of the contract's account can be changed after the invocation of some reentrant functions in a smart contract is completed. However, many functions in smart contracts are not designed to be reentrant. Therefore, if a malicious smart contract invokes these functions in a reentrant manner repeatedly, it may lead to ETH theft. The famous "The DAO (Decentralised Autonomous Organization)" attack took advantage of the reentrancy vulnerability through the fallback function *withdraw()* to steal about USD 60 million [140].

- **Gasless send.** When the sender tries to send ETH > 0 to the recipient, the fallback function in the recipient smart contract will be invoked with a fixed gas stipend determined by the EVM. However, if the gas consumption of the fallback function is designed to be higher than the current gas balance of the sender in the recipient contract, the sender will receive the exception "out of gas". Therefore, if the exception "out of gas" is not handled and broadcast appropriately, a malicious sender can send ETH to a recipient without costing transaction fee (i.e., gas in EVM).

- **Frozen ETH (locking).** Some smart contracts are designed to invoke certain functions from other smart contracts to operate ETH via *delegatecall()*. It means that these smart contracts entirely depend on the related functions of other smart contracts to manipulate ETH, as there is no actual ETH manipulating function in these smart contracts. When the smart contracts that provide the ETH manip-

ulating functions execute a self-destroy (suicide) operation, the smart contracts with only delegated calls cannot send ETH to others forever, so that all the ETH is frozen. In November 2017, the frozen ETH bug resulted in the Parity Wallet users permanently losing an estimated USD 150 million in funds [141].

- **Dangerous delegatecall.** The *delegatecall* opcode is designed for a caller smart contract to invoke other library contracts. Specifically, the caller contract can load the library contract's code and execute it in the context of the caller contract. Since the parameter of *delegatecall* is the address of a library contract, an attacker can execute arbitrary code in the caller contract by manipulating the parameter (i.e., the library contract's address) of *delegatecall*. This vulnerability has been exploited and resulted in a USD 30 million loss in a multi-signature wallet [142].

In a nutshell, when discussing the security of blockchain, People should not only focus on the threats to consensus mechanisms, but also the threats to both the carrier (smart contracts) and the running environments such as EVM. In current non-cryptocurrency sectors such as blockchain-enabled eHealth, there is no standard (or any best practice) to guide people to deploy secure blockchain-based systems. Even in the cryptocurrency area, the related standards and standard operation procedures (SOPs) are still in discussing and developing. Otherwise, there would not be numerous cases where a huge number of cryptocurrencies are stolen on different cryptocurrency trading platforms [143]. In my opinion, building up security standards (such as PKI [144]) and normalising the code writing of smart contracts for blockchain-enabled applications should be considered as the top priority in non-cryptocurrency areas.

2.2.3 Access control enhancement for blockchain-enabled EMRs

As EMRs hold personal data about patients that can be confidential in different ways to different stakeholders, approaches to construct flexible and granular access control for blockchain-enabled EMRs are needed. However, since a block is usually the fundamental access unit in blockchain, it cannot support a fine level of granularity in access control

to authorise queries when a medical person only requires partial information in a block for its work.

To address the challenge of access control for blockchain-enabled EMRs, numerous state-of-the-art studies are proposed based upon different cryptographic methods. The first prototype considering access control in blockchain-enabled EMRs is MedRec [145], where a system based on Ethereum smart contracts and two incentivising mining models are presented for an intelligent representation of EMRs that are stored in network nodes distributedly. The authors state their design can support fine-grained access control but without showing any concrete algorithms.

Another early thought of using blockchain in EMR management is an architecture Healthcare Data Gateways (HDG) proposed by Yue et al. [146]. In the HDG architecture, stored EMRs can be transmitted via a consensus gateway with the access policies to authorise data access. Even though the designed access policies are fine-grained, the access control is not flexible i.e., a user can only access the data of one attribute in one query but querying the data of several specific blocks is not supported. Furthermore, as the authors regard the consensus gateway as a trusted network, all users' private data (including access permissions) are exposed to the consensus network in their scheme. If some nodes in the consensus network are compromised, the transmitted personal data may be revealed to the attacker.

Dubovitskaya et al. [147] discuss a PKC-based permission control architecture for blockchain to implement secure EMRs. In this architecture, certificates are proposed to realise authentication and signing, while AES and DH are applied for encryption, but no detailed access control scheme or algorithm is proposed. Meanwhile, the authors claim that the fine-grained access control policies can regulate the access and update of EMRs but the authorisation method and access granularity are not presented. Besides, the suggested blockchain implementation Hyperledger Fabric with PBFT as the consensus mechanism still needs to be deliberated further when used to realise a blockchain-enabled EMR management system based upon the scalability issue of PBFT analysed in Section

2.1.4. [148] and [149] utilise blockchain to realise the EMR storage layer for EMR management and sharing. The authors state that the designed systems require access control but there is no exhaustive design of access control presented in their papers. Similarly, a superficial discussion about access control appears in many proposed architectures (or frameworks) of blockchain-enabled EMRs [81, 83, 150–153]. For example, Chen et al. [151] use a private blockchain to store EMRs for a medical institution and a consortium blockchain to share EMRs between different medical institutions but how to implement the mentioned requirements of authentication and authorisation is not illustrated. Wu et al. [81] and Omar et al. [152] propose a privacy-friendly platform of EMR management with blockchain but there is only a simple identity authentication design without considering the authorisation of EMR access.

More recently, there are more and more concrete solutions proposed to address the access control issue for blockchain-enabled EMRs. IBM (International Business Machines Corporation) proposes a healthcare data exchange platform built on blockchain that stores EMRs in databases but the granularity control is only document-based [154]. Therefore, a patient cannot hide certain parts that it does not wish others such as system administrators to know in its medical documents. Liu et al. [155] propose an access control scheme for sharing blockchain-enabled EMRs with CP-ABE and PoS called blockchain-based privacy-preserving data sharing (BPDS). The authors illustrate detailed algorithms of authentication and authorisation with fine-grained access policies. However, BPDS has a similar issue of flexibility to HDG. When data sharing with BPDS, if a user tries to acquire the data of three attributes in one patient's EMRs, it needs to query three times i.e., each query can only contain the data of one attribute. In other words, a user cannot query the data of many attributes in one patient's EMRs or the data of the same one attribute in many patients' EMRs in one query. This design may lead to higher time consumption and network throughput for bulk queries. Besides, since the applied CP-ABE algorithms are constructed upon a large number of BP operations, BPDS can sharply increase the computational cost of users' devices especially for the

resource-constrained devices (see Section 2.1.1.3 and 2.1.1.4). Tang et al. [156] also use BP to implement a multi-party signature scheme to ensure the integrity of the queried EMRs. However, this design is only used as a method of message authentication; it cannot support authorisation to restrict different parts in EMRs that different users can access.

Xu et al. [84] propose HealthChain to manage EMRs with two chains, user chain and doctor chain. IBE and IBS are utilised to ensure the confidentiality and integrity of the EMRs in HealthChain. The control of access granularity is realised by the definitions of the data that can be involved in the user chain and doctor chain. However, compared with BPDS, this access granularity is not fine-grained or flexible because all attributes in the user chain and doctor chain are fixed with respect to access policies. Furthermore, the authors only demonstrate an algorithm for key management for IBE and IBS but no concrete algorithm for access control.

In [157], the authors illustrate a data encryption scheme with BP to secure EMRs in communications. Meanwhile, an access control protocol is proposed with an access control list (ACL) to regulate what data can be read or written in blockchain-enabled EMRs by a specific user. This method can realise fine-grained and flexible authorisation for EMR queries because the access control list can contain many policies (read and write) for different users to access different parts of the EMRs. However, the authors do not present any concrete cryptographic design to perform this protocol. Furthermore, it is still unclear in this study how the policy design can fit the blockchain structure. Guo et al. [158] also use an ACL to control the actions (read, write and update) that can be performed by different users. Meanwhile, the ABAC (Attribute-based Access Control) approach is proposed to set the attributes different users can have to realise fine-grained and flexible access control. Unfortunately, as with [157], there is no concrete ABAC scheme presented based upon cryptography in [158]. Besides, Khatoon [159] proposes to put ACL into every block of the blockchain-enabled EMRs but this design may decrease the scalability of the proposed blockchain-based EMR management system. This is

because the ACL can become enormous and redundant if there are many users, roles and data attributes to generate numerous access policies in an ACL.

[77] proposes an implementation of blockchain-enabled EMRs with Hyperledger Fabric and IPFS (InterPlanetary File System) to manage EMRs in a permissioned blockchain. The authors apply role-based access control (RBAC) to allow different roles (patient and doctor) to execute different database operations (read, create, and update). The similar design is also adopted by Nguyen et al. [8], where doctors and patients can be bound to form access policies. However, this method cannot restrict different parts (attributes) of EMRs that users with different roles can access.

2.3 Comparison and analysis

After discussing recent advances of security and privacy in blockchain-enabled EMRs (Section 2.2.3), I summarise others' implemented features of security and privacy in Table 2.4. Furthermore, a deeper investigation of the cryptographic methods applied and the granular properties achieved in different access control solutions to blockchain-enabled EMRs is summarised in Table 2.5. Note that in the two tables, only the solutions having concrete implementations of access control (authentication or authorisation) are considered.

Table 2.4: The comparison of the implemented security and privacy features in different solutions of blockchain-enabled EMRs.

Solution	Confidentiality	Integrity	Access control		Year
			Authentication	Authorisation	
HDG [146]	√	√	√	√	2016
[147]	√	√	√	×	2017
[81]	√	√	√	×	2018
BPDS [155]	√	√	√	√	2018
MediBchain [152]	√	√	√	×	2019
[154]	√	√	√	√	2019
[157]	√	√	√	√	2019
HIDEchain [153]	×	√	√	×	2020

According to Table 2.4, most current solutions of blockchain-enabled EMRs can utilise

encryption, signature and other cryptographic methods to ensure the confidentiality and integrity of the transported EMRs. Meanwhile, all the compared solutions can realise authentication but only half of them consider authorisation. Therefore, access control (especially for authorisation) is a major concern to be addressed in this research to preserve patients' private information in blockchain-enabled EMRs.

Table 2.5: The comparison of the applied methods and properties in the access control of different blockchain-enabled solutions (given in Table 2.4).

Solution	Method		Granularity control		IoT
	Authentication	Authorisation	Fine-grain	Flexibility	
HDG [146]	hash certificate	BP ABE	√	×	×
[147]	certificate	×	×	×	×
[81]	BP	×	×	×	×
BPDS [155]	CP-ABE	CP-ABE	√	×	×
MediBchain [152]	ID/Pass	×	×	×	√
[154]	ID/Pass	ID/Pass	×	×	√
[157]	ABE	ACL	√	√	×
HIDEchain [153]	hash	×	×	×	√

To understand the merits and drawbacks of current solutions of access control for blockchain-enabled EMRs in a comprehensive manner, the applied cryptographic methods, granularity control including fine-grained, flexibility, and performance for resource-constrained IoT devices in eHealth (denoted by IoT) are summarised in Table 2.5 for the same blockchain-enabled solutions within Table 2.4. Note that if the applied cryptographic components are lightweight for resource-constrained IoT devices in an access control solution, it means this solution can support IoT devices with limited or low ICT resources. Apparently, compared to the relatively mature research on authentication, current discussion about authorisation mechanisms for blockchain-enabled EMRs is still in its infancy because of two main reasons.

Firstly, access authorisation is neglected by many studies as some stakeholders may not have realised the importance of patients' privacy in decentralised eHealth other than by using a traditional centralised eHealth design. In such a centralised eHealth design

such as at a hospital, people believe that the hospital can preserve the patients' privacy safely. However, there is no trusted central entity for EMR escrow (patients' private information) when a blockchain is applied to allow patients to manage their own EMRs in a more decentralised manner.

Secondly, granularity control has not been widely considered i.e., how to realise fine-grained and flexible granularity control for data queries to restrict access to private data in patients' EMRs and to be compatible with the blockchain structure. According to Section 2.2.3, most current studies do not consider authorisation or only present rudimentary thoughts to capture only one property (fine-grain or flexibility) of granularity control for blockchain-enabled EMRs. Although [157] shows a solution that can balance fine-grain and flexibility in the granularity control use in authorisation, there is still no concrete design in cryptography to practice its proposed ACL method or support the blockchain structure.

Furthermore, realising a complicated access control while taking care of limited or low performance resource-constrained IoT devices applied in eHealth should be studied further. This is because lightweight methods such as hash and identity with password (ID/Pass) can save resources for such IoT devices but they cannot achieve granularity control as shown in Table 2.5. In contrast, resource-consuming methods (such as BP) can be exploited for granularity control but they may consume much more computing resources, decreasing the service life of resource-constrained IoT devices.

2.4 Summary

In this chapter, I first introduce some fundamental concepts of public-key cryptography to help readers to ease the understanding of the proposed ideas and implementations in the following chapters. Then, a comprehensive investigation of the blockchain-eHealth combination is illustrated including the general background and the state-of-the-art security concerns for eHealth, blockchain and blockchain-enabled eHealth. Specifically, I

focus on access control for blockchain-enabled EMRs to present a comparison and analysis of the related state of the art. Furthermore, according to the limitations of the state-of-the-art access control solutions shown in Table2.5, the challenges of access control to be addressed for blockchain-enabled EMRs are concluded as follows:

1. Authorisation with fine-grained and flexible granularity control has not been thoroughly considered to protect patients' privacy in accessing blockchain-enabled EMRs;

2. Many designs of concrete schemes for access control (including authentication and authorisation) are not compatible with blockchain data structures;

3. The cryptographic components involved in many current access control solutions are not lightweight enough to fit resource-constrained IoT devices applied in eHealth with a limited computation performance.

Chapter 3

Block-based Access Control for Blockchain-enabled EMRs

This chapter first gives an overview of the whole system. Then it presents the first part of the system, an access control solution for exchanging blockchain-enabled Electronic Medical Records (EMRs) called BBACS (Block-based Access Control Scheme) that includes an access model and an access scheme. Unlike existing blockchain-oriented access control (authorisation) schemes for EMRs, the proposed access model can omit the agent layer (gateway) in order to authorise users' access with block level granularity, whilst maintaining compatibility with the underlying blockchain data structure.

3.1 System overview

Before demonstrating the proposed security and privacy enhancement schemes for blockchain-enabled EMRs in turn, I first describe an overview of how each scheme links together to comprise an access control system in Figure 3.1. Based upon the limitations analysed in Chapter 2, enhancing the authorisation for blockchain-enabled EMRs is the main objective. To realise authorisation, authentication and key distribution should be considered to validate users identities and to distribute the keys used in authorisation. Therefore,

when a user (or health device) needs to access the blockchain-enabled EMRs, its identity should first be authenticated. Then, the corresponding keys should be distributed to this user by the key generation centre (KGC) (see Section 4.2.1). The authentication and key distribution are discussed in Chapter 4.

After that, how to enhance authorisation for data access to blockchain-enabled EMRs is discussed. When the user requests access to the blockchain-EMRs, it needs to provide the corresponding access tokens to the requested EMRs. A primary authorisation scheme (BBACS) is presented in Chapter 3 to allow the blockchain-enabled EMR server to authorise EMR queries without the agent between the server and the users. To further improve the authorisation, Shamir secret sharing (SSS) is applied to propose a fine-grained authorisation scheme based upon BBACS to support granular authorisation and flexible queries in Chapter 4. During the authorisation, the user's keys (credentials) are exposed the agent (e.g., the pre-selected consensus nodes in the blockchain network), which may incur a risk of privacy leakage if the agent is compromised. Therefore, the idea of zero-knowledge proof (ZKP) is adopted to hide users' credentials for privacy preservation and anonymity during the authorisation in Chapter 5.

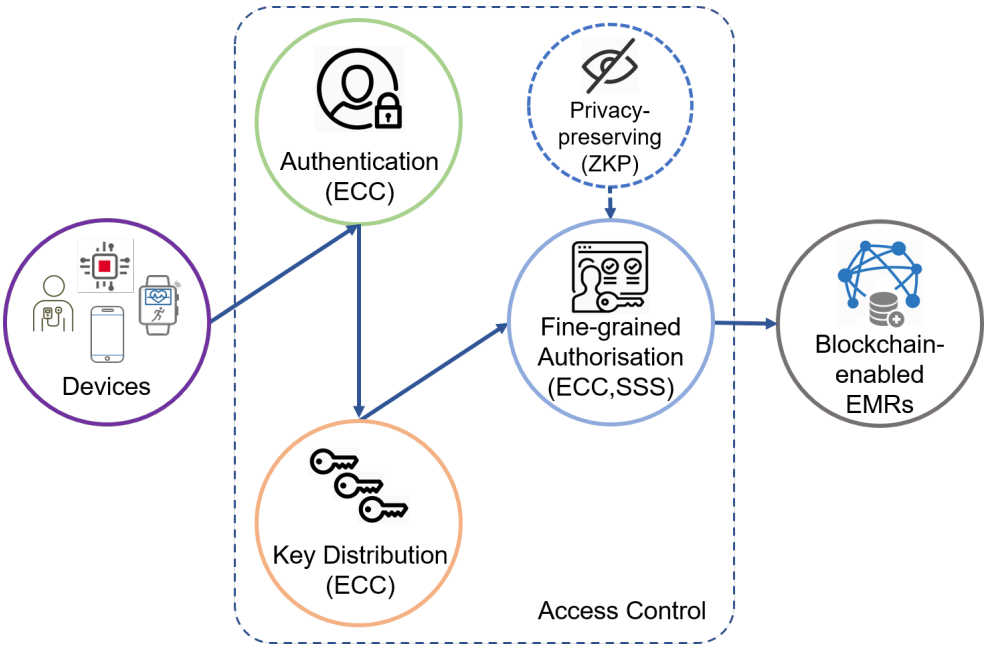

Figure 3.1: The system overview of the proposed schemes.

3.2 Motivation for adding access control to blockchain use with EMRs

Recently, the blockchain model (structure) is being investigated as a potential solution for the management of distributed and fragmented EMRs since blockchain is a highly distributed data structure suitable for EMR storage and queries. Furthermore, a feature of a blockchain is that it inherently enables all the operations (add, query and modify) in EMRs to be verified and recorded through a consensus of all the involved parties (or some permissioned parties) [6]. However, one of the key challenges here for using blockchain in EMRs is that the initial focus of the blockchain design is not to limit unauthorised data access to avoid exposing specific confidential parts when EMRs are constructed using blockchain [7]. It means the blockchain design can protect the data integrity of EMRs but it has no data access control.

As EMRs hold personal information about patients that can be confidential and private to the stakeholders, new approaches to constructing an access control solution to EMRs are needed. However, the current research, which applies blockchain to EMRs, usually also supports authentication to validate users' identities but without any authorisation design to determine what the users can access. For example, a doctor can only look through the EMRs of the patients he (or she) is responsible for. For a nurse, he (or she) should not know the diagnosis, the drug injection doses or other private data (e.g., social security numbers and home addresses) of the patients he (or she) is not taking care of. On the other hand, the agent (or gateway) design used in much current research may not be suitable for mid-scale (or small-scale) eHealth service providers (companies) as they may not be able to afford to set up such an agent. These challenges motivate me to present a new EMR access control solution to achieve precise access control (block-level granularity) for EMRs queries, without the need for agent support.

3.3 System model

In BBACS, there are two entities in the presented model (Figure. 3.2), which are users and the EMR server. The two entities are located in the same trusted cloud (distributed) network. Then, I illustrate the functions of each entity as in Figure. 3.2.

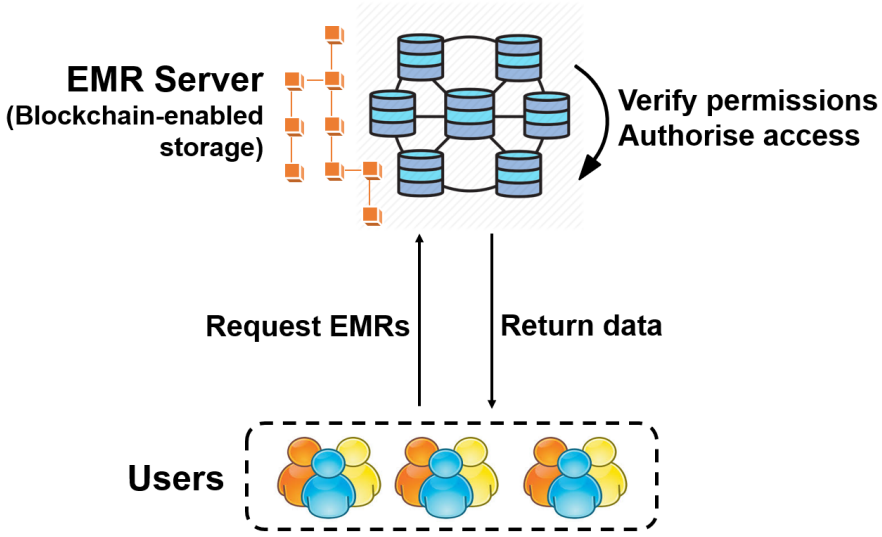

Figure 3.2: The proposed access model of the BBACS scheme.

Users. The users represent data inquirers e.g., doctors, patients and data analysts. The data inquirers initiate queries to obtain data from the EMR server. In general, the EMRs of each patient are constructed using several blocks. A unique token is allocated to each block when the block is added to the patient's EMRs. As the focus of the BBACS scheme is access authorisation during data exchange, the token distribution is not yet discussed in this chapter. Hence, it is assumed that the token for each block on the blockchain has been distributed to the corresponding valid data requesters (e.g., the patient, doctors and data analysts) by the medical management authority. Note that whilst a patient usually queries the personal blocks; the doctors and data analysts normally may query several blocks from different patients' EMRs. All the access permissions of the queries are verified through the EMR server.

EMR server. The EMR server is blockchain-enabled to store EMRs and provide

an inquiry service. The consensus nodes of the EMR server can authorise data access. Note that the proposed BBACS scheme focuses on data access authorisation but not on the blockchain properties such as the consensus mechanism. To be specific, there are two functions designed for the EMR server in the access model. First, the EMR server is used to store all the electronic medical records (EMRs) for eHealth in a blockchain [160]. Another function of a EMR server is to verify the access permissions and to authorise the block(s) access. If the data inquirers possess all the required permissions, the EMR server can authorise the access then return the queried data (blocks) to the users.

3.4 Algorithm description of block-based query authorisation

In this section, a block-based query authorisation scheme called BBACS (Block-based Access Control Scheme) is proposed. The correctness of the BBACS scheme is illustrated in Appendix A.

• **Hardness assumption**

Elliptic-curve Discrete Logarithm Problem (ECDLP). Let $E_p(a, b)$ be a cryptographic secure elliptic curve with the prime field $\mathbb{F}_p$ and a base point G. For any point $P \in E_p(a, b)$ and a random $u \in_R \mathbb{Z}_p^*$, any probabilistic polynomial-time algorithm $\mathcal{A}$ computes u with its advantage:

$$Adv^{ECDLP}_{\mathcal{A},E_p(a,b)} = Pr[c = u | u \in_R \mathbb{Z}_p^*, c = \mathcal{A}(P, uP)].$$

The ECDLP assumption can hold if for any probabilistic polynomial-time algorithm $\mathcal{A}$, its advantage $Adv^{ECDLP}_{\mathcal{A},E_p(a,b)}$ is negligible.

• **Setup(λ)**

This procedure outputs public parameters pp with the security parameter λ using the following steps.

1. Select a secure elliptic curve $E_p(a, b)$ and a base point G on $E_p(a, b)$, where $p \in \{0, 1\}^\lambda$ is a big prime, and a, b are the parameters of the elliptic curve.

2. Generate a random integer token $TK \in \{0, 1\}^\lambda$ for each block BLK (EMR) on the blockchain and write the token TK in the block BLK. Note that this step should be executed when a new block is created and added to the blockchain to keep the integrity of the blockchain. Furthermore, all the tokens have been distributed to the corresponding users correctly before the users request medical data in the blocks from the blockchain-enabled EMR server.

3. Select a symmetric encryption algorithm, e.g., AES (Advanced Encryption Standard [161]).

4. Select one secure cryptographic hash function, $H : \{0, 1\}^* \rightarrow \{0, 1\}^\lambda$.

5. Output the public parameters pp to finish the *Setup* phase, $pp = (E_p(a, b), G, AES, H)$. Note that it is assumed that all users can establish secure connections with the authenticated EMR server to obtain a correct pp before they query EMRs.

- **Query**(pp)

In this phase, the users prepare the data query Q via the following steps.

1. Prepare the sequence S_{id} including all the indexes of the queried blocks. For example, if a user want to request several (n) blocks on the blockchain, sequence S_{id} should contain all the indexes of the requested blocks n: $S_{id} = \{BLK_{id_i} | i = 1...n\}$.

2. Prepare the sequence S_{TK} including all the tokens of requested blocks. For the given sequence S_{id} in step 1, $S_{TK} = \{TK_{id_i} | i = 1...n\}$.

3. Use the sequence S_{TK} to calculate a value $k = TK_{id_1} \oplus TK_{id_2} \oplus ... \oplus TK_{id_{n-1}} \oplus TK_{id_n}$.

4. Calculate a point multiplication on elliptic curve $E_p(a, b)$ with k and the base point G to obtain a new point $P = kG$.

5. Send the query $Q = (S_{id}, P)$ to the EMR server.

- **Authorise**(pp, Q)

The EMR server validates the access permissions provided in Q from the user via the following steps.

1. Repeat step 2 in the *Query* phase with the indexes $S_{id} \in Q$ to obtain the token sequence $S'_{TK} = \{TK'_{id_i} | i = 1...n\}$ from the blockchain.
2. Use the sequence S'_{TK} to calculate the $k' = TK'_{id_1} \oplus TK'_{id_2} \oplus ... \oplus TK'_{id_{n-1}} \oplus TK'_{id_n}$.
3. Repeat step 4 in the *Query* phase to obtain the point $P' = k'G$.
4. If the two points $P = P'$ holds, it means the user has the correct access permissions to be authorised to access the queried blocks S_{id}, otherwise, the EMR server should deny the query Q from the user.

- **Encrypt**(pp, Q, k')

The EMR server encrypts the queried data via the following steps.

1. Prepare the queried blocks M based upon the sequence $S_{id} \in Q$ from the user then calculate the hash value $H_M = H(M)$ of the data M.
2. Use $AES \in pp$ to encrypt M and $H(M)$ with the key k' to output the ciphertext $C = AES_{k'}(M, H_M)$. Besides, $AES'_{k'}$ is defined as the decryption process to decrypt $C = AES_{k'}(M, H_M)$ to recover the plain data M, i.e., $M = AES'_{k'}(C = AES_{k'}(M, H_M))$.
3. Return the ciphertext C to the user (data requester) to finish the authorisation and data transmission.

- **Decrypt**(pp, k, C)

If the user has all the access permissions (tokens) for the queried blocks, the user is

authorised to access these during the *Authorise* phase. It means $P = P' \Leftrightarrow kG = k'G \Leftrightarrow k = k'$ holds. Then, the user can decrypt the ciphertext C generated by the *Encrypt* phase via the following steps.

1. Decrypt C to with the key k to retrieve the plaintext $(M, H_M) = AES'_k(C) = AES'_k(AES_{k'}(M, H_M))$.

2. If the condition $H(M) = H_M$ holds, the algorithm outputs M, otherwise, it outputs $\perp$ (error).

The following Figure. 3.3 shows the workflow of the BBACS scheme.

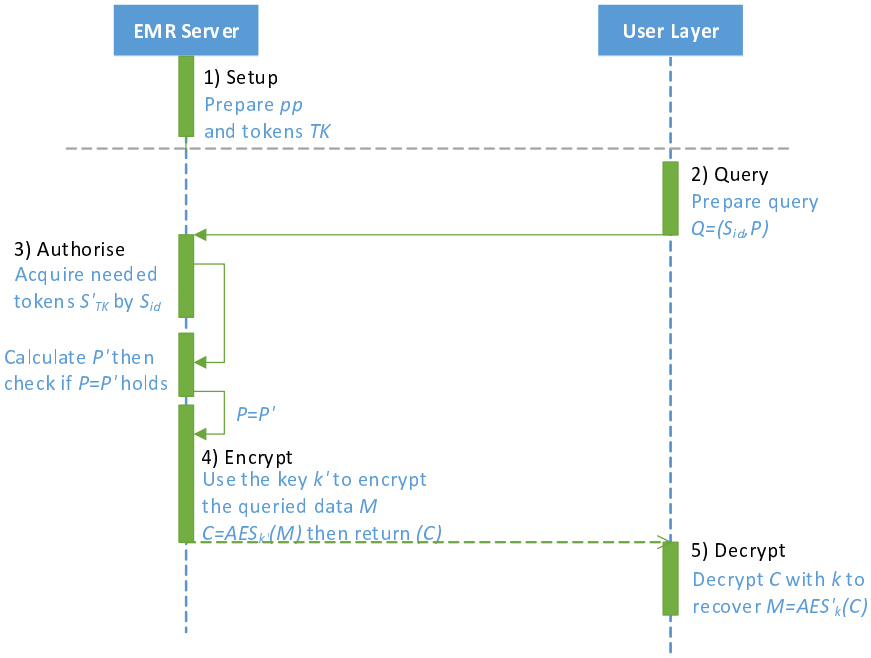

Figure 3.3: The workflow of the proposed scheme BBACS.

3.5 Analysis of performance and security

3.5.1 Experimental results

In this section, the two parts are evaluated as follows. First, the theoretical cryptographic operations are compared. Second, experiments are conducted to determine the time cost for transmitting encrypted data over Wi-Fi of BBACS. The reason for evaluating the time

cost of data transmission over Wi-Fi is that the lower time cost of data transmission via Wi-Fi can lead to a lower power cost, especially for resource-constrained eHealth devices.

As there is yet no clear best practice to be used as a baseline for comparison, I select an access control scheme based upon an agent for blockchain-based EMRs system named HDG [146] as the baseline. Note that compared with the HDG scheme that needs an agent to support access control, the scheme BBACS can verify and authorise the data access without the agent.

3.5.1.1 Theoretical comparison

The major cryptographic operation used in the scheme BBACS and the HDG scheme [146] is scalar multiplication. Hence, O_{mul} denotes an operation of the scalar multiplication for the following comparison. The result is shown in the Table 3.1. In terms of the encryption part, the scheme BBACS requires less scalar multiplication operations. Furthermore, BBACS does not use any complicated public-key cryptographic operation in the decryption.

Table 3.1: Theoretical comparison of the used cryptographic operation.

	BBACS	HDG
Encryption	$2O_{mul}$	$5O_{mul}$
Decryption	$0O_{mul}$	$1O_{mul}$

Note that there are other cryptographic operations and algorithms used in the schemes BBACS and HDG, e.g., XOR (Exclusive OR) operation, hash summary and AES-CBC (cipher-block chaining mode). However, when compared with the denoted operation O_{mul}, the time complexity of these operations and algorithms is negligible [162]. Therefore, these low time complexity operations and algorithms are not taken into account in the theoretical comparison.

3.5.1.2 Simulation comparison

In this section, the time efficiency of local computation and transmission in the two schemes is compared. Note that the two schemes are simulated. All the test results are averaged over 10 runs for each scheme. The devices used in the simulations are a conventional computer with an Intel i5-4200H processor running at 3.30GHz, and a Raspberry Pi 2 as a low-resource eHealth IoT device.

The first simulation is performed on the aforementioned conventional computer. I vary the number of the blocks requested by the user to compare the time consumption of the encryption and decryption algorithms in the two schemes (BBACS and HDG). The simulation is implemented, based upon MIRACL (Multiprecision Integer and Rational Arithmetic Cryptographic Library) [31], which can support all the necessary cryptographic operations for the two schemes. Note that the block size used in experiments is 16 Kbytes and the length of the index BLK_{id_i} and token TK_{id_i} for each block is 256 bits. The queried EMRs of patients vary from 2 to 10 with 10 blocks in each patient's EMRs. The number of the queried blocks set in the simulation varies from 20 to 100 with 10 runs for each number of the queried blocks to finally calculate the average results. Furthermore, the cryptographic security level [21] of all the implemented experiments is equivalent (and set to 128-bit security).

The comparison results are shown in the next Figure. 3.4. Since the scheme HDG involves an agent, it uses more scalar multiplication operations and AES algorithms to authorise the access and encrypt required data. As a result, the time efficiency of the scheme BBACS is significantly superior in terms of the encryption cost. On average, the scheme BBACS consumes 73% less time than the scheme HDG for encryption. Furthermore, the growth rate of the computational time for the encryption process in BBACS is lower than the relative growth rate in HDG as seen in Figure. 3.4.

Next, I keep all the above experimental parameters then repeat the simulation on a Raspberry Pi 2. The simulation result depicted in the Figure. 3.5 shows that the

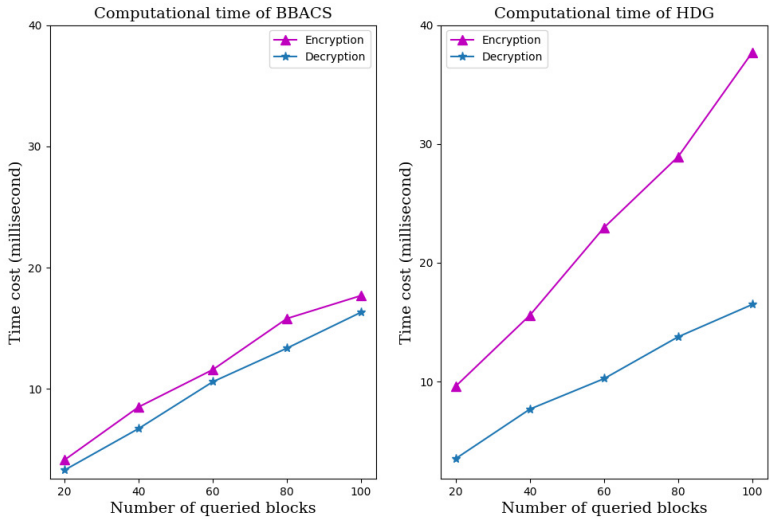

Figure 3.4: The computational time cost of encryption and decryption algorithms in the schemes BBACS and HDG on a conventional computer.

comparison of the computational time cost on the Raspberry Pi 2 is consistent with the result on a conventional computer (Figure. 3.4). On average, whilst the time consumption of BBACS is only around 29% that of the scheme HDG in terms of the encryption algorithm, the time cost of decryption in BBACS is 9% lower than that of the HDG scheme.

Furthermore, I test the time cost of transporting the encrypted data over Wi-Fi to and from the user for the two schemes. Note that the number of the queried blocks is set between 2 to 10; meanwhile, the block size is adjusted to 512 bytes. The data transmission process in the simulation is implemented based upon socket communication in Python-2.7. The results presented in Figure. 3.6 indicates that the time cost for transporting the encrypted data to (and from) the user via Wi-Fi increases linearly with the number of the queried blocks in both two schemes BBACS and HDG. However, the growth rate of the computational time for BBACS is lower than that of HDG. This is because point P (see Section IV.A.2) is the only data to authorise the access, and the data size of point P is fixed. However, HDG needs to transport the whole token list to the agent to authorise the user's data query. The length (data size) of the token list is

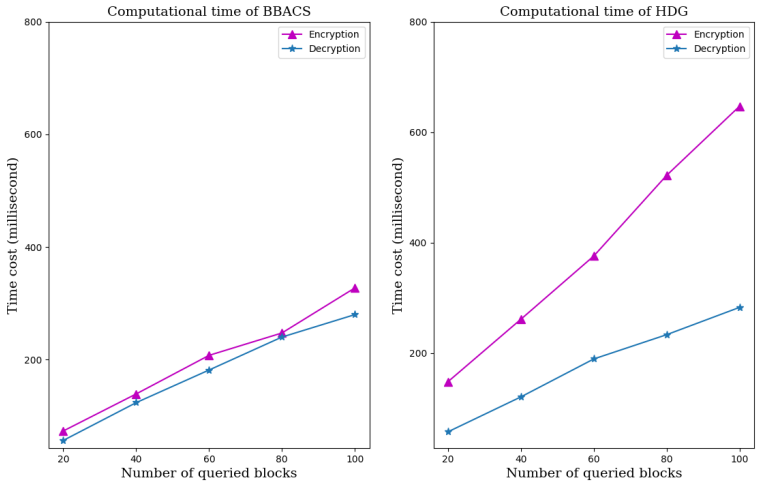

Figure 3.5: The computational time cost of encryption and decryption algorithms in the schemes BBACS and HDG on a low-resource IoT device (Raspberry Pi 2).

sensitive to the number of the queried blocks so that it leads to a higher growth rate for the time to encrypt data transmitted over Wi-Fi in HDG.

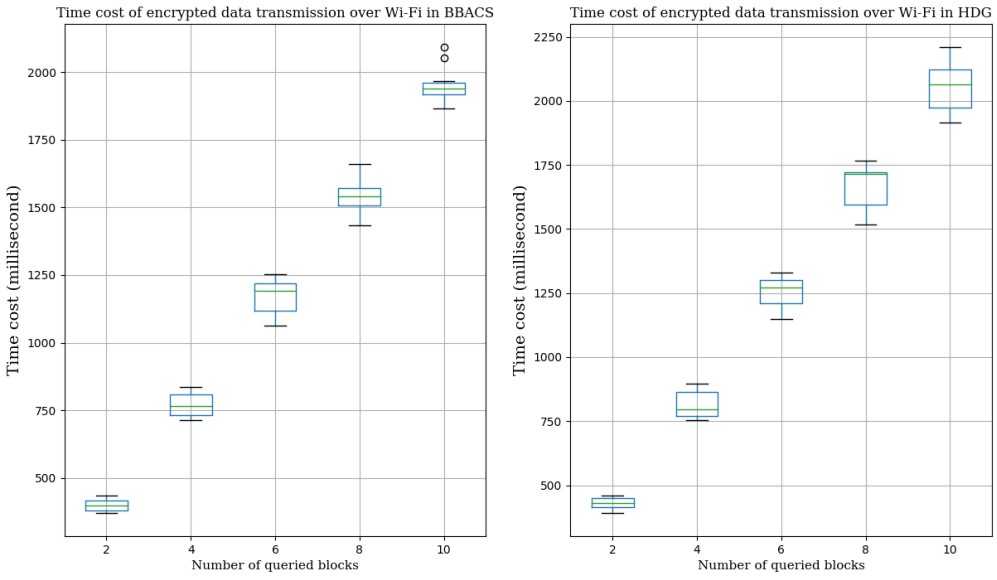

Figure 3.6: The comparison of the time cost for transporting encrypted data on a Raspberry Pi 2 in schemes BBACS and HDG.

The summary of the simulation comparison and the features of the two schemes are

given in Table. 3.2.

Table 3.2: Comparison of the simulation results and the features for BBACS and HDG.

	BBACS	HDG
Computational time cost	Low	High
Network throughput	Low	High
Trusted third party for authorisation	Unnecessary	Necessary

3.5.2 Confidentiality and integrity

In this section, I analyse the security of the proposed access control scheme BBACS from two perspectives, data confidentiality and integrity. In addition, the formal security verification of the proposed BBACS scheme is demonstrated in the last subsection.

3.5.2.1 Confidentiality

There are two potential attacks that may occur in the communications between the users and the EMR server based upon BBACS.

(i) The user tries to query a block but without the corresponding access permission. For example, the user plays the following game to query several blocks from the EMR server.

In the *Query* phase, the index sequence of the queried blocks is defined as $S_{id} = \{BLK_{id_1}, BLK_{id_2}, ..., BLK_{id_n}\}$. However, the user does not have the access permission (token) TK_{id_n} of the last block BLK_{id_n}. The partial access tokens that the user owns are presented as the sequence $\{TK_{id_1}, TK_{id_2}, ..., TK_{id_{n-1}}\}$ for the queried blocks $\{BLK_{id_1}, BLK_{id_2}, ..., BLK_{id_{n-1}}\}$. To construct a valid query, the user counterfeits the access token $TK^*_{id_n} \in \{0,1\}^\lambda$ for the block BLK_{id_n} and then organises the fake token sequence $S^*_{TK} = \{TK_{id_1}, TK_{id_2}, ..., TK_{id_{n-1}}, TK^*_{id_n}\}$. After that, the user follows step 3 and step 4 in the *Query* phase of BBACS to calculate the point $P^* = (TK_{id_1} \oplus TK_{id_2} \oplus ... \oplus TK_{id_{n-1}} \oplus TK^*_{id_n})G$. Finally, the user sends the query $Q^* = (S_{id}, P^*)$ to the EMR server.

Next, for the EMR server in the *Authorise* phase, the correct token sequence for the queried blocks S_{id} is $S'_{TK} = \{TK'_{id_1}, TK'_{id_2}, ..., TK'_{id_{n-1}}, TK'_{id_n}\}$. Then the EMR server follows the step 2 and the step 3 in *Authorise* phase to calculate the point $P^* = (TK'_{id_1} \oplus TK'_{id_2} \oplus ... \oplus TK'_{id_{n-1}} \oplus TK'_{id_n})G$. Based upon the correctness analysis above (see Section IV.B), the condition that allows the user to be authorised successfully is $P' = P^*$. Meanwhile, the condition $P' = P^*$ is equivalent to $S'_{TK} = S^*_{TK}$. Even though $\forall i \in \{1..n-1\}, TK_{id_i} = TK'_{id_i}$ $(TK_{id_i} \in S^*_{TK}, TK'_{id_i} \in S'_{TK})$ holds, the advantage $Pr[TK'_{id_n} = TK^*_{id_n}]$ is $\frac{1}{2^\lambda}$ to satisfy the condition $TK'_{id_n} = TK^*_{id_n}$, where λ represents the security parameter in the *Setup* phase. Note that $\frac{1}{2^\lambda}$ is quite small as λ is big enough so that the advantage $Pr[TK'_{id_n} = TK^*_{id_n}] = \frac{1}{2^\lambda}$ is negligible. It means that the probability of $S'_{TK} = S^*_{TK}$ is negligible as well. Hence, the user cannot be authorised to access the queried data in the *Authorise* phase since $Pr[P^* = P'] = Pr[S^*_{TK} = S'_{TK}] = Pr[TK'_{id_n} = TK^*_{id_n}] = \frac{1}{2^\lambda}$ is negligible based upon the above analysis.

(ii) The communications between the user and the EMR server could be eavesdropped upon by an evil attacker, who can acquire all the data of the communications between the user and the EMR server. The data contains the user's query $Q = (S_{id}, P)$ and the response data (C).

First, the attacker cannot retrieve the plain data M with the user's query Q and the returned data (C) because of two reasons. Firstly, the attacker cannot acquire the token sequence S_{TK} from the communications. Furthermore, based upon the ECDLP problem, the attacker cannot determine the value k with $P(= kG), G$ and pp. Secondly, the attacker cannot recover the plaintext M from the ciphertext C without the correct AES decryption key k (or k').

On the other hand, the modification in $S_{id} \in Q$ will lead to the incorrect token sequence S'_{TK} in the *Authorise* phase so that the point P' is changed as well. If the attacker modifies the point $P \in Q$ directly, the point P cannot be matched with the point P' calculated by the EMR server in the *Authorise* phase. As a result, the attacker cannot pass through the *Authorise* phase in the scheme BBACS based upon the above analysis.

3.5.2.2 Integrity

If the attacker intercepts the communications between the user and the EMR server, the attacker can challenge the integrity via forging the returned data C.

If the attacker distorts the original ciphertext C to C^*, the user can still decrypt C^* with the user's key k to retrieve the plaintext (M^*, H_M^*). However, based upon the AES algorithm, the decryption $AES'_k(C^*) = (M^*, H_M^*)$ is changed and the attacker cannot control M^* and H_M^* to make $H(M^*) = H_M^*$ hold via tampering with C^*. As a result, the step 2 in the *Decrypt* phase outputs $\perp$.

3.5.2.3 Formal verification

This section yields the formal verification of the proposed scheme BBACS by adopting the widely-used automated software-based security protocol simulator, termed "Casper/FDR" including communicating sequential processes (CSP) [163] compiler Casper [164] and a CSP model checker called Failures Divergences Refinement (FDR) [165]. CSP is a formal language to describe the interaction and states in concurrent systems to be used to model communicating and security protocols but directly writing CSP is time-consuming and error-prone [166]. Therefore, the CSP compiler Casper [164] is proposed to translate an abstract description of a security protocol into CSP. Note that automated software-based formal security verification tools have been applied and gained widely commendation among security researchers and practitioners in the last two decades. Apart from Casper/FDR, various other formal security analyzers can be found in the literature [167, 168].

The security properties of the proposed protocol BBACS are modelled using Casper and the CSP output is analysed with FDR. In the model, the user and EMR sever are represented by two roles Alice and Bob, respectively. The version of Casper used is 2.1 and the version of FDR used is FDR4. The results are demonstrated in Figure 3.7. Through such an analysis, it is shown that the proposed scheme BBACS is secure enough to ensure the confidentiality and integrity of its parameters in the communication for

authorisation.

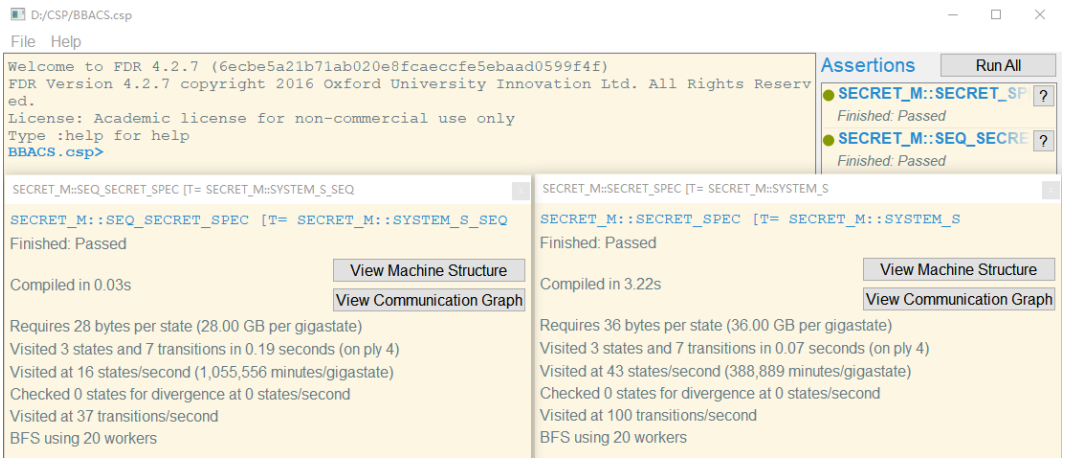

Figure 3.7: The formal verification results of BBACS with using Casper/FDR.

Overall, the proposed scheme BBACS can offer sufficient security to ensure that both the confidentiality of the transported data and the data integrity are protected based upon the security analysis and formal security verification above when attacks happen to the communication between users and the EMR server.

3.6 Summary

In this chapter, I proposed what I believe is the first access model without an agent layer or gateway support to realise the access control to authorise the data access from users to the blockchain-based EMR server. The proposed authorisation scheme BBACS can validate the access permission for each queried block to authorise the data queries from users. As a result, a blockchain-based EMR server can respond to the data requesters without the assistance of agent(s) or leaking unauthorised EMRs, especially for resource-constrained devices used in IoT systems for eHealth.

However, two aspects can be considered further to improve the presented BBACS scheme. Firstly, in the preparation phase (initialisation) of the whole system, key distribution should be considered to ensure each user can obtain the corresponding access

tokens before querying EMRs. Secondly, in the BBACS scheme, authorisation is designed at the block level i.e., an authorised user can acquire all the data in the queried blocks. But if a patient wants to restrict the data in an EMR block that a user can access to, the granularity of queries and authorisation should be defined and improved since the BBACS scheme only implements block-level authorisation. Henceforth, in the next chapter, I focus on addressing these two concerns mentioned above.

Chapter 4

Granular Authorisation for Blockchain-based EMRs

In this chapter, an advanced access authorisation scheme for blockchain-enabled EMRs is presented including a key distribution scheme and a granular authorisation scheme supporting three different types of flexible queries to address the discussed two issues in the last chapter (see Section 3.6) for the IoT-eHealth scenario.

4.1 Motivation – need for granular access authorisation supporting flexible queries for blockchain-enabled EMRs

To realise key distribution for accessing blockchain-enabled EMRs in IoT eHealth scenarios, where numerous IoT devices can be involved, cloud computing has been widely applied as a service infrastructure for IoT devices [169]. Therefore, trusted infrastructures of cloud computing are essential to large-scale IoT eHealth services. However, the expense of constructing a complete private trusted cloud computing infrastructure is so high that many small and medium scale enterprises cannot afford to do this. Instead, their data collection, key distribution and many other IoT eHealth services rely on public

clouds, which have the added concern from users about how to protect their data privacy when it is transported onto untrusted public clouds [170]. In order to realise the key distribution through untrusted public clouds for IoT devices to access the blockchain-enabled EMRs, a new key distribution scheme called semi-outsourcing key distribution (SOKD) is proposed.

Since EMRs hold personal data about patients that can be confidential in different ways to different stakeholders, approaches to construct flexible and granular access authorisation to blockchain-enabled EMRs is needed. For example, if a disease analyst tries to determine the incidence of different diseases in an EMR database, the data obtained by the analyst should only reveal the disease name in each block. When a nurse wants to check the drug injection doses of several patients, the information revealed to the nurse should only include patient names, IDs and drug dosage. But under the implemented authorisation granularity (block level) in Chapter 3, other private data of patients, e.g., other data such as social security numbers and home addresses may also be shown in the block, which are not related to the work of the analyst and the nurse, yet can be disclosed. Hence, the block-level granularity is not fine-grained enough for flexible queries and authorisation. This challenge motivates me to present a new access architecture for blockchain-enabled EMRs called granular access authorisation supporting flexible queries (GAA-FQ) to achieve fine-grained (i.e., more precise granularity control) authorisation and flexible queries.

4.2 System model

4.2.1 Key distribution model

There are three types of entity for key distribution in the system, which are the IoT devices, public (untrusted) clouds, and key generation centre (KGC) respectively. The system infrastructure is presented in Figure 4.1.

Organisations and companies fully control and trust their KGC but it is difficult

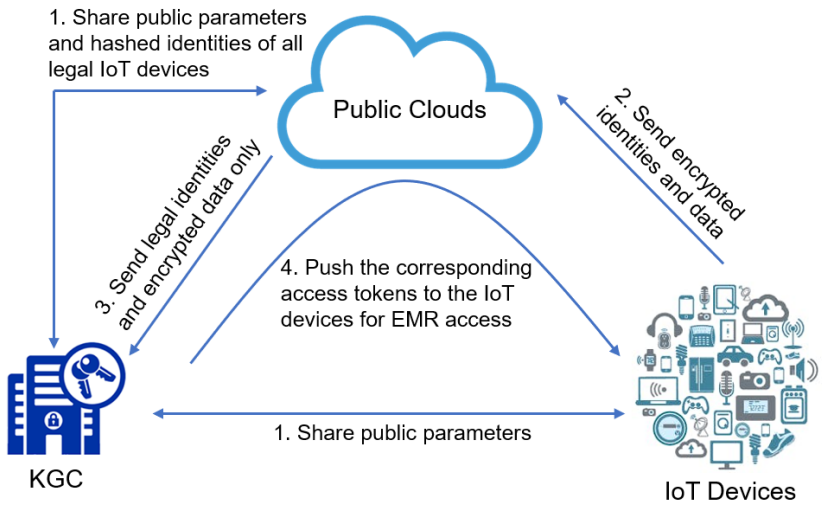

Figure 4.1: The system model and key distribution schematic of the SOKD scheme.

to use KGC to distribute the keys to the corresponding IoT devices directly because being connected by numerous IoT devices, can overburden the KGC's resource cost in terms of communication and computation and may lead to DoS/DDoS attacks to the KGC. Therefore, a public cloud service is usually applied as a gateway between the KGC and IoT devices for identity authentication and key distribution, which has enough resource capacity for computation and communication to prevent DoS/DDoS attacks [171, 172]. However, since public clouds may leak private information, they are untrusted when introduced (as the gateway) for data exchange in the process of key distribution. Henceforth, this also implies an additional IoT security data requirement which requires avoiding private information leakage from the IoT device to the clouds and from the clouds to the KGC.

The schematic work phases of the SOKD scheme model (Figure 4.1) are described as follows. The first phase is used to broadcast the public security parameters to all the entities from the KGC, which share an identity list with the untrusted public clouds at the same time. Note that all the elements of the identity list are the IoT devices' hashed identities. Then, an IoT device can transmit its hashed identity and encrypted data (e.g.,

symmetric key for further encryption and decryption) to the untrusted public in phase 2. After receiving the hashed identities and the encrypted data, the public clouds can validate the identities of the data senders (the IoT devices) by querying the identities in the identity list. If the IoT devices' identities are valid, the untrusted public clouds send the corresponding encrypted data from the identified IoT devices to the KGC; otherwise, the key distribution should be aborted. In the last phase, the received encrypted data are decrypted by the KGC to obtain the plain data and check the data integrity to detect tampering. After that, if the plain data is intact, the KGC uses the symmetric key in the plain data (provided by the IoT device) to encrypt the access tokens and then pushes the encrypted access tokens (T) to the corresponding IoT device via the public clouds to accomplish the key distribution for EMR access later.

4.2.2 Access model

There are three layers in the access model (Figure 4.2): user layer, agent layer and storage layer. The storage layer used for saving blockchain-enabled EMRs is usually constructed by a distributed storage system (e.g., IPFS [173]) in the trusted clouds. Next, I illustrate the functions of each layer in Figure 4.2.

User layer. This layer represents data inquirers e.g., doctors, patients, data analysts and even monitoring devices of health. The data inquirers initiate queries to obtain data from the storage layer. All the queries are processed through the agent layer. Note that doctors and patients normally query one or several blocks, while data analysts usually query values of different attributes (columns) in blocks, e.g., to analyse the incidence of morbidity in a sample of the population.

Agent layer. There are two major aims of the agent layer. The first aim is to aggregate the queried data since the data could be entire blocks or columns from blocks. The second aim is to check the inquirers' access permission for the queried data and to authorise the access from the valid inquirers. In this proposed model, the agent layer can check the access permission for every block and attribute. If the inquirers have all

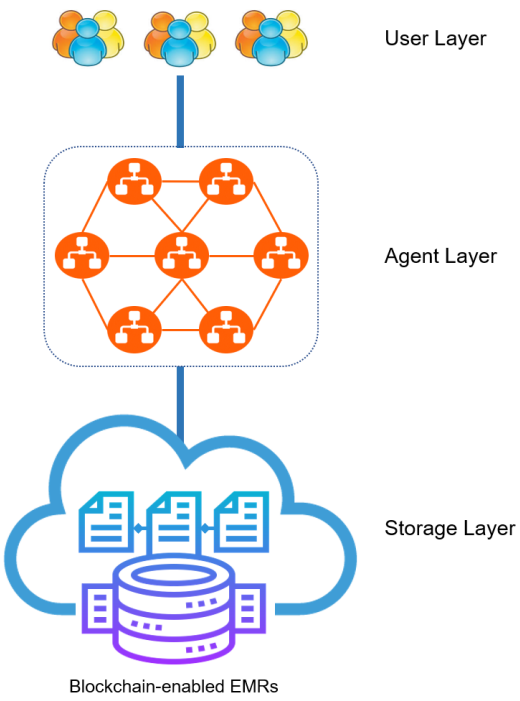

Figure 4.2: The proposed access model for accessing blockchain-enabled EMRs.

the access permission for the queried blocks and attributes, the agent layer can authorise the access and then return the requested data. Otherwise, the agent layer should deny the access request. To avoid a single point of failure (SPOF), the authorisation can be controlled by a consensus mechanism while using multiple agent nodes in design. Note that the style of the returned data is determined by the granularity of the queries. It can be many blocks or only the data of several attributes in certain blocks.

Storage layer. In this layer, all EMRs used for eHealth are stored in a blockchain-enabled architecture. Meanwhile, this layer should provide the queried data to the agent layer.

4.2.3 Granularity control of EMR queries

To satisfy different query requirements from different medical staffs, I define three different fine-grained queries for different users: (i) *block query*, users request one or certain

blocks on a chain; (ii) *attribute query*, users only request all the data of particular attributes; (iii) *mixed query*: users aim to obtain the data of certain blocks and particular attributes from specified blocks. An example for demonstrating the three defined granular levels of queries in a blockchain is shown in Figure. 4.3.

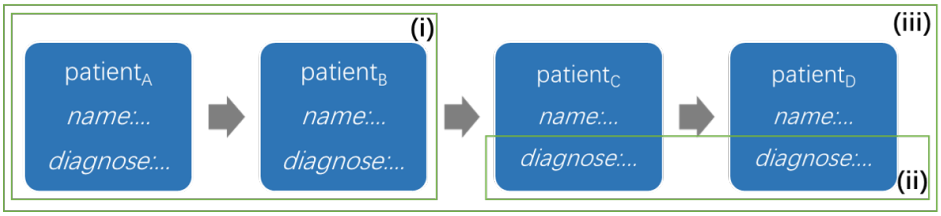

Figure 4.3: The example of three levels of granularity for queries.

Note that the proposed schemes SOKD and GAA-FQ focus on key distribution for authorisation and fine-grained data access authorisation but not on the blockchain properties such as the consensus mechanism.

4.3 Algorithm description

In this section, the scheme of key distribution is presented at first, which is followed by the granular access authorisation scheme illustrated in the second subsection.

4.3.1 Semi-outsourcing key distribution

The proposed key distribution scheme, SOKD (Semi-outsourcing Key Distribution) consists of six phases (algorithms), which are described under the hardness assumption of elliptic-curve Diffie-Hellman (ECCDH) problem (see Section 2.1.1.3).

- **Hardness assumption**

Elliptic-curve Diffie-Hellman (ECCDH) Assumption. Let $E_p(a, b)$ be a cryptographic secure elliptic curve with the prime field $\mathbb{F}_p$. For any point $P \in E_p(a, b)$ and two random integers $u, v \in_R \mathbb{Z}_p^*$, any probabilistic polynomial-time algorithm $\mathcal{A}$ computes

uvP with its advantage:

$$Adv^{ECCDH}_{\mathcal{A},E_p(a,b)} = Pr[c = uvP | u \in_R \mathbb{Z}^*_p, c = \mathcal{A}(P, uP, vP)].$$

The ECCDH assumption can hold if for any probabilistic polynomial-time algorithm $\mathcal{A}$, its advantage $Adv^{ECCDH}_{\mathcal{A},E_p(a,b)}$ is negligible.

• **Setup(λ)**

This algorithm uses the security parameter λ to generate the public parameters pp in five steps.

1. Pick a cryptographic secure elliptic curve group $\mathbb{G}$ with a base point G on the curve, where the order of $\mathbb{G}$ is p.

2. Select three cryptographic secure hash functions: $H_1 : \{0,1\}^* \rightarrow \{0,1\}^\lambda$, $H_2 : \mathbb{G} \rightarrow \{0,1\}^{2\lambda}$ and $H_3 : \mathbb{G}^3 \times \{0,1\}^{2\lambda} \rightarrow \{0,1\}^\lambda$.

3. Choose a symmetric encrypting and decrypting algorithm. Note that the selected algorithm to describe the SOKD scheme is the Advanced Encryption Standard (AES) [161].

4. The KGC calculates the hashed identity $H_{ID} = H_1(ID)$ for all IoT devices in the network to create an identity list including all IoT devices' hashed identities as the elements, $List - H_{ID}$, then shares $List - H_{ID}$ with the untrusted public clouds.

5. Output $pp = (\mathbb{G}, p, G, H_1, H_2, H_3, AES)$.

• **KGCInitialise(pp)**

This algorithm randomly picks two numbers $a, b \in_R \mathbb{Z}^*_p$ and calculates the key pair:

$$(pk, sk_1, sk_2) = ((A = aG, B = bG), (a), (b)).$$

The public keys A and B can be broadcast in the whole network. However, the private key $sk_2 = b$ is only shared with the public clouds , meanwhile, the private key,

$sk_1 = a$, is kept such that it is known only by the KGC secretly. Note that it is assumed that all IoT devices can establish secure connections with the public clouds to obtain the correct pp and pk before the key distribution.

- **Encrypt**(pp, M, ID, pk)

For the given sent data $M \in \{0,1\}^\lambda$ and the identity of the IoT device $ID \in \{0,1\}^\lambda$, the algorithm outputs the encrypted ciphertext $C = (C_1, C_2, C_3, C_4)$ via the following steps:

1. Compute $H_M = H_1(M)$, and $H_{ID} = H_1(ID)$.

2. Use AES to encrypt data M with the key H_{ID} then get the ciphertext $AES_{H_{ID}}(M)$. For decrypting $AES_{H_{ID}}(M)$ to recover the plaintext M, $AES'_{H_{ID}}$ is defined as the decryption process: $M = AES'_{H_{ID}}(AES_{H_{ID}}(M))$.

3. Pick two random numbers $r_1, r_2 \in_R \mathbb{Z}_p^*$ then compute

   $C_1 = r_1 G,$

   $C_2 = r_2 G,$

   $C_3 = AES_{H_{ID}}(H_M || M) \oplus H_2(r_1 A),$

   $C_4 = H_{ID} \oplus H_3(r_2 B, C_1, C_2, C_3).$

Note that the plain data M can involve the symmetric key $k \in M$ for encryption and decryption used by the IoT device.

- **Authentication**(pp, C, sk_2)

The public clouds receive the ciphertext C and computes $C_4 \oplus H_3(bC_2, C_1, C_2, C_3)$ to recover H_{ID}. Then, the public clouds check if H_{ID} belongs to the hashed identity list $List - H_{ID}$. If $H_{ID} \in List - H_{ID}$, the public clouds allow the transmission to send the ciphertext $C' = (H_{ID}, C_1, C_2, C_3)$ to the KGC; if $H_{ID} \notin List - H_{ID}$, the public clouds

deny the request for data transmission.

• **Decrypt**(pp, C', sk_1)

The KGC can execute the next steps to retrieve the plaintext M from the received ciphertext $C' = (H_{ID}, C_1, C_2, C_3)$:

1. Recover $AES_{H_{ID}}(H_M||M)$ via computing
$$C_3 \oplus H_2(aC_1).$$
2. Decrypt $AES_{H_{ID}}(H_M||M)$ with the key H_{ID}
$$H_M||M = AES'_{H_{ID}}(AES_{H_{ID}}(H_M||M)).$$
3. If $H_1(M) = H_M$ holds, this algorithm outputs M; otherwise, it outputs $\perp$ (error).

• **KeyDistribute**(pp, M, T)

The KGC first determines the access tokens T that the corresponding IoT device should have by its identity. After that, the KGC extracts the symmetric key k from M then encrypts the access tokens T and the hash value $H_{KD} = H_1(T)$ of T by computing $C_{dist} = AES_k(T, H_{KD})$. Finally, the KGC pushes C_{dist} to the corresponding IoT device via the public clouds.

After obtaining C_{dist}, the IoT device can retrieve its access tokens T by computing $AES'_k(C_{dist}) = AES'_k(AES_k(T, H_{KD})) = T, H_{KD}$. If $H_1(T) = H_{KD}$, the access tokens T are intact; otherwise, it means C_{dist} has been tampered and the key distribution should be aborted.

The entire workflow of this SOKD scheme is depicted in the Figure. 4.4.

4.3.2 Granular access authorisation supporting flexible queries

In this part, I propose the authorisation scheme, GAA-FQ (Granular Access Authorisation supporting Flexible Queries) based upon Lagrange interpolating polynomials and SSS (see Section 2.1.2.4).

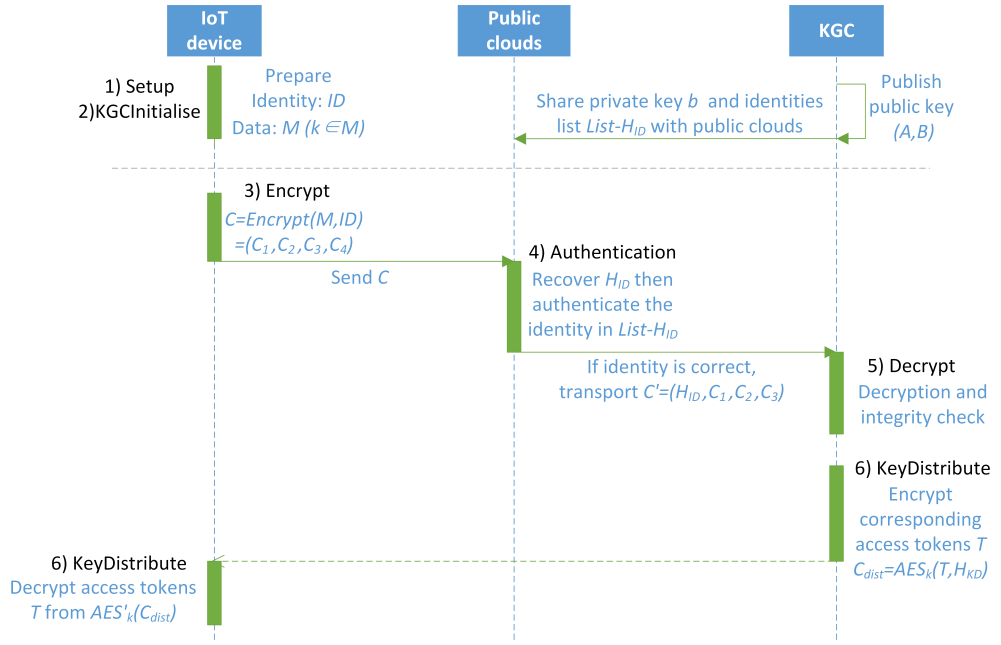

Figure 4.4: The workflow of the designed scheme SOKD.

- **Setup(λ)**

This procedure outputs public parameters pp with the security parameter λ using the following steps.

1. Generate a big prime q.

2. Generate a random integer token $BT \in \mathbb{Z}_q$ for each block B on the blockchain and write BT in B. Note this step should be executed when a new block is added to the blockchain to keep the integrity of the whole blockchain.

3. Generate a random integer token $AT \in \mathbb{Z}_q$ for each attribute A included in the blocks of the blockchain. After generating all access tokens (BT and AT), the storage layer should allow the KGC to know all the tokens for the key distribution in section 4.3.1. Note I assume that all the valid users have acquired the corresponding tokens that they should have through the prior use of a key distribution scheme SOKD before they request data from the blockchain.

4. Select one secure cryptographic hash function $H : \{0,1\}^* \rightarrow \{0,1\}^\lambda$.

5. Select a symmetric encryption algorithm, e.g., AES (Advanced Encryption Standard) [161].

6. Output $pp = (q, H, AES)$.

- **Query**(pp)

The user (data inquirer) prepares the query Q via following steps.

1. Choose the type of the query T from the defined three granularity queries. Note that for illustrating the remaining parts of the proposed scheme clearly, I assume the type of the query is (i) *block query* and the amount of queried blocks is $n \geqslant 1$.

2. Prepare the sequence S_1 including all the identities of requested blocks. For example, S_1 should include all the identities of blocks n: $S_1 = \{B_{id_i} | i = 1...n\}$ based upon the prior assumption in *Setup*.

3. Prepare the sequence S_2 including all the tokens of requested blocks. For the given sequence S_1 in step 2, $S_2 = \{BT_i | i = 1...n\}$.

4. Calculate the hash value H_T of S_2 via $H_T = H(BT_1, BT_2, ..., BT_n)$.

5. Send the query $Q = (S_1, H_T)$ to the agent layer via secure connections.

- **Authorise**(pp, Q)

The agent layer validates the access permission in Q from the user via the following steps.

1. Repeat step 3 in *Query* with $S_1 \in Q$ to obtain $S_2' = \{BT_i' | i = 1...n\}$ from the storage layer.

2. Calculate the hash value H_T' of S_2' via $H_T' = H(BT_1', BT_2', ..., BT_n')$.

3. If $H_T = H_T'$ holds, it means the user can be authorised to access the queried data and the next phases are conducted, otherwise, the agent layer should deny the

access request.

- **Encrypt**(pp, Q, S_2')

The agent layer encrypts the queried data via the following steps.

1. Acquire the queried data M based upon $S_1 \in Q$ from the storage layer then calculate the hash value H_M of the data M: $H_M = H(M)$.

2. Generate a secure key $k \in \mathbb{Z}_q$.

3. Use AES to encrypt M with key k to get the ciphertext $C = AES_k(M, H_M)$. For decrypting $AES_k(M, H_M)$ to recover the plain data M, AES_k' is defined as the decryption process: $M = AES_k'(C = AES_k(M, H_M))$.

4. Follow the *SSharing.Generation* in Section 2.1.2.4 to construct a polynomial $f(x)$ of degree n with k and S_2':
$$f(x) = k + BT_1'x + BT_2'x^2 + ... + BT_n'x^n \ (mod\ q).$$

5. Generate a random integer $x_p \in \mathbb{Z}_q$ and calculate a point $P(x_p, y_p = f(x_p))$.

6. Return (C, P) to the inquirer to finish the authorisation and data transmission.

- **Decrypt**(pp, Q, C, P)

The user has all the valid access permissions for the queried data and can decrypt the ciphertext C after the *Authorise* and the *Encrypt* phases via the following steps.

1. Use the sequence $S_2 = \{BT_i | i = 1...n\}$ organised in former *Query* phase to construct a polynomial $g(x)$ based upon the *SSharing.Generation* in Section 2.1.2.4 :
$$g(x) = a_0 + BT_1x + BT_2x^2 + ... + BT_nx^n \ (mod\ q).$$

2. Follow the *SSharing.Reconstruction* in Section 2.1.2.4 to recover the key $k = g(0) = a_0 \in \mathbb{Z}_q$ for AES decryption with $P(x_p, y_p)$:

$$k = y_p - BT_1 x_p - BT_2 x_p^2 - ... - BT_n x_p^n \ (mod\ q).$$

3. Decrypt C to retrieve the plaintext $(M, H_M) = AES'_k(C) = AES'_k(AES_k(M, H_M))$.

4. If $H(M) = H_M$ holds, this algorithm outputs M; otherwise, it outputs $\perp$ (error).

The following Figure. 4.5 shows the workflow of the proposed scheme GAA-FQ.

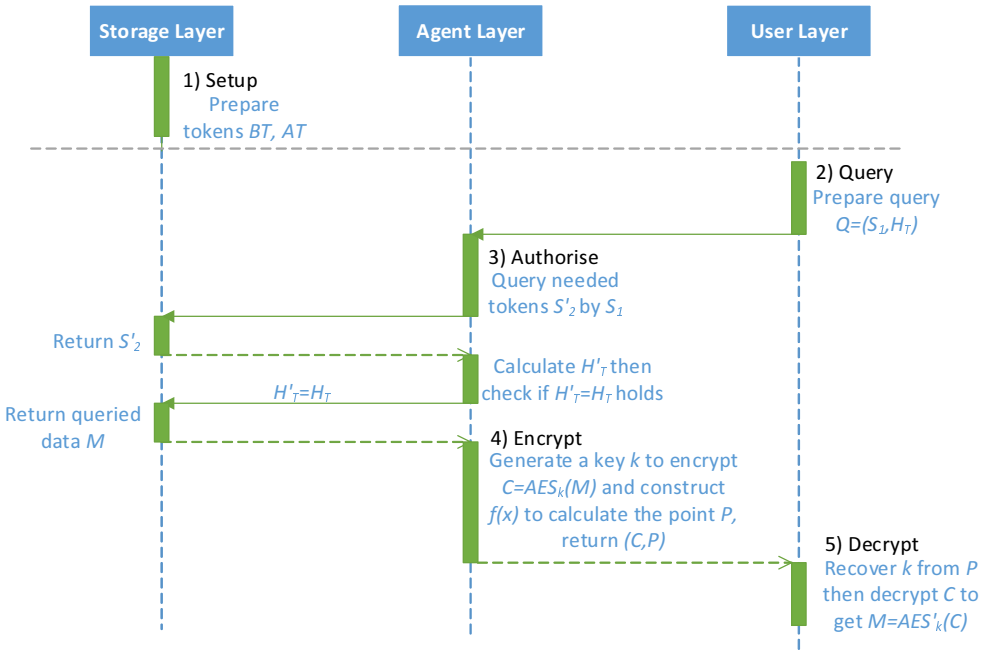

Figure 4.5: The workflow of the proposed authorisation scheme GAA-FQ.

4.4 Analysis of performance and security

In this section, an analysis of the performance from the designed simulations is presented in the first part for the proposed two schemes SOKD and GAA-FQ. After that, the confidentiality and integrity of the two schemes are briefly analysed when they confront different kinds of attacks.

4.4.1 Simulation results

The performance analysis mainly involves the time cost for computation and communication and the throughput measured in communications over Wi-Fi. To fit the resource-

constrained scenario with IoT devices, the simulations are conducted in not only several conventional computers but also a Raspberry Pi 2 as a low-resource IoT device.

4.4.1.1 SOKD

The performance of the proposed scheme SOKD is compared with the scheme JAAS (Java Authentication and Authorisation Service) in [174] used as a baseline in terms of two aspects: time efficiency and network throughput. To be specific, the time consumption of the data transportation over Wi-Fi and the local computation in the IoT device is compared. Then, the network throughput including the quantity and the size of the transmitted packages in the two schemes are evaluated. To build up the experiments, the low-resource IoT hub, Raspberry Pi 2, is selected as the IoT device. Meanwhile, a conventional laptop with an Intel processor (3.30GHz) is used to perform as a node in the public clouds.

I implement SOKD and the scheme JAAS in [174] based upon MIRACL [31], a cryptography SDK that can support all the required operations on elliptic curves and provide the needed hash functions and cryptographic algorithms. All the secure parameters and implemented experiments use an equivalent cryptographic security level (128-bit) [21] for both two schemes. Then the evaluation of the transmitted packages' quantity and size is implemented based upon TCP (Transmission Control Protocol) socket communication. The transmitted data size is set to 128 Bytes, and the used certificate size is 1024 Bytes (1 KB). In the experiments, AES-128 is selected for data encryption and decryption with SHA-256 as the hash algorithm. For public key operations, the elliptic curve I use is *secp160*.

- **Time Efficiency Comparison**

For comparison of the time efficiency, the experiment is executed 5 rounds with 10 times in each round. The average time cost (for each round) of transporting data over Wi-Fi is shown in Figure. 4.6. Compared with the time cost of the scheme JAAS in [174], the reduction of the time cost in SOKD is about 75% on average.

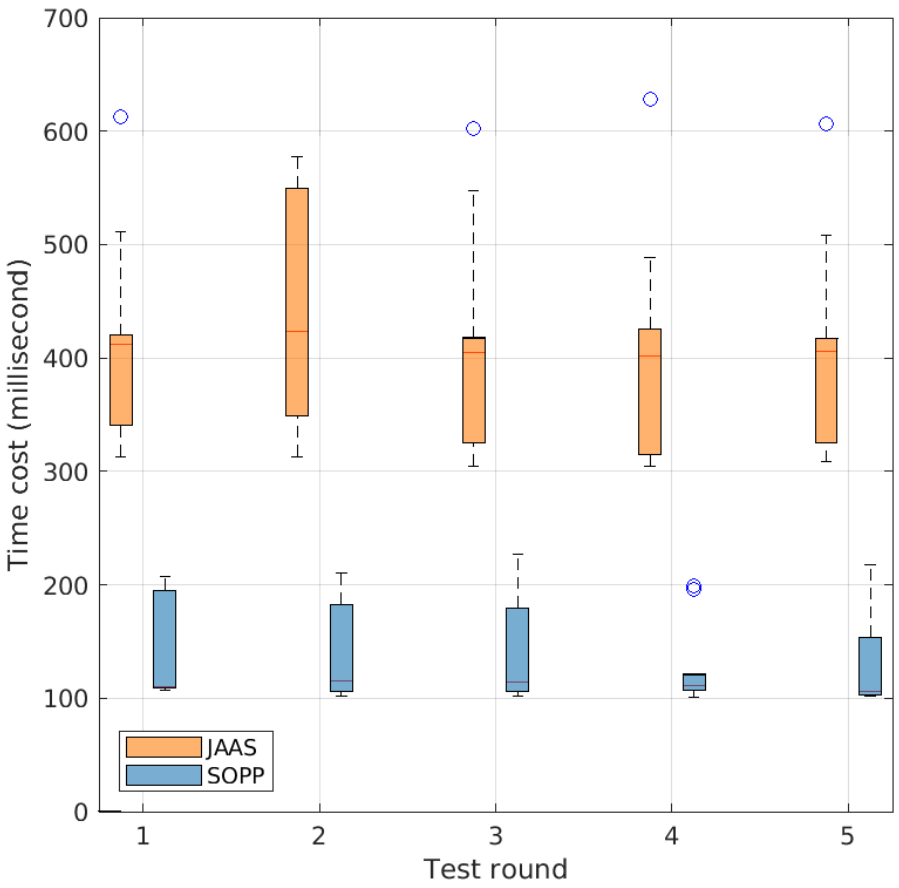

Figure 4.6: The comparison of time cost for data transmission over Wi-Fi.

Meanwhile, the average total time cost including the time consumption of both local computation and data transmission is depicted in the Figure. 4.7. Although the time cost of local computation in SOKD is twice that of the scheme JAAS, the total time cost of SOKD is around 60% less than that of the scheme JAAS in [174] because the data size required for authentication in the scheme SOKD is much smaller, leading to a much lower time cost for data transmission over Wi-Fi when compared with JAAS.

• **Communication Throughput Comparison**

In this experiment, the transmitted data size is set to be 256 bytes and 512 bytes respectively to obtain the average results (over 100 times), which are determined by monitoring the network throughput. For each data size, the average size and quantity of the sent packages by the IoT device (Raspberry Pi 2) for data transmission are summarised

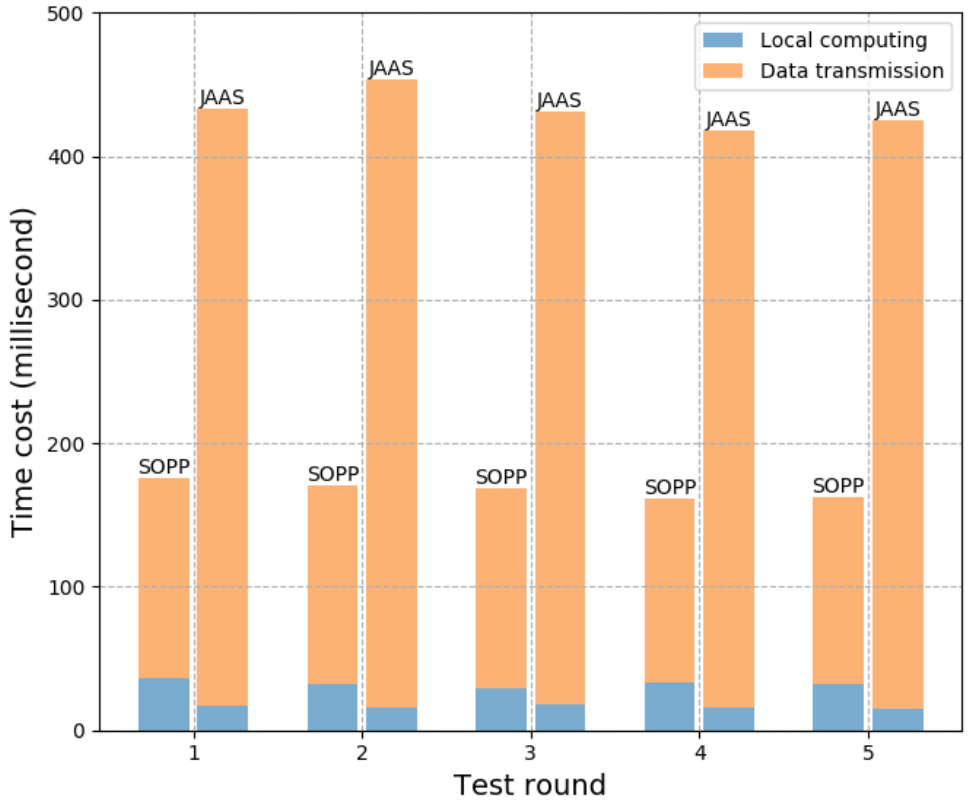

Figure 4.7: The comparison of the total time cost for local computing and data transmission.

in Table. 4.1. Compared with the package size and quantity of the scheme JAAS in [174], the size and the quantity of the sent packages are reduced by around 45% and 60% respectively because less data I require for authentication in SOKD.

Table 4.1: A comparison of average data quantity and total data size in each data transmission.

	Transported data size (byte)	Quantity	Total size (byte)
JAAS [174]	256	27	4435
SOKD	256	15	1866
JAAS [174]	512	58	9094
SOKD	512	33	3850

To summarise, the performance experiments demonstrate that the proposed scheme SOKD costs more time for local computing when compared with the scheme JAAS in [174]. However, the comparisons indicate that the proportion of the time cost for local

computation (about 20% in SOKD and 5% in JAAS [174]) is much smaller than the proportion of the time cost for data transmission, therefore, the effect of the time cost of local computing is much less important. Meanwhile, compared with the scheme JAAS in [174], SOKD costs less time for data transmission, while the total transported data size is significantly decreased to achieve a lower network throughput in the experiments. Hence, the resource-constrained IoT devices can save more energy and lead to a longer battery life [175]. In conclusion, I state that the proposed scheme SOKD is more appropriate for data collection when these use resource-constrained IoT devices as data providers.

4.4.1.2 GAA-FQ

In this part, the computational time efficiency for performance of GAA-FQ is evaluated with respect to the time cost for transmitting encrypted data over Wi-Fi. The reason for evaluating the time cost of data transmission over Wi-Fi is that a lower usage time of Wi-Fi means lower power cost for resource-constrained eHealth devices when transmitting or receiving data. Since there is as yet no clear best practice to be used as a baseline for comparison, I select a granularity access authorisation scheme based upon cloud computing named ESPAC [176] as the baseline. ESPAC is also applied in eHealth but uses conventional (non-blockchain) data structure. All the results are averaged over 10 runs for each number of the used tokens (*resp.* attributes). Note that the devices for the simulation use are a conventional computer with an Intel i5-4200H processor running at 3.30GHz, and a Raspberry Pi 2 as a low-resource eHealth device.

I first vary the number of the tokens used for authorisation in GAA-FQ (*resp.* attributes in ESPAC) to compare the time taken for the encryption and decryption algorithms in the two schemes on the above conventional computer. The simulation is implemented using MIRACL [31] and *cpabe* toolkits [38]. They can support all the necessary symmetric-key and asymmetric-key cryptographic algorithms, and ciphertext-policy attribute-based encryption (CP-ABE). Note that the number of the tokens (*resp.* attributes) used in the authorisation varies from 2 to 10 with 10 runs for each num-

ber of tokens to acquire the averaged results. All the experiments use an equivalent cryptographic security level (128-bit security) [21].

The simulation results are depicted in Figure. 4.8. Since the bilinear pairing operations cost much more time in ESPAC, the time efficiency of the scheme GAA-FQ is significantly superior. On average, the time consumption of GAA-FQ is only about 6% that of ESPAC in terms of the encryption and decryption algorithms. In addition, the time cost's growth rate of the encryption and decryption algorithms is lower than the relative growth rate in ESPAC as shown in Figure. 4.8.

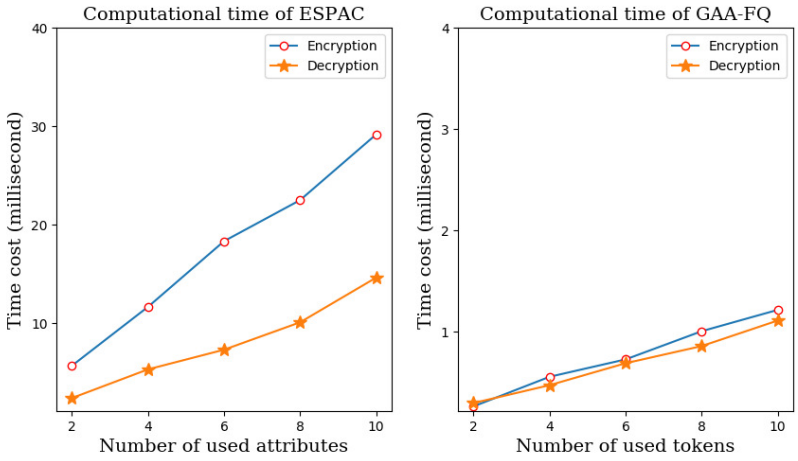

Figure 4.8: The computational time cost of encryption and decryption algorithms in the schemes ESPAC and GAA-FQ on a conventional computer (PC).

On the other hand, the theoretical comparison of cryptographic operations also shows a clear difference. The two major cryptographic operations used in ESPAC [176] are bilinear pairing and scalar multiplication. O_{pair} denotes an operation of the bilinear pairing, and O_{mul} an operation of the scalar multiplication in ESPAC. The notation O_{exp} represents the modular exponentiation operation needed in GAA-FQ. Because the used quantity of the cryptographic operations is determined by the number of required tokens in GAA-FQ (*resp.* attributes in ESPAC), I assume the number of the tokens (*resp.* attributes) used in the authorisation of the two schemes is k for the equal-scale comparison of used operations. The concrete results are given in Table 4.2.

Table 4.2: Theoretical comparison of used cryptographic operations.

	GAA-FQ	ESPAC
Encryption	kO_{exp}	$1O_{pair} + 2kO_{mul}$
Decryption	kO_{exp}	$k(2O_{pair} + O_{mul})$

Note that there are other cryptographic algorithms used in schemes GAA-FQ and ESPAC, e.g., hash summary and AES. However, time complexity of these algorithms is negligible [162] when compared to the denoted operations. Hence, these algorithms are not considered in the above comparison.

Next, I repeat the above simulation on the Raspberry Pi 2 with only changing the number of the used tokens (*resp.* attributes) to 5, 10 and 15. Meanwhile, I test the time cost of encrypted data transmission over Wi-Fi in the two schemes. The data transmission is implemented based upon Python-2.7 socket communication. The length of the plain response data from the agent layer to the user layer is 256 bytes.

The simulation results in Figure. 4.9 show that the comparison results of the computational time cost on the Raspberry Pi 2 are consistent with the results on the conventional computer. On the other hand, for the transmission efficiency, the time cost of the encrypted data transmission over Wi-Fi increases linearly with the number of attributes in ESPAC. However, the corresponding time cost in GAA-FQ is lower and kept stable i.e., it is non-sensitive to the number of tokens. This is because the additional contents in the encrypted data of ESPAC include all the used attributes for the following decryption, while in GAA-FQ, the additional part is only the contents of one point regardless of the number of the used tokens.

4.4.2 Confidentiality and integrity

In this section, I analyse the security of the key distribution scheme SOKD in terms of confidentiality, authenticity and integrity, and the authorisation scheme GAA-FQ from two angles, data confidentiality and integrity. Then, the formal security verifications of the schemes SOKD and GAA-FQ are illustrated. Additionally, a formal analysis

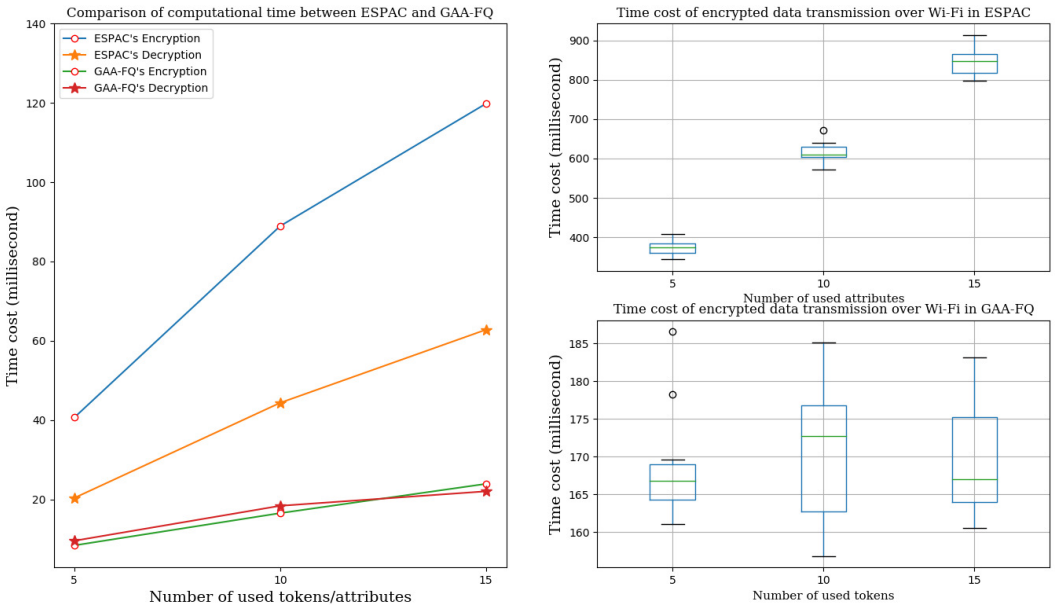

Figure 4.9: The comparison of the computation and transmission time cost on a Raspberry Pi 2 for ESPAC and GAA-FQ.

of SOKD's theoretical confidentiality is presented in Appendix B.1. Meanwhile, the formal analyses of GAA-FQ's theoretical confidentiality and integrity (unforgeability) are illustrated in Appendix B.2.

4.4.2.1 SOKD

- **Confidentiality**

The focus of the scheme's confidentiality is to ensure the untrusted public clouds cannot decrypt the encrypted data part C_3 of C in the *Authentication* phase and C_{dist} in the *KeyDistribute* phase.

After receiving the encrypted data C from the IoT device, the public clouds can only retrieve H_{ID} with the algorithm *Authentication* in Section 4.3.1. If the public clouds attempt to decrypt C_3, they need an algorithm to calculate a number r^* equal to a or r_1 (because $AES_{H_{ID}}(H_M||M) = C_3 \oplus H_2(aC_1) = C_3 \oplus H_2(r_1 A)$), where the public key (A, B) and the ciphertext C are known. However, the public clouds have no probabilistic polynomial-time algorithm to use $G, C_1 = r_1 G, A = aG$ to calculate $r^* = a$ (or r_1) based

upon the ECCDH assumption. Hence, the advantage of the untrusted public clouds $\mathcal{B}$

$$\begin{aligned} Adv_{\mathcal{B}} &= Pr[c = M | c = \mathcal{B}(pp, C, sk_2)] \\ &= Pr[c = a \vee c = r_1 | c = \mathcal{B}(pp, C_3, H_{ID}, sk_2)] \end{aligned}$$

is negligible for generating r^* to recover M from C successfully, which means the confidentiality of M can be secured in the phase *Authentication*.

In the *KeyDistribute* phase, since the public clouds do not know the symmetric key k to compute C_{dist}, the access tokens T cannot be recovered by computing $AES'_k(C_{dist}) = AES'_k(AES_k(T, H_1(T)))$. Hence, the confidentiality of T can be maintained in this phase.

• **Authenticity**

The public clouds first recover H_{ID} from the received data in the *Authentication* phase. If there is no matched identity when H_{ID} is searched for in the identity list $List - H_{ID}$, the integrity check indicates that the identity of the data source (the IoT device) is invalid or that the data has been corrupted. Since H_{ID} is recovered from C_4, and C_4 is generated from H_{ID}, C_1, C_2, C_3, H_{ID} decrypted from $C_4 = H_{ID} \oplus H_3(r_2 B, C_1, C_2, C_3)$ would not be found in the list $List - H_{ID}$ if H_{ID}, C_1, C_2, or C_3 is forged in transmission. Therefore, the calculation dependency between C_4 and (H_{ID}, C_1, C_2, C_3) can ensure the invalid transmission can be found and is blocked in *Authentication*.

• **Integrity**

In the proposed scheme SOKD, the data integrity means to ensure that the transmitted data can be tamper-proof in the data transmission from public clouds to the KGC. Here, two potential attacks are discussed during the data transmission.

Firstly, the encrypted data $C_3 = AES_{H_{ID}}(H_M || M) \oplus H_2(r_1 A)$ or the identity H_{ID} from the *Authentication* phase is manipulated by the attacker during data transmission.

According to the *Decrypt* phase, C_3 can be decrypted with the AES decryption

algorithm to retrieve the plaintext M and H_M. However, if C_3 or H_{ID} (the key for AES decryption) is changed, the AES decryption algorithm cannot output the correct $H_M||M$. This is because AES encryption and decryption are symmetric, which means even if there is only one incorrect bit in C_3 or H_{ID}, the encryption or decryption results can be wrong. In this situation, the hash validation in the *Decrypt* phase will fail, i.e., $H_1(M)\neq H_M$ holds because of the incorrect M and H_M. Therefore, I can state that SOKD can take advantage of the integrity validation to block the manipulated data by the attacker between the untrusted public clouds and the KGC. Note that the data integrity in the *KeyDistribute* phase can be analysed similarly as above.

Secondly, the attacker can control the untrusted public clouds and the identity list $List - H_{ID}$ has been disclosed.

Since the public clouds have been controlled, the attacker can recover the IoT device's identity H_{ID} from the received ciphertext $C = (C_1, C_2, C_3, C_4)$ in the *Authentication* phase. Furthermore, the attacker can also substitute a forged invalid identity or another valid identity in the identity list $List - H_{ID}$ for H_{ID}. On the other hand, it is possible for the attacker to modify the ciphertext $C_3 \in C$ directly under this situation. However, the probability for the attacker to decrypt C_3 is still negligible based upon the security analysis in Section V.B and V.D. Regardless of the validity of the selected identity H^*_{ID} and the forged C^*_3 by the attacker, when the KGC receives the encrypted data $C'^* = (H^*_{ID}, C_1, C_2, C^*_3)$, the KGC can use the algorithm *Decrypt* to validate the integrity of the decrypted data M^* and H^*_M from C^* to detect falsification because of the analysis illustrated in a). Hence, SOKD can ensure data integrity and avoid the plain data M to be leaked when public clouds are manipulated by the attacker.

Overall, according to the above security analysis, when attacks occur during the data transmission from the IoT device to the KGC via the untrusted public clouds, SOKD can provide sufficient safeguards to ensure the confidentiality and integrity of the transmitted private data.

4.4.2.2 GAA-FQ

• **Confidentiality**

The confidentiality I focus on is between the user layer and the agent layer. There are two potential attacks may occur in the communications between the user layer and the agent layer.

(i) The user tries to access the data but without the corresponding access permission. For example, the user requests the data for several attributes $S_1 = \{A_1, A_2, ..., A_n\}$, however, this user only has the partial access tokens $\{AT_1, AT_2, ..., AT_{n-1}\}$ for $\{A_1, A_2, ..., A_{n-1}\}$. Therefore, the user forges the access token $AT_n^* \in \mathbb{Z}_q$ for the attribute A_n then constructs the fake $S_2^* = \{AT_1, AT_2, ..., AT_{n-1}, AT_n^*\}$ to calculate the hash value $H_T^* = H(S_2^*) = H(AT_1, AT_2, ..., AT_{n-1}, AT_n^*)$. Finally, the user sends the query $Q^* = (S_1, H_T^*)$ to the agent layer.

For the agent layer, the true token sequence queried from the storage layer is $S_2' = \{AT_1', AT_2', ..., AT_{n-1}', AT_n'\}$. Although $\forall i \in \{1..n-1\}, AT_i = AT_i'$ holds, the user only has the advantage $Pr[AT_n' = AT_n^*] = \frac{1}{|\mathbb{Z}_q|}$ to satisfy the condition $AT_n' = AT_n^*$, where $|\mathbb{Z}_q|$ represents the number of all the elements in $\mathbb{Z}_q$. Note that $\mathbb{Z}_q$ is a large discrete space as q is a big prime. Thus, $|\mathbb{Z}_q|$ is big enough to keep the user's advantage $Pr[AT_n' = AT_n^*] = \frac{1}{|\mathbb{Z}_q|}$ is negligible. Also, the probability of $S_2^* = S_2'$ is negligible as well. As a result, the user cannot pass the *Authorise* phase (see Section IV.A.3) since $Pr[H_T^* = H_T'] = Pr[S_2^* = S_2'] = Pr[AT_n' = AT_n^*] = \frac{1}{|\mathbb{Z}_q|}$ is negligible based upon the above analysis.

(ii) The communications between the user layer and the agent layer are eavesdropped on by the attacker. The attacker can obtain all the data from the communications between the user layer and the agent layer. Based upon the algorithms in the scheme, the authorisation depends on two hash values H_T in Q and H_M in C. Therefore, the communication data includes the query $Q = (S_1, H_T)$ and the response (C, P).

If the attacker modifies the query Q, it cannot pass through the *Authorise* phase in

the scheme based upon the security analysis (i). On the other hand, the attacker cannot retrieve the plain data M only with Q and (C, P) because of the following two reasons. Firstly, the attacker cannot recover the correct sequence of access tokens from the hash value H_T since the secure cryptographic hash function is a one-way function. Thus, the attacker cannot determine the correct coefficients of the polynomial $g(x)$. Secondly, the proper polynomial including the point P cannot be determined since the only one point P could be on an infinite number of polynomials. As a result, the attacker cannot carry on the *SSharing.Reconstruction* or recover the key $k = a_0 \in \mathbb{Z}_q$ to decrypt ciphertext C without the correct $g(x)$.

• **Integrity**

If the attacker eavesdrops upon the communications between the user layer and the agent layer, the attacker can challenge the integrity with the intercepted data (C, P) through three methods.

• *Forged* C^*: If the attacker changes C to C^*, the user can still recover the correct key k to decrypt C^*. However, based upon the AES algorithm, the decrypted $AES'_k(C^*) = (M^*, H_{M^*})$ is changed and the attacker cannot control M^* and H_{M^*} to make $H(M^*) = H_{M^*}$ hold via tampering C^*.

• *Forged* P^*: If the attacker replaces the point P to a fake point P^*, the user can only recover an incorrect key k^* with *SSharing.Reconstruction* to decrypt C. Furthermore, based upon the AES algorithm, the decrypted $AES'_{k^*}(C) = (M^*, H_{M^*})$ is changed and the attacker cannot control M^* and H_{M^*} to make $H(M^*) = H_{M^*}$ hold via tampering k^*.

• *Forged* (C^*, P^*): The integrity analysis for this situation is a combination of the former two situations. According to the analysis for *Forged* C^* and *Forged* P^*, I conclude that the attacker cannot control M^* or H_{M^*} to make $H(M^*) = H_{M^*}$ hold via tampering C^* or P^*. Hence, step 4 of the *Decrypt* phase outputs $\perp$ to indicate the failed integrity check for the three attack methods.

4.4.2.3 Formal verification

In this section, I follow the same procedure as in Section 3.5.2.3 to conduct formal verification for the proposed schemes SOKD and GAA-FQ using Casper/FDR. In the verification, for the scheme SOKD, the IoT device, KGC and untrusted clouds are represented by three roles Alice, Bob, and Server, respectively. Then, for the scheme GAA-FQ, the user layer, agent layer and storage layer are represented by Alice, Server and Bob, respectively. The results are demonstrated in Figure 4.10 and Figure 4.11. It is evident that the proposed schemes SOKD and GAA-FQ are sufficiently secure to preserve the confidentiality and integrity of the transmitted data.

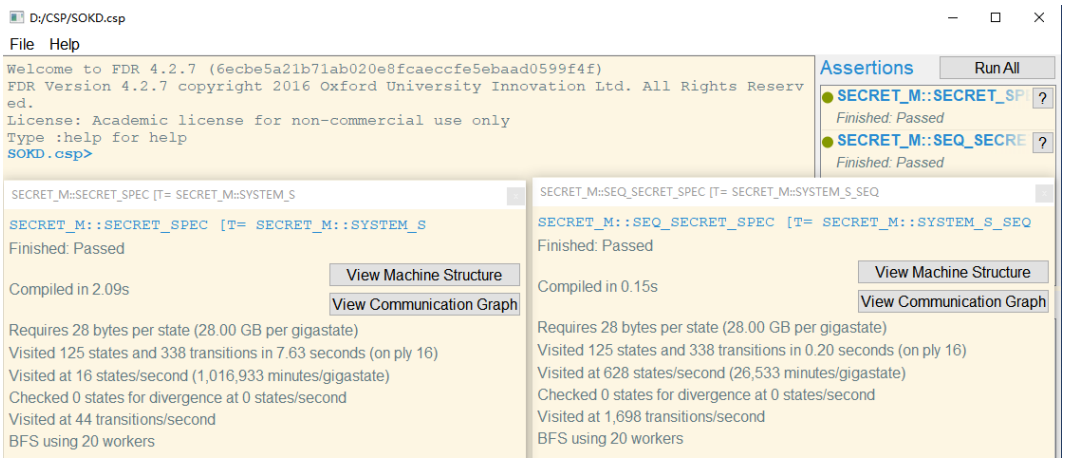

Figure 4.10: The formal verification results of SOKD with using Casper/FDR.

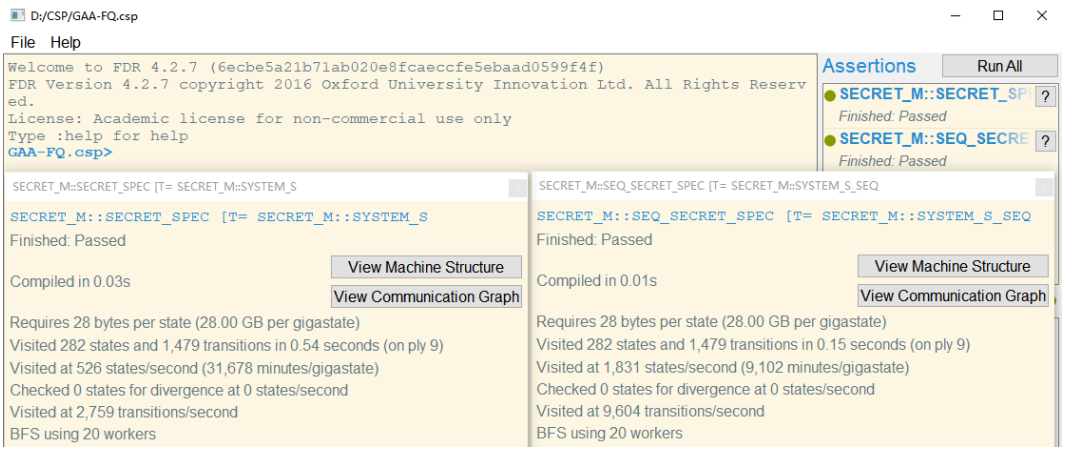

Figure 4.11: The formal verification results of GAA-FQ with using Casper/FDR.

Overall, when attacks happen to the communication between the user layer and the agent layer, the proposed scheme GAA-FQ can offer sufficient security to ensure that both the confidentiality of the transported data and the data integrity are preserved based upon the security analysis and formal security verification.

4.5 Summary

In this chapter, I present two novel privacy-preserving schemes for accessing blockchain-enabled EMRs. The first scheme is Semi-Outsourcing Key Distribution (SOKD) for use in key (access tokens) distribution to IoT devices. Compared with original work constructed with PKI and trusted clouds for authentication before key distribution, the use of trusted private clouds with a high construction expense is not needed to deploy SOKD. The authentication and key distribution are divided (semi-outsourcing) to adapt to a more general cloud architecture to distribute the access tokens to the corresponding IoT device. Meanwhile, SOKD is more suitable to be deployed in resource-constrained IoT devices because it can lower the network throughput and time cost in data transmission based upon the experiments. Therefore, SOKD is more economical and practical for use by small and medium-sized enterprises and organisations that cannot afford the cost for constructing large-scale trusted clouds but instead tend to take advantage of untrusted public clouds as part of their Information Communications Technology (ICT) service infrastructure.

The second scheme is granular access authorisation supporting flexible queries (GAA-FQ), which is the first authorisation architecture with flexible granularity for accessing blockchain-oriented EMRs in eHealth. Compared with the existing work, e.g., ESPAC, GAA-FQ does not require PKI to authorise the access or encrypt/decrypt the queried EMRs after the key distribution by the scheme SOKD. The proposed access authorisation scheme combined with three exemplar types of queries achieves a finer granular control when authorising different queries. As a result, a blockchain-based EMR can respond to a requester without leaking unauthorised private data efficiently, especially for resource-

constrained IoT devices in eHealth.

Chapter 5

Challenge-Response Assisted Anonymous Authorisation over Permissioned Blockchains

In this chapter, a challenge-response based authorisation scheme for permissioned blockchain networks named Challenge-Response Assisted Access Authorisation (CRA^3) is presented to protect users' credentials during authorisation. In CRA^3, the pre-selected nodes in the consensus network of permissioned blockchain do not require users' credentials to authorise data access requests (inspired by ZKP [177], see Section 2.1.2.5) to prevent privacy leakage when these nodes are compromised or manipulated by attackers.

5.1 Motivation to avoid exposing users' credentials to untrusted nodes in a consensus network when authorising data access

Permissioned blockchains can be applied for sharing data among permitted users to authorise the data access requests, which is a desirable blockchain structure for sharing

EMRs among doctors, patients, pathologists, medical staffs, and so on [178, 179]. A consensus network constructed using pre-selected nodes can verify a data requester's credentials to determine if he or she have the correct permissions to access the queried data.

In current studies, pre-selected consensus nodes are normally assumed to be trusted in order to authorise the data access and to transport users' private data in a consensus network [180]. However, one important issue that has not been considered is that users' private data can be utilised by the pre-selected nodes in the consensus network during authorisation, since these nodes may be manipulated by the attacker to become untrusted (malicious). To be specific, in many current designs of authorisation for permissioned blockchains [64, 181], the nodes in a consensus network require the use of many users' private information (e.g., credentials and personal information) to authorise their data access requests. In this condition, "untrusted" means that when some nodes are compromised by an attacker, the users' private information can be disclosed to an attacker without any barriers.

Furthermore, if pre-selected nodes in the consensus network are malicious, such nodes can exploit a user's private data and can analyse his or her behaviour without such a user's right to know, and this violates their privacy. For example, when a permissioned blockchain network is applied to smart power grids or smart charging, the behaviour of the registered user can be analysed based upon the uploading time and length of his or her electric bills if the nodes are malicious or compromised in the consensus network [182, 183]. Similarly, in an eHealth scenario, a patient's private information could be leaked with respect to the above situation. Therefore, the private data exposed to the consensus network should be tightly restricted.

5.2 System model

In this section, the structure of permissioned blockchain and the scheme model are first introduced. Then, the scheme definitions of CRA^3 are described under the scheme model.

5.2.1 Permissioned blockchain

The network structure of a permissioned blockchain shown in Figure 5.1 is the same as that of a public blockchain. All the nodes are anonymous in the network. However, the data (ledger) in each node are private, which means the data access between the two nodes should be authorised to ensure one node has the permissions to access the data in another node.

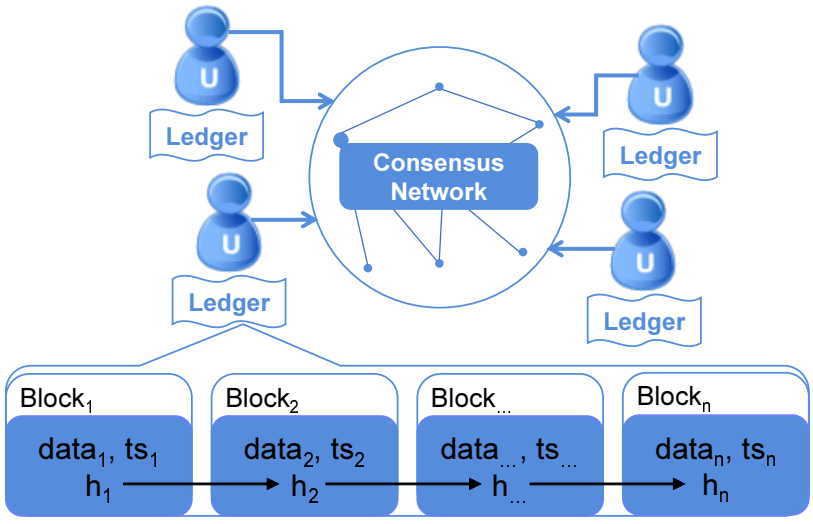

Figure 5.1: The structure of the permissioned blockchain network.

5.2.2 Scheme model

The proposed scheme (access) model is demonstrated in Figure 5.2 with three entities: two users (nodes) and the consensus network (constructed by pre-selected nodes in the permissioned blockchain network), denoted by U_A (data requester), U_B (data provider) and CN_{pm}, respectively. Since all the nodes in the network are anonymous (unknown identities), one node cannot trust another node in the permissioned blockchain network. Therefore, the consensus network CN_{pm} is needed as a mediator to finish the validations

in the challenge-response phase. If U_A has the correct permissions to access his/her requested data, U_B establishes a secret channel with U_A to transport the encrypted data after the authorisation.

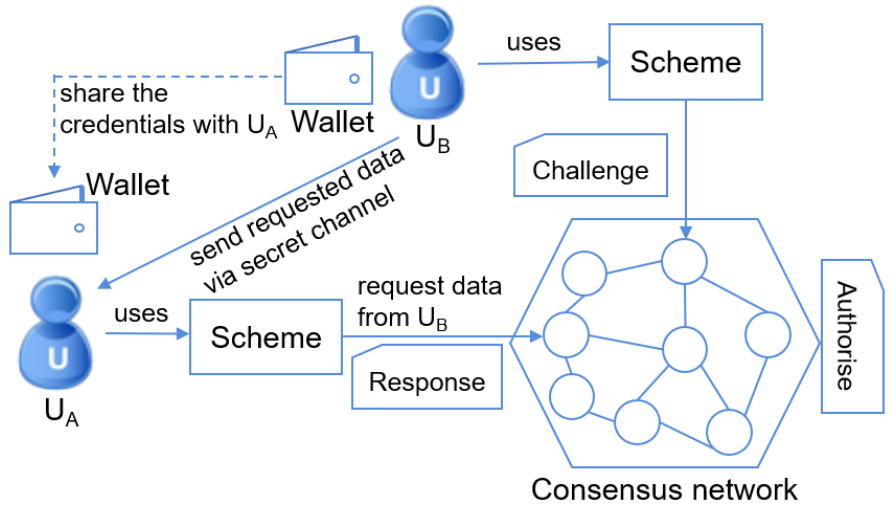

Figure 5.2: The scheme model of the proposed Challenge-Response Assisted Anonymous Authorisation (CRA^3) scheme.

Meanwhile, the private ledger of each node is a blockchain structure. Each block contains data, a cryptographic hash value (h) and a timestamp (ts) in the blockchain [7]. The hash value for establishing the link between two blocks is generated by the following rules:

$$h_i = \begin{cases} Hash(data_1||ts_1) & , \ i = 1 \\ Hash(h_{i-1}||data_i||ts_i), & i = 2...n \end{cases}$$

5.2.3 Scheme definitions

In this section, I introduce the credentials used in the authorisation and define all of the seven algorithms that comprise the proposed CRA^3 scheme.

• *Credentials for authorisation*: The credentials used for authorisation are the identity attributes. For instance, in the hospital scenario, if a doctor wants to request a patient's medical records, the identity attributes can be the patient's name, age, medical record number, social number, and so on. It is assumed that in reality, the patient has shared

the identity attributes and a unique reference number to identify the needed identity attributes for the doctor's data request before inquiring about the data. The unique reference number (key) and the identity attributes (values) are denoted by Rn and the sequence $AT^v = \{AT_1^v, AT_2^v, ..., AT_n^v\}$, respectively.

- *Setup (λ)*: This algorithm takes the security parameter λ and generates the public parameter pp.
- *Request (pp)*: U_A uses the algorithm to send a data access request Q to U_B via CN_{pm}.
- *Challenge (pp, Q)*: U_B constructs and sends the challenge Ch to U_A with the identity attributes sequence AT^v then sends the correct answer (response) Ch' of the challenge Ch to CN_{pm}.
- *Response (pp, Ch)*: U_A uses its own sequence AT'^v to calculate and send the response Re to the CN_{pm}.
- *Authorise (Ch',Re)*: CN_{pm} validates the correctness of the response Re based upon the correct response Ch'.
- *Encrypt (pp, M, AT^v)*: U_B uses this algorithm to encrypt the requested data M then return the ciphertext C and a point P (for decryption use) to U_A.
- *Decrypt (pp, C,P,AT'^v)*: U_A decrypts the encrypted data C to retrieve the requested data M with the given point P.

5.3 Problem statement and algorithm description

In this section, I first describe the targeted security problem I try to address. Then, seven algorithms are illustrated to construct the CRA^3 scheme in the second part.

5.3.1 Problem statement

Since the nodes in CN_{pm} are assumed to be untrusted, one untrusted node can output incorrect results of authorisation and collect the user's private data from the communication in the challenge-response phase. If a node outputs incorrect authorisation results, the consensus network can punish this dishonest node according to the applied consensus mechanism (e.g., Byzantine fault tolerance). On the other hand, if a malicious node collects the user's data from the challenge-response communication and then attempts to reveal the user's credentials (or other private information), the proposed scheme should prevent the private data leakage. Hence, the purpose of the proposed scheme is to authorise data access without involving the users' credentials ($U_{id} = \{U_1, U_2, ..., U_n\}$) whilst ensuring the transported data is confidential, i.e., $\forall M \in \{0,1\}^*, C = f(M)$, and any probabilistic polynomial-time algorithm $\mathcal{A}$ computes M or U_{id} with its advantage $Adv_{\mathcal{A}}^{CN_{pri}} = Pr[c = M \vee c \subseteq U_{id} | c = \mathcal{A}(C, D_{Ch}, D_{Re})] < \varepsilon$, where M is plaintext data, C is encrypted data that transmitted between U_A and U_B, D_{Ch} and D_{Re} denote the data used in the processes *Challenge* and *Response*, respectively, and ε represents a negligible probability.

5.3.2 Algorithm description of CRA^3

In this section, I illustrate the proposed CRA^3 scheme (Challenge-Response Assisted Anonymous Authorisation) with the seven algorithms defined in Section 5.2.3, including Setup, Request, Challenge, Response, Authorise, Encrypt, and Decrypt. In CRA^3, *AES* (Advanced Encryption Standard [161]) is used to encrypt the requested data and Lagrange interpolating polynomial is utilised to construct challenge-response authorisation and protect the encrypting/decrypting key of *AES* (see Section 2.1.2.4). Furthermore, the concept of ZKP is applied to restrict the information leakage in the authorisation (see Section 2.1.2.5) [177]. Note that the correctness of the CRA^3 scheme is presented in Appendix C.1.

- **Setup (λ)**

This procedure outputs public parameters pp with the security parameter λ using the following steps.

1. Generate a big prime q ($q > 2^{\lambda}$);
2. Select one secure cryptographic hash function $H : \{0,1\}^{*} \rightarrow \{0,1\}^{\lambda}$;
3. Select a symmetric encryption algorithm, e.g., AES (Advanced Encryption Standard);
4. Output the public parameters $pp = (q, H, AES)$.

- **Request (pp)**

The user U_A (data inquirer) prepares the query Q via the following steps.

1. Decide on the data to be requested. Note that U_A should have the corresponding identity attributes (a sequence, AT'^{v}) and the unique reference number (R_n) that is shared by U_B. For illustrating the remaining parts of the proposed scheme, I assume the requested data is in one block B_{id};
2. Prepare the unique reference number Rn then send the request $Q = (B_{id}, Rn)$ to U_B through CN_{pm}. Note that users can establish secure connections with CN_{pm}.

- **Challenge (pp, Q)**

U_B generates the challenge Ch based upon the request Q from U_A via the following steps.

1. Prepare the sequence of the identity attributes (values): $AT^{v} = \{AT_1^{v}, AT_2^{v}, ..., AT_n^{v}\}$ based upon the unique reference number $Rn \in Q$;
2. Calculate the hash value of each element in the sequence AT^{v} to get the sequence $AH^{v} = \{H(AT_1^{v}), H(AT_2^{v}), ..., H(AT_n^{v})\}$;
3. Construct a polynomial $f(x) = H(AT_1^{v}) + H(AT_2^{v})x + ... + H(AT_n^{v})x^{n-1}$ $(mod\ q)$,

then pick n random points on the polynomial $f(x)$ as a set:

$$P = \{(x_i, y_i) | (x_i, y_i) \in f(x) \wedge i = 1...n\};$$

4. Construct two sequences P_x and P_y of all the x_i and all the y_i in P: $P_x = \{x_i | x_i \in f(x) \wedge (x_i, f(x_i)) \in P \wedge i = 1...n\}$ and $P_y = \{y_i | y_i = f(x_i) \wedge x_i \in P_x \wedge i = 1...n\}$;

5. Calculate the hash value PH_y of the sequence P_y, where $PH_y = H(y_1, y_2, ..., y_n),\ y_1, y_2, ..., y_n \in P_y$;

6. Send the challenge $Ch = P_x$ to U_A. Note that CN_{pm} should keep the correct response $Ch' = (PH_y)$ to execute the following *Authorise* phase.

- **Response** (pp, Ch)

U_A generates the response Re to the challenge P_x from U_B via the following steps.

1. Prepare the sequence of the identity attributes $AT'^v = \{AT_1'^v, AT_2'^v, ..., AT_n'^v\}$ (shared by U_B) based upon $R_n \in Q$;

2. Construct a new polynomial $g(x) = H(AT_1'^v) + H(AT_2'^v)x + ... + H(AT_n'^v)x^{n-1}\ (mod\ q)$;

3. Take $P_x \in Ch$ to calculate the sequence $P'_y = \{y'_i | y'_i = g(x_i) \wedge x_i \in P_x \wedge i = 1...n\}$ and then hash the sequence P'_y: $PH'_y = H(y'_1, y'_2, ..., y'_n),\ y'_1, y'_2, ..., y'_n \in P'_y$;

4. Send the response $Re = (PH'_y)$ to the consensus network CN_{pm}.

- **Authorise** (Ch', Re)

The consensus network CN_{pm} validates the two hash values in Ch' and Re. If $PH_y(\in Ch') = PH'_y(\in Re)$ holds (consensus check point), it means that the user U_A can be authorised to access the requested data B_{id} and the next phases are conducted; otherwise, the agent layer should deny the access request from U_A.

- **Encrypt** (pp, Q, AT^v)

U_B encrypts the requested data via the following steps.

1. Acquire the requested data M based upon $B_{id} \in Q$ from U_A and then calculate the hash value H_M of the data M: $H_M = H(M)$;

2. Generate a secure key $k \in \mathbb{Z}_q$ for the symmetric encryption;

3. Use AES to encrypt M with key k to get the ciphertext $C = AES_k(M, H_M)$. For decrypting $AES_k(M, H_M)$ to recover the plain data M, AES'_k is defined as the decryption process: $M = AES'_k(C = AES_k(M, H_M))$;

4. Construct a polynomial $f^*(x)$ of degree n with k and AT^v: $f^*(x) = k + H(AT_1^v)x + H(AT_2^v)x^2 + ... + H(AT_n^v)x^n \ (mod\ q)$;

5. Generate a random integer $x_p \in \mathbb{Z}_q$ and calculate a point $P(x_p, y_p = f^*(x_p))$;

6. Return (C, P) to U_A through a secret channel.

• **Decrypt** (pp, C, P, AT'^v)

U_A can decrypt the ciphertext C after passing the *Authorise* phase via the following steps.

1. Use the sequence AT'^v organised in the former *Response* phase to construct a polynomial $g^*(x)$: $g^*(x) = a_0 + H(AT_1'^v)x + H(AT_2'^v)x^2 + ... + H(AT_n'^v)x^n \ (mod\ q)$. Note that a_0 is an unknown coefficient;

2. Follow the Lagrange interpolation polynomial to reconstruct the polynomial $g^*(x)$ fully, and then recover the key $k = g(0) = a_0 \in \mathbb{Z}_q$ for AES decryption with the point $P(x_p, y_p)$: $k = y_p - H(AT_1'^v)x_p - H(AT_2'^v)x_p^2 - ... - H(AT_n'^v)x_p^n \ (mod\ q)$;

3. Decrypt C to retrieve the plaintext $(M, H_M) = AES'_k(C) = AES'_k(AES_k(M, H_M))$;

4. If $H(M) = H_M$ holds, this algorithm outputs M; otherwise, it outputs $\perp$.

5.4 Analysis of performance and security

5.4.1 Simulation results

The performance simulations and results are illustrated and discussed in this section. Two Raspberry Pi 2s with Wi-Fi (as the mobile devices of the users U_A and U_B) and one conventional computer with an Intel i5 processor running at 3.30 GHz (as a node of the consensus network in the permissioned blockchain) were used to perform the simulations. The local computational time efficiency for executing CRA^3 was evaluated with respect to the time cost for transmitting encrypted data over Wi-Fi and the transaction fee (gas) of the consensus node in the simulations. Since there is yet no clear best practice to be used as a baseline for comparison, I selected an authorisation scheme for blockchain-based storage named *Decentralizing Privacy* (DP) [181] as the baseline. The authorisation supported by a trusted third party (TTP) in DP is policy-based but not anonymous, since the TTP knows the users' identities. However, the designed authorisation in CRA^3 is attribute-based and anonymous. Note that all the implemented experiments used the equivalent cryptographic security level (128-bit) [21], and that the transaction fee (gas) was calculated based upon the bytecodes generated by Ethereum Virtual Machine (EVM) [184] with PoA (Proof of Authority) [185] as the consensus mechanism.

First, the number of the attributes used for authorisation was varied from two to 10 in CRA^3 (respective of policies in DP) to compare the time taken for local computation including authorisation, encryption, and decryption algorithms in the two schemes implemented on a conventional computer. The averaged results over 10 runs are shown in Figure 5.3. In the authorisation phase (Figure 5.3a), the time cost in both schemes increased with a similar trend when the number of attributes used was small. When the number of attributes used rose up to 10, the CRA^3 scheme needed 25% more time to authorise the access when compared with the DP scheme. For the encryption and decryption phases, the time cost for the DP scheme stays stable whilst the time cost of the CRA^3 scheme increased slowly, increasing with the number of attributes. On average, the time cost of the CRA^3 scheme was 55% lower than that of the DP scheme;

see Figure 5.3 (b) and (c).

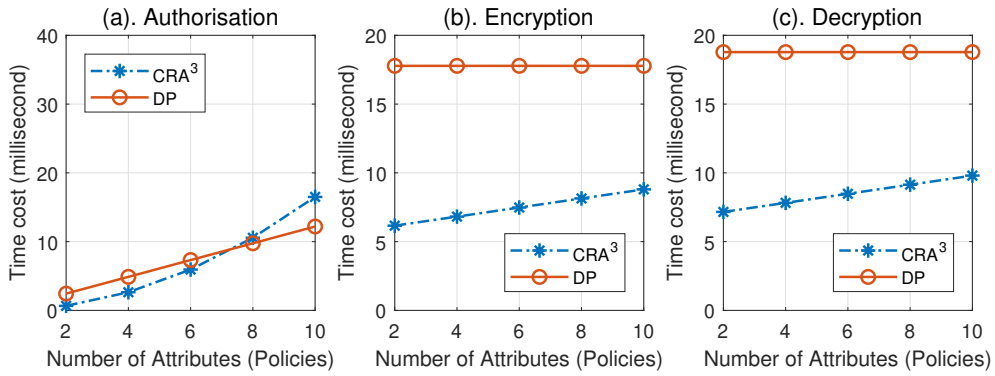

Figure 5.3: The time cost comparison of local computation on a Raspberry Pi 2 between CRA^3 and DP (*Decentralise Privacy*).

Meanwhile, I measured the time cost for transporting data between users and CN_{pm} over Wi-Fi (Figure 5.4). The data included the attributes (i.e., policies) used for authorisation, the encrypted data (128 bytes) and the keys used for decryption in the two schemes. Since CRA^3 only transmits two points in the *Authorise* phase whereas the DP scheme requires two policy lists for authorisation, the time consumption for transmitting data via Wi-Fi in CRA^3 was about 24% lower than that in the DP scheme. Furthermore, the time cost in CRA^3 had a lower growth rate when compared with the DP scheme.

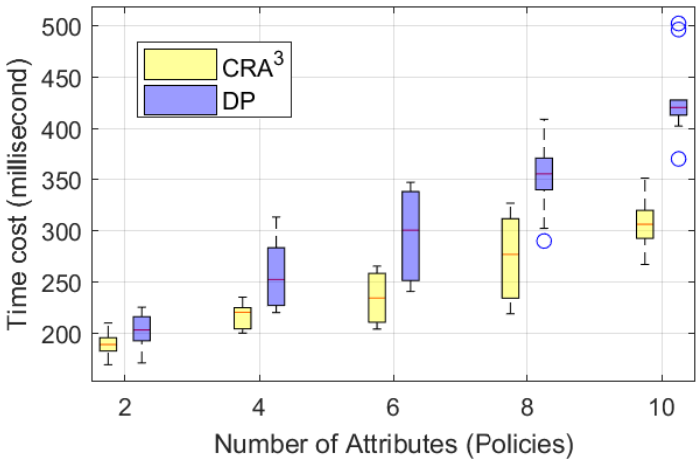

Figure 5.4: Comparison of the time costs of CRA^3 and DP for transmitting data over Wi-Fi.

Thus, I summarise the total time cost of both local computation and data transmission via Wi-Fi in Figure 5.5. The total time cost in CRA^3 was around 30% lower than that in the DP scheme. When the number of used attributes (i.e., policies) increased, the DP scheme consumed far more time than CRA^3, in total.

Finally, the transaction fee (gas) for the *Authorise* phase performed in the consensus network was evaluated in a conventional computer (Figure 5.6). While the transaction fee of CRA^3 kept stable (and was non-sensitive to the variation of used attributes), the transaction fee increased by the number of used policies in the DP scheme. This is because the DP scheme compares two policy lists in the transaction for authorisation but CRA^3 only compares two points regardless of the number of used attributes.

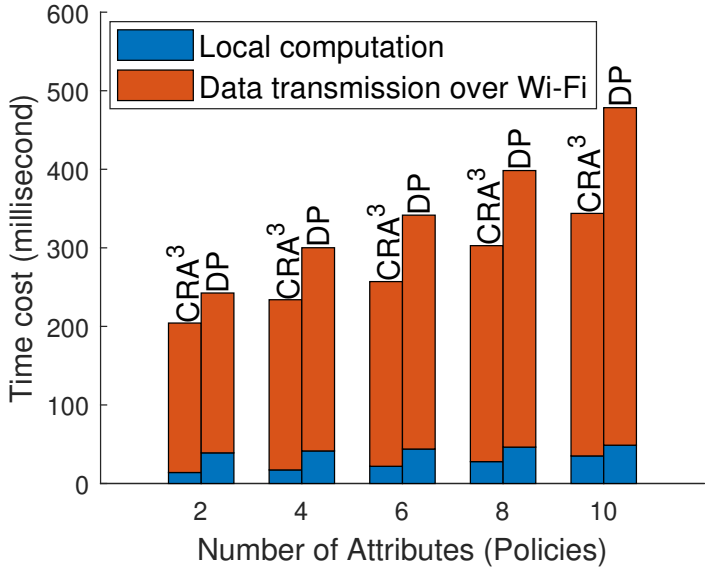

Figure 5.5: Comparison of total time cost of CRA^3 and DP.

5.4.2 Confidentiality and integrity

In this section, the confidentiality of the identity attributes and the ciphertext and the integrity of the ciphertext is considered. Additionally, the formal theoretical proof of the ciphertext confidentiality is illustrated in Appendix C.2. Note that the formal security verification is not repeated since the authorisation of CRA^3 is same with that of GAA-

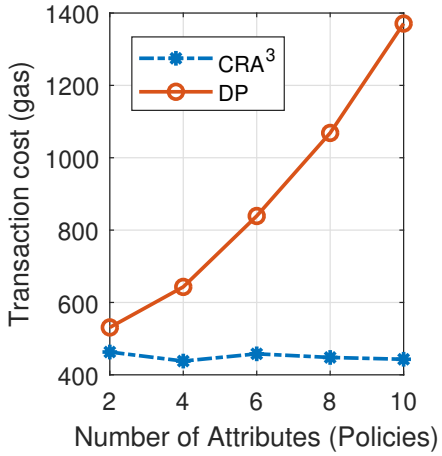

Figure 5.6: Transaction fee of CRA^3 and DP for authorisation.

FQ.

5.4.2.1 Confidentiality

There are two kinds of potential attacks that may occur in the communication between the users and the consensus network CN_{pm}.

Firstly, the EMR requester tries to access the EMRs without the corresponding identity attributes. For example, the queried EMRs require the sequence of the identity attributes $S_1 = \{AT_1'^v, AT_2'^v, ..., AT_n'^v\}$ and the corresponding sequence in the *Challenge* phase is $S_0 = \{AT_1^v, AT_2^v, ..., AT_n^v\}$. Note that $S_1 = S_0$ should hold if the EMR requester has the correct identity attributes. However, the EMR requester only has the partial identity attributes $\{AT_1'^v, AT_2'^v, ..., AT_{n-1}'^v\}$. Therefore, the requester forges the last identity attribute $AT_n'^{v*} \in \mathbb{Z}_q$ for the attribute $AT_n'^v$ to build the fake sequence $S_1^* = \{AT_1'^v, AT_2'^v, ..., AT_{n-1}'^v, AT_n'^{v*}\}$ to construct the polynomial $g^*(x) = H(AT_1'^v) + H(AT_2'^v)x + ... + H(AT_{n-1}'^v)x^{n-2} + H(AT_n'^{v*})x^{n-1}$ $(mod\ q)$. After that, the requester follows step 3 of the algorithm *Response* in CRA^3 to compute $PH_y'^*$ with P_x. Finally, the requester sends the response $Re = PH_y'^*$ to the consensus network CN_{pm}.

Although $\forall i \in \{1..n-1\}$, the element $AT_i'^v$ of S_1 is same as element $AT_i'^v$ of S_1^*, the requester only has the advantage $Pr = \frac{1}{|\mathbb{Z}_q|}$ to guess a correct $AT_n'^{v*}$ to satisfy the

condition $AT_n'^v = AT_n'^{v*}$, where $|\mathbb{Z}_q|$ represents the number of all the elements in $\mathbb{Z}_q$. Note that $\mathbb{Z}_q$ is a large discrete space as q is a big prime. Thus, $|\mathbb{Z}_q|$ is big enough to keep the advantage Pr is negligible. Meanwhile, the probability of $S_1^* = S_1$ is negligible as well. As a result, the EMR requester cannot pass the *Authorise* phase since $Pr[PH_y = PH_y'^*] = Pr[S_0 = S_1^*] = Pr[S_1 = S_1^*] = \frac{1}{|\mathbb{Z}_q|}$ is negligible based upon the above analysis, where $PH_y \in Ch'$ and $PH_y'^* \in Re$.

Secondly, the communications between users and CN_{pm} are eavesdropped upon by the attacker. The attacker can obtain all the data from the communications including the challenge-response information Ch' and Re based upon the CRA^3 scheme. Obviously, the attacker cannot recover the correct sequence of identity attributes from the hash value Ch' or Re since the secure cryptographic hash function is a one-way function. Thus, the attacker cannot determine the correct coefficients of the polynomial $f(x)$ (or $g(x)$) to continue the *Encrypt* and *Decrypt* algorithms. Even though the ciphertext C and the point P are leaked, the attacker still cannot retrieve the plain data M because one point P is not enough to determine the proper polynomial $f(x)$ (or $g(x)$) to recover the key k based upon Shamir Secret Sharing (see Section 2.1.2.4). Therefore, the CRA^3 scheme can ensure the confidentiality of the identity attributes and the ciphertext.

5.4.2.2 Integrity

In the CRA^3 scheme, the hash value $H_M = H(M)$ generated in the *Encrypt* algorithm can provide the data integrity of M. In the *Decrypt* algorithm, if the received C or P is incorrect or manipulated by the attacker in the communication between U_A and U_B, the wrong C (or P) leads to abnormal AES decryption results so that $(M, H_M) = AES'_k(C)$ are incorrect (where $C = AES_k(M, H_M)$ and k is computed from P. Therefore, the condition $H(M) = H_M$ (step 4) cannot hold, which means the data integrity check can detect an abnormal C or P to protect the data integrity of M.

5.4.3 Comparison of security features

In Table 5.1, I compare the implemented security features of different blockchain-related authorisation schemes from the state-of-the-art of related work with that of my CRA^3 scheme. It is clear that most of the compared schemes can support permissioned blockchains but CRA^3 is the only one that can support an untrusted consensus network. Meanwhile, CRA^3 can also provide authorisation, confidentiality, and integrity for data access. However, in other compared schemes, the integrity feature is only implemented by [186] and no scheme considers confidentiality. The *Decentralizing Privacy* (DP) [181] scheme requires a database as a storage media; however, the DP scheme itself cannot support confidentiality or integrity.

5.5 Summary

In this chapter, I introduced a privacy-enhanced authorisation CRA^3 scheme under a consideration of untrusted nodes occurring in a consensus network of permissioned blockchain. Unlike existing work [64, 146, 181], CRA^3 is inspired by the thought of ZKP to avoid exposing users' credentials to the untrusted nodes in the consensus network for authorising data access. By applying CRA^3 in a permissioned blockchain, users (data providers) can share private data with valid data requesters without leaking their private information. Therefore, CRA^3 can help people to safeguard their privacy and prevent potential privacy leakage (e.g., caused by attackers) in permissioned blockchains. In terms of the communication overhead, CRA^3 reduces the time cost for the communication during the authorisation since the size of the required data for authorising data access request is much smaller when compared with other methods. Furthermore, the consensus verification only relies on one equation and other computational work is executed by the data requester and receiver; hence, the consensus cost (transaction fee) is visibly cut down to save the user's cost and the computational resource of the consensus network (i.e., lower workload) simultaneously.

Table 5.1: Comparison of the security features in different blockchain-related authorisation schemes.

Scheme	**Blockchain Type**	**Consensus Network Type**	**Security Features**		
			Authorisation	**Confidentiality**	**Integrity**
[181]	Public/Permissioned	Trusted	√	×[1]	×[1]
[146]	Public	Trusted	√	×	×
[187]	Permissioned	Trusted	×	×	×
[186]	Permissioned	Trusted	√	×	√
CRA^{3*}	Permissioned	Trusted/Untrusted	√	√	√

[1] The scheme depends on the deployed database to support the mentioned security feature.
* CRA^3: the proposed scheme, Challenge-Response Assisted Access Authorisation.

Chapter 6

Conclusions and Future Work

6.1 Conclusions

In this thesis, the principal objective is to tackle the challenges of granularity control in access control when applying blockchain in EMRs to safeguard user privacy. Unlike many current high-level architectures of blockchain-enabled EMRs (and eHealth) which lack detailed access control algorithms or schemes, four concrete access control schemes including three authorisation schemes and one key distribution scheme (with authentication) constructed by lightweight cryptographic primitives are proposed to impose the privacy preservation for the users in eHealth.

The addressed access control schemes can realise fine-grained and flexible authorisation for querying blockchain-enabled EMRs, which are used to store patients' medical data. In the design, the major cryptographic components utilised are SSS and ECC to ensure the proposed schemes are time-saving (lightweight) enough to be deployed in numerous resource-constrained IoT devices in eHealth to achieve low resource consumption (in computation and communication) shown in the performance experiments. Meanwhile, the designed granularity levels can fit the blockchain structure to achieve seamless access control schemes for blockchain-enabled storage and transactions applied

in EMRs. Furthermore, to avoid exposing users' identities in a blockchain context where the participants of access control can be untrusted, protecting users' credentials during authorisation is also investigated to form an anonymous authorisation scheme with a challenge-response strategy for the permissioned blockchain.

In the performance experiments, the proposed access control schemes can reduce the time cost of both computation and communication in data access when compared with the state-of-the-art baselines. Furthermore, the theoretical validations of correctness, confidentiality and unforgeability manifest the proposed schemes can work correctly and protect the transmitted EMRs to avoid invalid decryption or forgery in communication.

6.2 Future Work

There are two potential directions to extend this work in the future including key distribution and consensus mechanism.

- **Key distribution.** Key distribution is an essential procedure to initialise a security system to ensure all the participants possess valid security keys for identity verification, encryption and so on under the same cryptographic context. Most current security applications require TTPs to issue and distribute keys, but this pattern breaks the natural anonymity of blockchain. In this thesis, I only discuss how to distribute keys to users but do not involve how to generate the corresponding keys for each user based upon its identity. Therefore, when a blockchain is applied in non-cryptocurrency fields and access control is demanded, how to keep the participants anonymous and generate valid keys for every participant should be discussed continuously in future blockchain-enabled systems with some promising cryptographic methods such as attribute-based encryption (ABE), certificateless encryption and multi-party computation (MPC) [188].

- **Consensus mechanism.** Another attempt could be utilising access control methods (e.g., SSS, IBE and ABE) to design new consensus mechanisms to replace the

resource-consuming consensus mechanisms such as PoW supported by the mining operation. Similar to PoS, the attributes a participant possesses can be a criterion to determine the ratio of voting and bonus. For example, Shamir secret sharing (SSS) may be transformed to a consensus mechanism so that if the number of the participants that can pass the access control reaches a threshold, it implies all the participants consent to the current proposal (transaction).

Bibliography

[1] Market Data Forecast, "Global eHealth Market Size, Share, Trends and Growth Analysis Report — 2020 to 2025," Accessed: Feb. 2020. [Online]. Available: https://www.marketdataforecast.com/market-reports/e-Health-market, 2020.

[2] S. B. Baker, W. Xiang, and I. Atkinson, "Internet of things for smart healthcare: Technologies, challenges, and opportunities," *IEEE Access*, vol. 5, pp. 26 521–26 544, 2017.

[3] B. Farahani, F. Firouzi, V. Chang, M. Badaroglu, N. Constant, and K. Mankodiya, "Towards fog-driven iot ehealth: Promises and challenges of iot in medicine and healthcare," *Future Generation Computer Systems*, vol. 78, pp. 659–676, 2018.

[4] D. Lu, R. Han, Y. Shen, X. Dong, J. Ma, X. Du, and M. Guizani, "xTSeH: A Trusted Platform Module Sharing Scheme towards Smart IoT-eHealth Devices," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 2, pp. 370–383, 2020.

[5] M. A. Khan and K. Salah, "Iot security: Review, blockchain solutions, and open challenges," *Future Generation Computer Systems*, vol. 82, pp. 395–411, 2018.

[6] A. Dubovitskaya, Z. Xu, S. Ryu, M. Schumacher, and F. Wang, "How blockchain could empower ehealth: An application for radiation oncology," in *VLDB Workshop on Data Management and Analytics for Medicine and Healthcare*. Springer, 2017, pp. 3–6.

[7] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," bitcoin.org, Bitcoin, Tech. Rep., 2008.

[8] D. C. Nguyen, P. N. Pathirana, M. Ding, and A. Seneviratne, "Blockchain for secure ehrs sharing of mobile cloud based e-health systems," *IEEE access*, vol. 7, pp. 66 792–66 806, 2019.

[9] M. A. Uddin, A. Stranieri, I. Gondal, and V. Balasubramanian, "Blockchain leveraged decentralized iot ehealth framework," *Internet of Things*, vol. 9, p. 100159, 2020.

[10] S. Cao, G. Zhang, P. Liu, X. Zhang, and F. Neri, "Cloud-assisted secure eHealth systems for tamper-proofing EHR via blockchain," *Information Sciences*, vol. 485, pp. 427–440, 2019.

[11] W. Stallings, L. Brown, M. D. Bauer, and A. K. Bhattacharjee, *Computer security: principles and practice*. Pearson Education Upper Saddle River, NJ, USA, 2012.

[12] A. J. Menezes, P. C. Van Oorschot, and S. A. Vanstone, *Handbook of applied cryptography*. CRC press, Boca Raton, FL, USA, 2018.

[13] Wikipedia, "Public-key cryptography." Accessed: Oct. 22, 2020. [Online]. Available: https://en.wikipedia.org/wiki/Public-key_cryptography.

[14] T. Kleinjung, K. Aoki, J. Franke, A. K. Lenstra, E. Thomé, J. W. Bos, P. Gaudry, A. Kruppa, P. L. Montgomery, D. A. Osvik *et al.*, "Factorization of a 768-bit rsa modulus," in *Annual Cryptology Conference*. Springer, 2010, pp. 333–350.

[15] N. Koblitz, "Elliptic curve cryptosystems," *Mathematics of computation*, vol. 48, no. 177, pp. 203–209, 1987.

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

[17] A. Joux and K. Nguyen, "Separating decision diffie–hellman from computational diffie–hellman in cryptographic groups," *Journal of cryptology*, vol. 16, no. 4, pp. 239–247, 2003.

[18] D. Lehmer, "On euler's totient function," *Bulletin of the American Mathematical Society*, vol. 38, no. 10, pp. 745–751, 1932.

[19] Wikipedia, "Euler's theorem." Accessed: Oct. 01, 2020. [Online]. Available: https://en.wikipedia.org/wiki/Euler%27s_theorem.

[20] A. S. Wander, N. Gura, H. Eberle, V. Gupta, and S. C. Shantz, "Energy analysis of

public-key cryptography for wireless sensor networks," in *Third IEEE international conference on pervasive computing and communications*. IEEE, 2005, pp. 324–328.

[21] E. Barker, W. Barker, W. Burr, W. Polk, and M. Smid, "Recommendation for key management part 1: General (revision 5)," NIST Special Publication (NIST SP), NIST, Gaithersburg, MD, USA, Tech. Rep. NIST.SP.800-57pt1r5, 2020.

[22] J. Doerner, Y. Kondi, E. Lee, and A. Shelat, "Secure two-party threshold ecdsa from ecdsa assumptions," in *2018 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2018, pp. 980–997.

[23] P. Szczechowiak, L. B. Oliveira, M. Scott, M. Collier, and R. Dahab, "Nanoecc: Testing the limits of elliptic curve cryptography in sensor networks," in *European conference on Wireless Sensor Networks*. Springer, 2008, pp. 305–320.

[24] L. Nguyen and R. Safavi-Naini, "Efficient and provably secure trapdoor-free group signature schemes from bilinear pairings," in *International Conference on the Theory and Application of Cryptology and Information Security*. Springer, 2004, pp. 372–386.

[25] D. He, N. Kumar, S. Zeadally, A. Vinel, and L. T. Yang, "Efficient and privacy-preserving data aggregation scheme for smart grid against internal adversaries," *IEEE Transactions on Smart Grid*, vol. 8, no. 5, pp. 2411–2419, 2017.

[26] D. Boneh, B. Lynn, and H. Shacham, "Short signatures from the weil pairing," in *International conference on the theory and application of cryptology and information security*. Springer, 2001, pp. 514–532.

[27] K. Lauter, "The advantages of elliptic curve cryptography for wireless security," *IEEE Wireless communications*, vol. 11, no. 1, pp. 62–67, 2004.

[28] H. Debiao, C. Jianhua, and H. Jin, "An id-based proxy signature schemes without bilinear pairings," *Annals of telecommunications-annales des télécommunications*, vol. 66, no. 11-12, pp. 657–662, 2011.

[29] G. De Meulenaer, F. Gosset, F.-X. Standaert, and O. Pereira, "On the energy cost of communication and cryptography in wireless sensor networks," in *2008 IEEE International Conference on Wireless and Mobile Computing, Networking*

and Communications. IEEE, 2008, pp. 580–585.

[30] B. Lynn, "Pairing based cryptography-benchmarks." Accessed: Jun. 14, 2013. [Online]. Available: http://crypto.stanford.edu/pbc/times.html.

[31] M. Scott, "MIRACL - Multiprecision Integer and Rational Arithmetic C/C++ Library." Accessed: Nov. 06, 2012. [Online]. Available: https://github.com/miracl/MIRACL.

[32] N. R. Potlapally, S. Ravi, A. Raghunathan, and N. K. Jha, "A study of the energy consumption characteristics of cryptographic algorithms and security protocols," *IEEE Transactions on mobile computing*, vol. 5, no. 2, pp. 128–143, 2005.

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

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

[35] J. Herranz, F. Laguillaumie, and C. Ràfols, "Constant size ciphertexts in threshold attribute-based encryption," in *International Workshop on Public Key Cryptography*. Springer, 2010, pp. 19–34.

[36] J. Liu, X. Huang, and J. K. Liu, "Secure sharing of personal health records in cloud computing: ciphertext-policy attribute-based signcryption," *Future Generation Computer Systems*, vol. 52, pp. 67–76, 2015.

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

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

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

[40] J. Han, W. Susilo, Y. Mu, and J. Yan, "Privacy-preserving decentralized key-

policy attribute-based encryption," *IEEE transactions on parallel and distributed systems*, vol. 23, no. 11, pp. 2150–2162, 2012.

[41] J. Lai, R. H. Deng, Y. Li, and J. Weng, "Fully secure key-policy attribute-based encryption with constant-size ciphertexts and fast decryption," in *Proceedings of the 9th ACM symposium on Information, computer and communications security*, 2014, pp. 239–248.

[42] J. Li, W. Yao, J. Han, Y. Zhang, and J. Shen, "User collusion avoidance cp-abe with efficient attribute revocation for cloud storage," *IEEE Systems Journal*, vol. 12, no. 2, pp. 1767–1777, 2017.

[43] K. Yang, X. Jia, K. Ren, B. Zhang, and R. Xie, "Dac-macs: Effective data access control for multiauthority cloud storage systems," *IEEE Transactions on Information Forensics and Security*, vol. 8, no. 11, pp. 1790–1801, 2013.

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

[45] S. Ma, Q. Huang, M. Zhang, and B. Yang, "Efficient public key encryption with equality test supporting flexible authorization," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 3, pp. 458–470, 2014.

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

[47] T. Ermakova and B. Fabian, "Secret sharing for health data in multi-provider clouds," in *2013 IEEE 15th Conference on Business Informatics*. IEEE, 2013, pp. 93–100.

[48] M. Naor and A. Wool, "Access control and signatures via quorum secret sharing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 9, no. 9, pp. 909–922, 1998.

[49] L. Harn and C. Lin, "Detection and identification of cheaters in (t, n) secret sharing scheme," *Designs, Codes and Cryptography*, vol. 52, no. 1, pp. 15–24, 2009.

[50] Z. Wang, M. Karpovsky, and L. Bu, "Design of reliable and secure devices realizing

shamir's secret sharing," *IEEE Transactions on Computers*, vol. 65, no. 8, pp. 2443–2455, 2015.

[51] S. Goldwasser, S. Micali, and C. Rackoff, "The knowledge complexity of interactive proof systems," *SIAM Journal on computing*, vol. 18, no. 1, pp. 186–208, 1989.

[52] J.-J. Quisquater, M. Quisquater, M. Quisquater, M. Quisquater, L. Guillou, M. A. Guillou, G. Guillou, A. Guillou, G. Guillou, and S. Guillou, "How to explain zero-knowledge protocols to your children," in *Conference on the Theory and Application of Cryptology*. Springer, 1989, pp. 628–631.

[53] A. Fiat and A. Shamir, "How to prove yourself: Practical solutions to identification and signature problems," in *Conference on the theory and application of cryptographic techniques*. Springer, 1986, pp. 186–194.

[54] G. Yang, C. H. Tan, Q. Huang, and D. S. Wong, "Probabilistic public key encryption with equality test," in *Cryptographers' Track at the RSA Conference*. Springer, 2010, pp. 119–131.

[55] R. E. Scott and M. Mars, "Principles and framework for ehealth strategy development," *Journal of medical Internet research*, vol. 15, no. 7, p. e155, 2013.

[56] Mordor Intelligence, "E-HEALTH MARKET- GROWTH, TRENDS, AND FORECASTS (2020 - 2025)," Accessed: Feb. 2020. [Online]. Available: https://www.mordorintelligence.com/industry-reports/e-health-market, 2020.

[57] J. Lloret, L. Parra, M. Taha, and J. Tomás, "An architecture and protocol for smart continuous ehealth monitoring using 5g," *Computer Networks*, vol. 129, pp. 340–351, 2017.

[58] T. Sahama, L. Simpson, and B. Lane, "Security and privacy in ehealth: Is it possible?" in *2013 IEEE 15th International Conference on e-Health Networking, Applications and Services (Healthcom 2013)*. IEEE, 2013, pp. 249–253.

[59] H. Lin, J. Shao, C. Zhang, and Y. Fang, "Cam: cloud-assisted privacy preserving mobile health monitoring," *IEEE Transactions on Information Forensics and Security*, vol. 8, no. 6, pp. 985–997, 2013.

[60] R. Mircea, G. Saplacan, G. Sebestyen, N. Todor, L. Krucz, and C. Lelutiu, "ehealth: towards a healthcare service-oriented boundary-less infrastructure," *Applied Med-*

ical Informatics., vol. 27, no. 3, pp. 1–14, 2010.

[61] B. Farahani, M. Barzegari, F. S. Aliee, and K. A. Shaik, "Towards collaborative intelligent iot ehealth: From device to fog, and cloud," *Microprocessors and Microsystems*, vol. 72, p. 102938, 2020.

[62] S. Aggarwal, R. Chaudhary, G. S. Aujla, N. Kumar, K.-K. R. Choo, and A. Y. Zomaya, "Blockchain for smart communities: Applications, challenges and opportunities," *Journal of Network and Computer Applications*, vol. 144, pp. 13 – 48, 2019.

[63] M. Pustišek and A. Kos, "Approaches to front-end iot application development for the ethereum blockchain," *Procedia Computer Science*, vol. 129, pp. 410–419, 2018.

[64] E. Androulaki, A. Barger, V. Bortnikov, C. Cachin, K. Christidis, A. De Caro, D. Enyeart, C. Ferris, G. Laventman, Y. Manevich *et al.*, "Hyperledger fabric: a distributed operating system for permissioned blockchains," in *Proceedings of the thirteenth EuroSys conference*. ACM, 2018, pp. 1–15.

[65] J. Garzik, "Public versus private blockchains," *BitFury Group, San Francisco, USA, White Paper*, vol. 1, 2015.

[66] A. Suliman, Z. Husain, M. Abououf, M. Alblooshi, and K. Salah, "Monetization of iot data using smart contracts," *IET Networks*, vol. 8, no. 1, pp. 32–37, 2018.

[67] A. Back, "Hashcash - a denial of service counter-measure," Accessed: Aug 01, 2002. [Online]. Available: http://www.hashcash.org/papers/hashcash.pdf, 09 2002.

[68] S. King and S. Nadal, "Ppcoin: Peer-to-peer crypto-currency with proof-of-stake," *self-published paper*, vol. 19, pp. 1–6, 2012.

[69] S. Duan, *Building Reliable and Practical Byzantine Fault Tolerance*. University of California, Davis, 2014.

[70] Y. Wu, P. Song, and F. Wang, "Hybrid Consensus Algorithm Optimization: A Mathematical Method Based on PoS and PBFT and Its Application in Blockchain," *Mathematical Problems in Engineering*, vol. 2020, p. 7270624, Apr 2020.

[71] J. Gray and L. Lamport, "Consensus on transaction commit," *ACM Transactions*

on Database Systems (TODS), vol. 31, no. 1, pp. 133–160, 2006.

[72] H. Sukhwani, J. M. Martínez, X. Chang, K. S. Trivedi, and A. Rindos, "Performance modeling of pbft consensus process for permissioned blockchain network (hyperledger fabric)," in *2017 IEEE 36th Symposium on Reliable Distributed Systems (SRDS)*, 2017, pp. 253–255.

[73] T. T. A. Dinh, J. Wang, G. Chen, R. Liu, B. C. Ooi, and K.-L. Tan, "Blockbench: A framework for analyzing private blockchains," in *Proceedings of the 2017 ACM International Conference on Management of Data*, 2017, pp. 1085–1100.

[74] S. Liu, P. Viotti, C. Cachin, V. Quéma, and M. Vukolić, "{XFT}: Practical fault tolerance beyond crashes," in *12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16)*, 2016, pp. 485–500.

[75] M. Milutinovic, W. He, H. Wu, and M. Kanwal, "Proof of luck: An efficient blockchain consensus protocol," in *Proceedings of the 1st Workshop on System Software for Trusted Execution*, ser. SysTEX '16. New York, NY, USA: ACM, 2016, pp. 2:1–2:6. [Online]. Available: http://doi.acm.org/10.1145/3007788.3007790

[76] L. Chen, L. Xu, N. Shah, Z. Gao, Y. Lu, and W. Shi, "On security analysis of proof-of-elapsed-time (poet)," in *International Symposium on Stabilization, Safety, and Security of Distributed Systems*. Springer, 2017, pp. 282–297.

[77] S. Chenthara, K. Ahmed, H. Wang, and F. Whittaker, "A novel blockchain based smart contract system for ereferral in healthcare: Healthchain," in *International Conference on Health Information Science*. Springer, 2020, pp. 91–102.

[78] N. K. Suryadevara and S. C. Mukhopadhyay, "Wireless sensor network based home monitoring system for wellness determination of elderly," *IEEE sensors journal*, vol. 12, no. 6, pp. 1965–1972, 2012.

[79] W. Leister, M. Hamdi, H. Abie, and S. Poslad, "An evaluation framework for adaptive security for the iot in ehealth," *International Journal on Advances*, vol. 7, no. 3, pp. 93–109, 2014.

[80] M. Al Ameen, J. Liu, and K. Kwak, "Security and privacy issues in wireless sensor networks for healthcare applications," *Journal of medical systems*, vol. 36, no. 1,

pp. 93–101, 2012.

[81] H.-T. Wu and C.-W. Tsai, "Toward blockchains for health-care systems: Applying the bilinear pairing technology to ensure privacy protection and accuracy in data sharing," *IEEE Consumer Electronics Magazine*, vol. 7, no. 4, pp. 65–71, 2018.

[82] Q. Zhu, S. W. Loke, R. Trujillo-Rasua, F. Jiang, and Y. Xiang, "Applications of distributed ledger technologies to the internet of things: A survey," *ACM Computing Surveys (CSUR)*, vol. 52, no. 6, pp. 1–34, 2019.

[83] C. Gan, A. Saini, Q. Zhu, Y. Xiang, and Z. Zhang, "Blockchain-based access control scheme with incentive mechanism for ehealth systems: patient as supervisor," *Multimedia Tools and Applications*, pp. 1–17, 2020.

[84] J. Xu, K. Xue, S. Li, H. Tian, J. Hong, P. Hong, and N. Yu, "Healthchain: A blockchain-based privacy preserving scheme for large-scale health data," *IEEE Internet of Things Journal*, vol. 6, no. 5, pp. 8770–8781, 2019.

[85] M. Omoogun, P. Seeam, V. Ramsurrun, X. Bellekens, and A. Seeam, "When ehealth meets the internet of things: Pervasive security and privacy challenges," in *2017 International Conference on Cyber Security And Protection Of Digital Services (Cyber Security)*. IEEE, 2017, pp. 1–7.

[86] N. Dong, H. Jonker, and J. Pang, "Challenges in ehealth: From enabling to enforcing privacy," in *International Symposium on Foundations of Health Informatics Engineering and Systems*. Springer, 2011, pp. 195–206.

[87] M. van der Haak, A. C. Wolff, R. Brandner, P. Drings, M. Wannenmacher, and T. Wetter, "Data security and protection in cross-institutional electronic patient records," *International journal of medical informatics*, vol. 70, no. 2-3, pp. 117–130, 2003.

[88] W. Tang, J. Ren, K. Zhang, D. Zhang, Y. Zhang, and X. Shen, "Efficient and privacy-preserving fog-assisted health data sharing scheme," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 10, no. 6, pp. 1–23, 2019.

[89] K. P. Kibiwott, F. Zhang, A. A. Omala, and D. Adu-Gyamfi, "Secure cloudlet-based ehealth big data system with fine-grained access control and outsourcing decryption from abe." *IJ Network Security*, vol. 20, no. 6, pp. 1149–1162, 2018.

[90] J. W. Kim, K. Edemacu, and B. Jang, "Mppds: Multilevel privacy-preserving data sharing in a collaborative ehealth system," *IEEE Access*, vol. 7, pp. 109 910–109 923, 2019.

[91] K. Edemacu, B. Jang, and J. W. Kim, "Collaborative ehealth privacy and security: An access control with attribute revocatfion based on obdd access structure," *IEEE Journal of Biomedical and Health Informatics*, vol. 24, no. 10, pp. 2960–2972, 2020.

[92] N. R. Hardiker and M. J. Grant, "Factors that influence public engagement with ehealth: A literature review," *International journal of medical informatics*, vol. 80, no. 1, pp. 1–12, 2011.

[93] G. Ateniese, R. Curtmola, B. De Medeiros, and D. Davis, "Medical information privacy assurance: Cryptographic and system aspects," in *International Conference on Security in Communication Networks*. Springer, 2003, pp. 199–218.

[94] A. Dubovitskaya, V. Urovi, M. Vasirani, K. Aberer, and M. I. Schumacher, "A cloud-based ehealth architecture for privacy preserving data integration," in *IFIP International Information Security and Privacy Conference*. Springer, 2015, pp. 585–598.

[95] P. Samarati and L. Sweeney, "Protecting privacy when disclosing information: k-anonymity and its enforcement through generalization and suppression," Computer Science Laboratory, SRI International, Menlo Park, CA, USA, Tech. Rep. SRI-CSL-98-04, 1998.

[96] J. Liu, J. Ma, W. Wu, X. Chen, X. Huang, and L. Xu, "Protecting mobile health records in cloud computing: A secure, efficient, and anonymous design," *ACM Transactions on Embedded Computing Systems (TECS)*, vol. 16, no. 2, pp. 1–20, 2017.

[97] C. Li, A. Raghunathan, and N. K. Jha, "Hijacking an insulin pump: Security attacks and defenses for a diabetes therapy system," in *2011 IEEE 13th International Conference on e-Health Networking, Applications and Services*. IEEE, 2011, pp. 150–156.

[98] P. Pharow and B. Blobel, "Mobile health requires mobile security: Challenges, solutions, and standardization," *Studies in health technology and informatics*, vol.

136, pp. 697–702, 2008.

[99] Q. Jing, A. V. Vasilakos, J. Wan, J. Lu, and D. Qiu, "Security of the internet of things: perspectives and challenges," *Wireless Networks*, vol. 20, no. 8, pp. 2481–2501, 2014.

[100] X. Lin, R. Lu, X. Shen, Y. Nemoto, and N. Kato, "Sage: a strong privacy-preserving scheme against global eavesdropping for ehealth systems," *IEEE Journal on Selected Areas in Communications*, vol. 27, no. 4, pp. 365–378, 2009.

[101] C. Garcia-Perez, A. Diaz-Zayas, A. Rios, P. Merino, K. Katsalis, C.-Y. Chang, S. Shariat, N. Nikaein, P. Rodriguez, and D. Morris, "Improving the efficiency and reliability of wearable based mobile ehealth applications," *Pervasive and Mobile Computing*, vol. 40, pp. 674–691, 2017.

[102] F. Kargl, E. Lawrence, M. Fischer, and Y. Y. Lim, "Security, privacy and legal issues in pervasive ehealth monitoring systems," in *2008 7th International Conference on Mobile Business*. IEEE, 2008, pp. 296–304.

[103] K. Park and H. Lee, "On the effectiveness of route-based packet filtering for distributed dos attack prevention in power-law internets," *ACM SIGCOMM computer communication review*, vol. 31, no. 4, pp. 15–26, 2001.

[104] H. Wang, L. Xu, and G. Gu, "Floodguard: A dos attack prevention extension in software-defined networks," in *2015 45th Annual IEEE/IFIP International Conference on Dependable Systems and Networks*. IEEE, 2015, pp. 239–250.

[105] P. Sangkatsanee, N. Wattanapongsakorn, and C. Charnsripinyo, "Practical real-time intrusion detection using machine learning approaches," *Computer Communications*, vol. 34, no. 18, pp. 2227–2235, 2011.

[106] R. Doshi, N. Apthorpe, and N. Feamster, "Machine learning ddos detection for consumer internet of things devices," in *2018 IEEE Security and Privacy Workshops (SPW)*. IEEE, 2018, pp. 29–35.

[107] A. A. Diro and N. Chilamkurti, "Distributed attack detection scheme using deep learning approach for internet of things," *Future Generation Computer Systems*, vol. 82, pp. 761–768, 2018.

[108] Y. Li, D. E. Quevedo, S. Dey, and L. Shi, "Sinr-based dos attack on remote state

estimation: A game-theoretic approach," *IEEE Transactions on Control of Network Systems*, vol. 4, no. 3, pp. 632–642, 2016.

[109] Y. Yuan, H. Yuan, L. Guo, H. Yang, and S. Sun, "Resilient control of networked control system under dos attacks: A unified game approach," *IEEE transactions on Industrial Informatics*, vol. 12, no. 5, pp. 1786–1794, 2016.

[110] X. Zhang, S. Poslad, and Z. Ma, "A semi-outsourcing secure data privacy scheme for iot data transmission," in *2017 IEEE 28th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*. IEEE, 2017, pp. 1–5.

[111] M. Barua, X. Liang, R. Lu, and X. Shen, "Peace: An efficient and secure patient-centric access control scheme for ehealth care system," in *2011 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*. IEEE, 2011, pp. 970–975.

[112] F. Gonçalves, J. Macedo, M. J. Nicolau, and A. Santos, "Security architecture for mobile e-health applications in medication control," in *2013 21st International Conference on Software, Telecommunications and Computer Networks-(SoftCOM 2013)*. IEEE, 2013, pp. 1–8.

[113] N. Shrestha, A. Alsadoon, P. Prasad, L. Hourany, and A. Elchouemi, "Enhanced e-health framework for security and privacy in healthcare system," in *2016 Sixth International Conference on Digital Information Processing and Communications (ICDIPC)*. IEEE, 2016, pp. 75–79.

[114] Y. He, H. Li, X. Cheng, Y. Liu, C. Yang, and L. Sun, "A blockchain based truthful incentive mechanism for distributed p2p applications," *IEEE Access*, vol. 6, pp. 27 324–27 335, 2018.

[115] G. S. Veronese, M. Correia, A. N. Bessani, and L. C. Lung, "Spin one's wheels? byzantine fault tolerance with a spinning primary," in *2009 28th IEEE International Symposium on Reliable Distributed Systems*. IEEE, 2009, pp. 135–144.

[116] E. Heilman, A. Kendler, A. Zohar, and S. Goldberg, "Eclipse attacks on bitcoin's peer-to-peer network," in *24th {USENIX} Security Symposium ({USENIX} Security 15)*, 2015, pp. 129–144.

[117] S. Zhang and J.-H. Lee, "Eclipse-based stake-bleeding attacks in pos blockchain systems," in *Proceedings of the 2019 ACM International Symposium on Blockchain and Secure Critical Infrastructure*, 2019, pp. 67–72.

[118] Y. Marcus, E. Heilman, and S. Goldberg, "Low-resource eclipse attacks on ethereum's peer-to-peer network." *IACR Cryptology ePrint Archive*, vol. 2018, no. 236, 2018.

[119] G. Xu, B. Guo, C. Su, X. Zheng, K. Liang, D. S. Wong, and H. Wang, "Am i eclipsed? a smart detector of eclipse attacks for ethereum," *Computers & Security*, vol. 88, p. 101604, 2020.

[120] M. Walck, K. Wang, and H. S. Kim, "Tendrilstaller: Block delay attack in bitcoin," in *2019 IEEE International Conference on Blockchain (Blockchain)*. IEEE, 2019, pp. 1–9.

[121] N. Atzei, M. Bartoletti, and T. Cimoli, "A survey of attacks on ethereum smart contracts (sok)," in *International Conference on Principles of Security and Trust*. Springer, 2017, pp. 164–186.

[122] S. Tikhomirov, E. Voskresenskaya, I. Ivanitskiy, R. Takhaviev, E. Marchenko, and Y. Alexandrov, "Smartcheck: Static analysis of ethereum smart contracts," in *2018 IEEE/ACM 1st International Workshop on Emerging Trends in Software Engineering for Blockchain (WETSEB)*. IEEE, 2018, pp. 9–16.

[123] J. He, M. Balunović, N. Ambroladze, P. Tsankov, and M. Vechev, "Learning to fuzz from symbolic execution with application to smart contracts," in *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, 2019, pp. 531–548.

[124] Y. Fu, M. Ren, F. Ma, H. Shi, X. Yang, Y. Jiang, H. Li, and X. Shi, "Evmfuzzer: detect evm vulnerabilities via fuzz testing," in *Proceedings of the 2019 27th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*. ACM, 2019, pp. 1110–1114.

[125] CVE, "CVE-2016-2085," Accessed: Jan. 27, 2016. [Online]. Available: https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2016-2085.

[126] R. Hund, C. Willems, and T. Holz, "Practical timing side channel attacks against kernel space aslr," in *2013 IEEE Symposium on Security and Privacy*. IEEE,

2013, pp. 191–205.

[127] CVE, "CVE-2018-18920," Accessed: Nov. 03, 2018. [Online]. Available: https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2018-18920.

[128] CVE, "CVE-2018-19184," Accessed: Nov. 11, 2018. [Online]. Available: https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2018-19184.

[129] CVE, "CVE-2018-20421," Accessed: Dec. 23, 2018. [Online]. Available: https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2018-20421.

[130] Y. Hirai, "Defining the ethereum virtual machine for interactive theorem provers," in *International Conference on Financial Cryptography and Data Security*. Springer, 2017, pp. 520–535.

[131] E. Hildenbrandt, M. Saxena, X. Zhu, N. Rodrigues, P. Daian, D. Guth, and G. Rosu, "Kevm: A complete semantics of the ethereum virtual machine," IDEALS, University of Illinois, Tech. Rep. 2142/97207, 2017.

[132] D. Park, Y. Zhang, M. Saxena, P. Daian, and G. Roşu, "A formal verification tool for ethereum vm bytecode," in *Proceedings of the 2018 26th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*. ACM, 2018, pp. 912–915.

[133] F. Ma, Y. Fu, M. Ren, M. Wang, Y. Jiang, K. Zhang, H. Li, and X. Shi, "Evm*: From offline detection to online reinforcement for ethereum virtual machine," in *2019 IEEE 26th International Conference on Software Analysis, Evolution and Reengineering (SANER)*. IEEE, 2019, pp. 554–558.

[134] L. Luu, D.-H. Chu, H. Olickel, P. Saxena, and A. Hobor, "Making smart contracts smarter," in *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*. ACM, 2016, pp. 254–269.

[135] B. Jiang, Y. Liu, and W. Chan, "Contractfuzzer: Fuzzing smart contracts for vulnerability detection," in *Proceedings of the 33rd ACM/IEEE International Conference on Automated Software Engineering*. ACM, 2018, pp. 259–269.

[136] S. Kalra, S. Goel, M. Dhawan, and S. Sharma, "Zeus: Analyzing safety of smart contracts." in *Network and Distributed System Security Symposium (NDSS)*, 2018.

[137] C. Liu, H. Liu, Z. Cao, Z. Chen, B. Chen, and B. Roscoe, "Reguard: finding reen-

trancy bugs in smart contracts," in *Proceedings of the 40th International Conference on Software Engineering: Companion Proceeedings*. ACM, 2018, pp. 65–68.

[138] M. Di Angelo and G. Salzer, "A survey of tools for analyzing ethereum smart contracts," in *2019 IEEE International Conference on Decentralized Applications and Infrastructures (DAPPCON)*. IEEE, 2019.

[139] I. Nikolić, A. Kolluri, I. Sergey, P. Saxena, and A. Hobor, "Finding the greedy, prodigal, and suicidal contracts at scale," in *Proceedings of the 34th Annual Computer Security Applications Conference*, 2018, pp. 653–663.

[140] X. Zhao, Z. Chen, X. Chen, Y. Wang, and C. Tang, "The dao attack paradoxes in propositional logic," in *2017 4th International Conference on Systems and Informatics (ICSAI)*. IEEE, 2017, pp. 1743–1746.

[141] P. Technologies, "Security alert," Accessed: Nov. 08, 2017. [Online]. Available: https://www.parity.io/security-alert-2/.

[142] S. PALLADINO, "The parity wallet hack explained," Accessed: Jul. 19, 2017. [Online]. Available: https://blog.openzeppelin.com/on-the-parity-wallet-multisig-hack-405a8c12e8f7/.

[143] Wikipedia, "Cryptocurrency and security," Accessed: Sep. 17, 2019. [Online]. Available: https://en.wikipedia.org/wiki/Cryptocurrency_and_security.

[144] C. Adams and S. Lloyd, *Understanding PKI: concepts, standards, and deployment considerations*. Addison-Wesley Professional, Boston, MA, USA, 2003.

[145] A. Azaria, A. Ekblaw, T. Vieira, and A. Lippman, "Medrec: Using blockchain for medical data access and permission management," in *2016 2nd International Conference on Open and Big Data (OBD)*. IEEE, 2016, pp. 25–30.

[146] X. Yue, H. Wang, D. Jin, M. Li, and W. Jiang, "Healthcare data gateways: found healthcare intelligence on blockchain with novel privacy risk control," *Journal of medical systems*, vol. 40, no. 10, p. 218, 2016.

[147] A. Dubovitskaya, Z. Xu, S. Ryu, M. Schumacher, and F. Wang, "Secure and trustable electronic medical records sharing using blockchain," in *AMIA annual symposium proceedings*, vol. 2017. American Medical Informatics Association, 2017, p. 650.

[148] N. Rifi, E. Rachkidi, N. Agoulmine, and N. C. Taher, "Towards using blockchain technology for ehealth data access management," in *2017 Fourth International Conference on Advances in Biomedical Engineering (ICABME)*. IEEE, 2017, pp. 1–4.

[149] P. Genestier, S. Zouarhi, P. Limeux, D. Excoffier, A. Prola, S. Sandon, and J.-M. Temerson, "Blockchain for consent management in the ehealth environment: A nugget for privacy and security challenges," *Journal of the International Society for Telemedicine and eHealth*, vol. 5, pp. GKR–e24, 2017.

[150] P. Zhang, J. White, D. C. Schmidt, G. Lenz, and S. T. Rosenbloom, "Fhirchain: applying blockchain to securely and scalably share clinical data," *Computational and structural biotechnology journal*, vol. 16, pp. 267–278, 2018.

[151] J. Chen, X. Ma, M. Du, and Z. Wang, "A blockchain application for medical information sharing," in *2018 IEEE International Symposium on Innovation and Entrepreneurship (TEMS-ISIE)*. IEEE, 2018, pp. 1–7.

[152] A. Al Omar, M. Z. A. Bhuiyan, A. Basu, S. Kiyomoto, and M. S. Rahman, "Privacy-friendly platform for healthcare data in cloud based on blockchain environment," *Future generation computer systems*, vol. 95, pp. 511–521, 2019.

[153] R. Bosri, A. R. Uzzal, A. Al Omar, M. Z. A. Bhuiyan, and M. S. Rahman, "Hidechain: A user-centric secure edge computing architecture for healthcare iot devices," in *IEEE INFOCOM 2020-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*. IEEE, 2020, pp. 376–381.

[154] IBM, "Medical-Blockchain: Store private healthcare data off-chain and manage medical data using blockchain," Accessed: Feb. 27, 2019. [Online]. Available: https://github.com/IBM/Medical-Blockchain.

[155] J. Liu, X. Li, L. Ye, H. Zhang, X. Du, and M. Guizani, "Bpds: A blockchain based privacy-preserving data sharing for electronic medical records," in *2018 IEEE Global Communications Conference (GLOBECOM)*. IEEE, 2018, pp. 1–6.

[156] F. Tang, S. Ma, Y. Xiang, and C. Lin, "An efficient authentication scheme for blockchain-based electronic health records," *IEEE access*, vol. 7, pp. 41 678–41 689, 2019.

[157] T. T. Thwin and S. Vasupongayya, "Blockchain-based access control model to preserve privacy for personal health record systems," *Security and Communication Networks*, vol. 2019, 2019.

[158] H. Guo, W. Li, M. Nejad, and C.-C. Shen, "Access control for electronic health records with hybrid blockchain-edge architecture," in *2019 IEEE International Conference on Blockchain (Blockchain)*. IEEE, 2019, pp. 44–51.

[159] A. Khatoon, "A blockchain-based smart contract system for healthcare management," *Electronics*, vol. 9, no. 1, p. 94, 2020.

[160] R. Li, T. Song, B. Mei, H. Li, X. Cheng, and L. Sun, "Blockchain for large-scale internet of things data storage and protection," *IEEE Transactions on Services Computing*, vol. 12, no. 5, pp. 762–771, 2018.

[161] J. Daemen and V. Rijmen, "Rijndael, the advanced encryption standard," *Dr. Dobb's Journal*, vol. 26, no. 3, pp. 137–139, 2001.

[162] T. Eisenbarth and S. Kumar, "A survey of lightweight-cryptography implementations," *IEEE Design & Test of Computers*, vol. 24, no. 6, 2007.

[163] C. A. R. Hoare, "Communicating sequential processes," *Communications of the ACM*, vol. 21, no. 8, pp. 666–677, 1978.

[164] G. Lowe, "Casper: A compiler for the analysis of security protocols," in *Proceedings 10th Computer Security Foundations Workshop*. IEEE, 1997, pp. 18–30.

[165] T. Gibson-Robinson, P. Armstrong, A. Boulgakov, and A. W. Roscoe, "FDR3 — A Modern Refinement Checker for CSP," in *Tools and Algorithms for the Construction and Analysis of Systems*, E. Ábrahám and K. Havelund, Eds., vol. 8413, 2014, pp. 187–201.

[166] M. Aiash and J. Loo, "Introducing a novel authentication protocol for secure services in heterogeneous environments using casper/fdr," *International Journal of Communication Systems*, vol. 27, no. 12, pp. 3600–3618, 2014.

[167] A. Armando, D. Basin, Y. Boichut, Y. Chevalier, L. Compagna, J. Cuéllar, P. H. Drielsma, P.-C. Héam, O. Kouchnarenko, J. Mantovani *et al.*, "The avispa tool for the automated validation of internet security protocols and applications," in *International conference on computer aided verification*. Springer, 2005, pp. 281–

285.

[168] B. Blanchet, "Modeling and verifying security protocols with the applied pi calculus and proverif," *Foundations and Trends® in Privacy and Security*, vol. 1, no. 1-2, pp. 1–135, 2016.

[169] L. Atzori, A. Iera, and G. Morabito, "The internet of things: A survey," *Computer networks*, vol. 54, no. 15, pp. 2787–2805, 2010.

[170] R. Chow, P. Golle, M. Jakobsson, E. Shi, J. Staddon, R. Masuoka, and J. Molina, "Controlling data in the cloud: outsourcing computation without outsourcing control," in *Proceedings of the 2009 ACM workshop on Cloud computing security*. ACM, 2009, pp. 85–90.

[171] J. Huang, L. Kong, G. Chen, M.-Y. Wu, X. Liu, and P. Zeng, "Towards secure industrial iot: Blockchain system with credit-based consensus mechanism," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 6, pp. 3680–3689, 2019.

[172] M. H. Eldefrawy, N. Pereira, and M. Gidlund, "Key distribution protocol for industrial internet of things without implicit certificates," *IEEE Internet of Things Journal*, vol. 6, no. 1, pp. 906–917, 2018.

[173] C. Patsakis and F. Casino, "Hydras and ipfs: a decentralised playground for malware," *International Journal of Information Security*, vol. 18, no. 6, pp. 787–799, 2019.

[174] T. Fan and Y. Chen, "A scheme of data management in the Internet of Things," in *Network Infrastructure and Digital Content, 2010 2nd IEEE International Conference on*. IEEE, 2010, pp. 110–114.

[175] G. Margelis, R. Piechocki, D. Kaleshi, and P. Thomas, "Low throughput networks for the IoT: Lessons learned from industrial implementations," in *Internet of Things (WF-IoT), 2015 IEEE 2nd World Forum on*. IEEE, 2015, pp. 181–186.

[176] 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.

[177] L. Lu, J. Han, Y. Liu, L. Hu, J.-P. Huai, L. Ni, and J. Ma, "Pseudo trust: Zero-knowledge authentication in anonymous p2ps," *IEEE Transactions on Parallel and*

Distributed Systems, vol. 19, no. 10, pp. 1325–1337, 2008.

[178] X. Liang, J. Zhao, S. Shetty, J. Liu, and D. Li, "Integrating blockchain for data sharing and collaboration in mobile healthcare applications," in *2017 IEEE 28th annual international symposium on personal, indoor, and mobile radio communications (PIMRC)*. IEEE, 2017, pp. 1–5.

[179] Z. Xiao, Z. Li, Y. Liu, L. Feng, W. Zhang, T. Lertwuthikarn, and R. S. M. Goh, "Emrshare: A cross-organizational medical data sharing and management framework using permissioned blockchain," in *2018 IEEE 24th International Conference on Parallel and Distributed Systems (ICPADS)*. IEEE, 2018, pp. 998–1003.

[180] S. Smetanin, A. Ometov, M. Komarov, P. Masek, and Y. Koucheryavy, "Blockchain evaluation approaches: State-of-the-art and future perspective," *Sensors*, vol. 20, no. 12, p. 3358, 2020.

[181] G. Zyskind, O. Nathan *et al.*, "Decentralizing privacy: Using blockchain to protect personal data," in *Security and Privacy Workshops (SPW), 2015 IEEE*. IEEE, 2015, pp. 180–184.

[182] J. Quirós-Tortós, L. F. Ochoa, and B. Lees, "A statistical analysis of ev charging behavior in the uk," in *2015 IEEE PES Innovative Smart Grid Technologies Latin America (ISGT LATAM)*. IEEE, 2015, pp. 445–449.

[183] O. Hafez and K. Bhattacharya, "Queuing analysis based pev load modeling considering battery charging behavior and their impact on distribution system operation," *IEEE Transactions on Smart Grid*, vol. 9, no. 1, pp. 261–273, 2016.

[184] G. Wood *et al.*, "Ethereum: A secure decentralised generalised transaction ledger," *Ethereum project yellow paper*, vol. EIP-151, pp. 1–32, 2014.

[185] P. Ekparinya, V. Gramoli, and G. Jourjon, "The attack of the clones against proof-of-authority," in *Network and Distributed System Security Symposium (NDSS)*, 2020.

[186] K. Gai, Y. Wu, L. Zhu, L. Xu, and Y. Zhang, "Permissioned blockchain and edge computing empowered privacy-preserving smart grid networks," *IEEE Internet of Things Journal*, vol. 6, no. 5, pp. 7992–8004, 2019.

[187] X. Min, Q. Li, L. Liu, and L. Cui, "A permissioned blockchain framework for sup-

porting instant transaction and dynamic block size," in *2016 IEEE Trustcom/BigDataSE/ISPA*. IEEE, 2016, pp. 90–96.

[188] C. Hong, J. Katz, V. Kolesnikov, W.-j. Lu, and X. Wang, "Covert security with public verifiability: Faster, leaner, and simpler," in *Advances in Cryptology – EUROCRYPT 2019*, Y. Ishai and V. Rijmen, Eds. Cham: Springer International Publishing, 2019, pp. 97–121.

Appendix A

Correctness for Block-based Access Control Scheme in Chapter 3

In the *Authorise* phase, the two points $P = P'$ is the key condition for a user to pass the authorisation phase.

$P = P'$

$\Leftrightarrow kG = k'G$

$\Leftrightarrow (TK_{id_1} \oplus ... \oplus TK_{id_n})G = (TK'_{id_1} \oplus ... \oplus TK'_{id_n})G$

$\Leftrightarrow TK_{id_1} \oplus ... \oplus TK_{id_n} = TK'_{id_1} \oplus ... \oplus TK'_{id_n}$

$\Leftrightarrow \{TK_{id_1}, ..., TK_{id_n}\} = \{TK'_{id_1}, ..., TK'_{id_n}\}.$

It means the user can pass the *Authorise* phase if and only if this user has offered all the required access permission tokens in Q correctly for the requested blocks.

In the *Decrypt* phase, the user can succeed to decrypt the ciphertext C and ensure

$H_M = H(M)$ if the condition $k = k'$ is fulfilled. According to the correctness analysis for the *Authorise* phase, the condition $k = k'$ can hold if the user is authorised to access the queried blocks successfully. To be detailed, the condition $k = k'$ is equivalent to $\{TK_{id_1}, TK_{id_2}, ..., TK_{id_n}\} = \{TK'_{id_1}, TK'_{id_2}, ..., TK'_{id_n}\}$, i.e., the sequences $S_{TK} = S'_{TK}$. Therefore, the keys k and k' in the phases *Encrypt* and *Decrypt* respectively are identical.

Hence, the authorised user can decrypt the ciphertext C with the key k correctly.

Appendix B

Theoretical Security Analysis for the Proposed Schemes in Chapter 4

B.1 Semi-outsourcing Key Distribution (SOKD) Scheme

B.1.1 Correctness

In the *Authentication* phase, the public clouds can recover H_{ID} via computing $C_4 \oplus H_3(bC_2, C_1, C_2, C_3)$ based upon the algorithm *Authentication* in Section IV.A.4).

$C_4 \oplus H_3(bC_2, C_1, C_2, C_3)$

$= H_{ID} \oplus H_3(r_2B, C_1, C_2, C_3) \oplus H_3(bC_2, C_1, C_2, C_3)$

$= H_{ID} \oplus H_3(r_2B, C_1, C_2, C_3) \oplus H_3(br_2G, C_1, C_2, C_3)$

$= H_{ID} \oplus H_3(r_2B, C_1, C_2, C_3) \oplus H_3(r_2(bG), C_1, C_2, C_3)$

$= H_{ID} \oplus H_3(r_2B, C_1, C_2, C_3) \oplus H_3(r_2B, C_1, C_2, C_3)$

$= H_{ID}$

In the *Decrypt* phase, When the data centre receives $C' = (H_{ID}, C_1, C_2, C_3)$ from the public clouds, the data centre can retrieve encrypted data $AES_{H_{ID}}(H_M||M)$ via computing $C_3 \oplus H_2(aC_1)$ based upon the algorithm *Decrypt* in Section IV.A.5).

$C_3 \oplus H_2(aC_1)$

$= AES_{H_{ID}}(H_M||M) \oplus H_2(r_1A) \oplus H_2(aC_1)$

$= AES_{H_{ID}}(H_M||M) \oplus H_2(r_1A) \oplus H_2(ar_1G)$

$= AES_{H_{ID}}(H_M||M) \oplus H_2(r_1A) \oplus H_2(r_1(aG))$

$= AES_{H_{ID}}(H_M||M) \oplus H_2(r_1A) \oplus H_2(r_1A)$

$= AES_{H_{ID}}(H_M||M)$

Then the data centre can decrypt $AES_{H_{ID}}(H_M||M)$ with the AES key $H_{ID} \in C'$ to get the plaintext $H_M||M$.

B.1.2 Theoretical confidentiality

B.1.2.1 OW-CCA confidentiality model

In this section, the adversary and the definition of OW-CCA (i.e., one-wayness under a chosen ciphertext attack) confidentiality model are defined for our SOKD scheme.

• *Type-I adversary*: Formally, the adversary defined for the confidentiality of the scheme SOKD is that the adversary cannot retrieve the plain message from the challenge ciphertext in the *Authentication* phase.

• OW-CCA model

The definition of OW-CCA confidentiality model with the *Type-I adversary* for the *Authentication* phase in SOKD is described in following *Game 1*.

Game 1. *$\mathcal{A}_1$ is the given Type-I adversary, and the target device's index is t ($1 \leqslant t \leqslant n$). The game 1 between the challenger $\mathcal{C}$ and $\mathcal{A}_1$ is operated as follows:*

- *Setup*

$\mathcal{C}$ firstly generates the public parameter pp via running the algorithm $Setup$. Then, $\mathcal{C}$ generates n public and private key pairs (pk_i, sk_i) $(1 \leqslant i \leqslant n)$ via running the algorithm $KeyGenerate$. The generated pp and all pk_i are given to the adversary $\mathcal{A}_1$.

- *Queries*

The following queries can be requested by $\mathcal{A}_1$ for polynomial times.

1. *Key retrieve query*(i): $\mathcal{C}$ responds with the private key sk_i.

2. *Authentication query*(i, C): $\mathcal{C}$ returns the trapdoor $\mathcal{T}_i$ to recover $f(ID)$.

3. *Decryption query*(i, C'): $\mathcal{C}$ decrypts C' with sk_i via running the algorithm $Decrypt(C', sk_i)$, and responds with the output message.

- *Challenge*

$\mathcal{C}$ picks a message M^* randomly, then computes the challenge ciphertext $C^* = Encrypt(M^*, pk_t)$ and finally responds the challenge ciphertext C^*.

- *Constraints*

1. The target device's index t is not allowed to appear in the above *Key retrieve query*.

2. The target device's index t and the challenge ciphertext C^* is not allowed to appear in the above *Decryption query*.

- *Guess*

$\mathcal{A}_1$ can win the game if its output $M^{*'}$ satisfies the condition $M^{*'} = M^*$.

Now, the advantage of $\mathcal{A}_1$ could be defined as:

$$Adv_{\mathcal{A}_1}^{OW-CCA}(\lambda) = Pr[M^* = M^{*'}].$$

Definition 1 (OW-CCA Security). *The proposed scheme SOKD is OW-CCA secure if the advantage $Adv_{\mathcal{A}_1}^{OW-CCA}(\lambda)$ of any probabilistic polynomial-time adversary $\mathcal{A}_1$ is negligible.*

B.1.2.2 Proof of OW-CCA confidentiality

Theorem 1. *According to Definition 1, our proposed scheme SOKD is OW-CCA secure based upon the ECCDH assumption against a Type-I adversary in the random oracle model.*

To be specific, let H_1, H_2 and H_3 be three random oracles and $\mathcal{A}_1$ be a Type-I adversary with the advantage $Adv_{\mathcal{A}_1}$ against our proposed scheme. Hypothetically, $\mathcal{A}_1$ requests a total of $Q_{H_2} > 0$ queries to the oracle H_2, then there is an algorithm $\mathcal{E}$ that can solve the ECCDH problem with the advantage at least of $\frac{1}{Q_{H_2}}(Adv_{\mathcal{A}_1} - \frac{1}{2^\lambda})$.

Proof. The selected elliptic curve $E_p(a, b)$ with cryptographic security, the group $\mathbb{G}$ is based upon $E_p(a, b)$ and the three points on the curve $(G, \mu G, \upsilon G) \in E_p(a, b)$ consist of an instance of the ECCDH problem and the target device's index is defined as t $(1 \leqslant t \leqslant n)$. $\mathcal{E}$ aims to compute $\delta^* = \mu\upsilon G$ via executing $\mathcal{A}_1$ as the subroutine. Next, $\mathcal{E}$ and $\mathcal{A}_1$ play the game defined by **Game 1**.

- *Setup*

$\mathcal{E}$ firstly generates the public parameter pp and then sends pp to $\mathcal{A}_1$. After that, $\mathcal{E}$ operates the algorithm *KeyInitialise* to generate n public and private key pairs (pk_i, sk_1^i, sk_2^i) $(1 \leqslant i \leqslant n,\ i \neq t)$. In this process, the target device's public key is defined as $pk_t = (A_t, B_t),\ A_t = \mu_t G,\ B_t = \upsilon_t G$, where $\mu_t, \upsilon_t \in_R \mathbb{Z}_p^*$ is picked randomly. All pk_i are revealed to the adversary $\mathcal{A}_1$. Finally, $\mathcal{E}$ initialises three empty lists $List_{H_1}$, $List_{H_2}$ and $List_{H_3}$, and updates them continuously in response to random oracle queries. If the same input

is asked multiple times, the same answer will be returned as well.

- *Queries*

$\mathcal{E}$ can respond to the queries requested by $\mathcal{A}_1$ in the following ways:

1. $Query_{H_1}(\gamma_1)$: $\mathcal{E}$ picks $\delta_1 \in \{0,1\}^{\lambda}$ randomly and stores a new item (γ_1, δ_1) into $List_{H_1}$ and returns δ_1 as the answer.

2. $Query_{H_2}(\gamma_2)$: $\mathcal{E}$ picks $\delta_2 \in \{0,1\}^{2\lambda}$ randomly and stores a new item (γ_2, δ_2) into $List_{H_2}$ and returns δ_2 as the answer.

3. $Query_{H_3}(\gamma_3, C_1, C_2, C_3)$: $\mathcal{E}$ picks $\delta_3 \in \{0,1\}^{\lambda}$ randomly and stores a new item $(\gamma_3, C_1, C_2, C_3, \delta_3)$ into $List_{H_3}$ and returns δ_3 as the answer.

4. *Key retrieve query*(i): $\mathcal{E}$ sends the private key $sk_1^i = (\mu_i), sk_1^i = (\upsilon_i)$ to $\mathcal{A}_1$.

5. *Authentication query*(i, C): $\mathcal{E}$ returns the trapdoor $\mathcal{T}_i = H_3(\upsilon_i C_2, C_1, C_2, C_3)$ to recover H_{ID}, where $C = (C_1, C_2, C_3, C_4)$.

6. *Decryption query*(i, C'): The definition of parameter C' is $C' = (C_1, C_2, C_3)$, and there is a conditional branch caused by i to be discussed.

† $i = t$: For each item (γ_2, δ_2) in the $List_{H_2}$, $\mathcal{E}$ performs the following operations.

(i) Compute $AES_{H_{ID}}(H_M||M) = C_3 \oplus \delta_2$ and $H_{ID} = C_4 \oplus H_3(\upsilon_t C_2, C_1, C_2, C_3)$;

(ii) Compute $M = AES'_{H_{ID}}(AES_{H_{ID}}(H_M||M))$;

(iii) If $H_1(M) = H_M$ holds, $\mathcal{E}$ returns M to $\mathcal{A}_1$. If there is no item in the $List_{H_2}$ satisfies the above condition, $\mathcal{E}$ returns $\perp$ to $\mathcal{A}_1$.

‡ $i \neq t$: $\mathcal{E}$ runs algorithm $Decrypt(pp, C, sk_1^i)$, and then sends the output to $\mathcal{A}_1$ as the answer.

- *Challenge*

$\mathcal{E}$ picks two random numbers, one is $r_1 \in_R \mathbb{Z}_p^*$ and the other one is $r^* \in \{0,1\}^{2\lambda}$. Then, $\mathcal{E}$ generates a random message $M^* \in \{0,1\}^{\lambda}$ and computes the ciphertext $C^* = (C_1^*, C_2^*, C_3^*, C_4^*)$ via the following operations.

$$C_1^* = \upsilon G$$
$$C_2^* = r_2 G$$
$$C_3^* = r^*$$
$$C_4^* = H_{ID} \oplus H_3(r_2 B_t, C_1^*, C_2^*, C_3^*)$$

Note that the process of decrypting C_3^* is $r^* \oplus H_2(\mu C_1^*) = r^* \oplus H_2(\mu \upsilon G) = r^* \oplus H_2(\delta^*)$ by the definition of *Decrypt*.

Finally, $\mathcal{E}$ sends the ciphertext C^* to the adversary $\mathcal{A}_1$.

- *Constraints*

1. The target device's index t is not allowed to appear in the *Key retrieve query*;

2. The target device's index t and the challenge ciphertext C' are not allowed to appear in *Decryption query*.

- *Guess*

$\mathcal{A}_1$ outputs $M^{*'} \in \{0,1\}^{\lambda}$ to response the challenge from $\mathcal{E}$. And at the same time, $\mathcal{E}$ picks a random item (γ_2, δ_2) from the $List_{H_2}$ as the solution to the above given instance of ECCDH problem.

- *Analysis*

I first define an event E that means the adversary $\mathcal{A}_1$ issues a query $H_2(\delta^*)$ at a time point during the described game. Apparently, δ^* is at least in one item of $List_{H_2}$ at the end of this game if E happened.

However, if E does not happen, I can state that $Pr[M^* = M^{*'} | \neg E] = \frac{1}{2^{\lambda}}$. Furthermore, based upon the definition of Type-I adversary ($\mathcal{A}_1$), $Adv_{\mathcal{A}_1} \leqslant Pr[M^* = M^{*'}]$ holds. Then, I can present the following derivation.

$Pr[M^* = M^{*'}]$

$= Pr[M^* = M^{*'}|E]Pr[E] + Pr[M^* = M^{*'}|\neg E]Pr[\neg E]$

$\leqslant Pr[E] + Pr[M^* = M^{*'}|\neg E]Pr[\neg E]$

$= Pr[E] + \frac{1}{2^\lambda}Pr[\neg E]$

$= Pr[E] + \frac{1}{2^\lambda}(1 - Pr[E])$

$= \frac{1}{2^\lambda} + (1 - \frac{1}{2^\lambda})Pr[E]$

Therefore, the following inequation holds:

$$\frac{1}{2^\lambda} + (1 - \frac{1}{2^\lambda})Pr[E] \geqslant Pr[M^* = M^{*'}] \geqslant Adv_{\mathcal{A}_1}.$$

Finally, I can simplify the inequation to get:

$$Pr[E] \geqslant Adv_{\mathcal{A}_1} - \frac{1}{2^\lambda}.$$

In conclusion, at the end of the game between $\mathcal{E}$ and $\mathcal{A}_1$, the probability of δ^* in the item(s) of $List_{H_2}$ is at least $Adv_{\mathcal{A}_1} - \frac{1}{2^\lambda}$. For $\mathcal{E}$, the probability of generating the correct answer $M^{*'} = M^*$ to solve the ECCDH problem is at least $\frac{1}{Q_{H_2}}(Adv_{\mathcal{A}_1} - \frac{1}{2^\lambda})$. Therefore, the probability $Adv_{\mathcal{A}_1}$ is negligible when the ECCDH assumption is intact. □

B.2 Granular Access Authorisation supporting Flexible Queries (GAA-FQ) Scheme

B.2.1 Correctness

In the *Authorise* phase, $H_T = H'_T$ is the condition for a successful authorisation.

$$
\begin{aligned}
& H_T = H'_T \\
& \Leftrightarrow H(S_2) = H(S'_2) \\
& \Leftrightarrow H(\{BT_1, BT_2, ..., BT_n\}) = H(\{BT'_1, BT'_2, ..., BT'_n\}) \\
& \Leftrightarrow \{BT_1, BT_2, ..., BT_n\} = \{BT'_1, BT'_2, ..., BT'_n\}.
\end{aligned}
$$

It means the user can pass the *Authorise* phase if and only if this user has all the corresponding access permission tokens for the requested blocks.

In the *Decrypt* phase, the authorised inquirer can retrieve the key $k = a_0 \in \mathbb{Z}_q$ for *AES* decryption based upon the *SSharing.Reconstruction*. Given a point $P(x_p, y_p)$ on the polynomial $f(x)$, it should be on the correct reconstructed polynomial as well. Because the condition $\{BT_1, BT_2, ..., BT_n\} = \{BT'_1, BT'_2, ..., BT'_n\}$ holds after the authorisation, the reconstructed polynomial $g(x)$ is same as the original polynomial $f(x)$ except for the unknown first coefficient a_0. This means determining the secret $g(0) = a_0 = k \in \mathbb{Z}_q$ requires only one point (shareholder) $P(x_p, y_p)$:

$$
\begin{aligned}
& k = a_0 \\
& = g(x_p) - BT_1 x_p - BT_2 x_p^2 - ... - BT_n x_p^n \\
& = y_p - BT_1 x_p - BT_2 x_p^2 - ... - BT_n x_p^n \\
& = f(x_p) - BT_1 x_p - BT_2 x_p^2 - ... - BT_n x_p^n \\
& = f(x_p) - BT'_1 x_p - BT'_2 x_p^2 - ... - BT'_n x_p^n \\
& = k \ (mod\ q).
\end{aligned}
$$

Hence, the authorised user can reconstruct the polynomial $g(x)$ and restore the correct key k for the *Decrypt* phase.

B.2.2 Theoretical confidentiality

B.2.2.1 IND-CCA confidentiality model

In this section, the adversary and the definition of IND-CCA (i.e., indistinguishability under chosen-ciphertext attack) confidentiality model are defined for our GAA-FQ scheme.

Formally, the adversary defined to prove the theoretical security of our proposed GAA-FQ scheme is: *Type-IND adversary.*

• *Type-IND adversary*: In the *Authorise* phase, the adversary cannot determine the message that the given challenge ciphertext is encrypted from, without knowing the target block tokens $BT^t = \{BT_1^t, BT_2^t, ..., BT_n^t\}$.

The definition of IND-CCA confidentiality model with the *Type-IND adversary* for the *Authorise* phase in GAA-FQ scheme is described in following *Game 2.*

• IND-CCA model

Game 2. *Let $\mathcal{A}_1$ be the given Type-IND adversary, and the index of the target block tokens be t ($1 \leqslant t \leqslant m$). The game played by the algorithm $\mathcal{C}$ and the adversary $\mathcal{A}_1$ is described with the following five phases:*

• *Setup*

$\mathcal{C}$ first generates the public parameter *pp* via running the algorithm *Setup*. The generated *pp* and all indexes $1 \leqslant i \leqslant m$ are given to the adversary $\mathcal{A}_1$. Note that all the queries are assumed to be block query.

• *Queries*

The following queries can be requested by $\mathcal{A}_1$ for polynomial times in the game:

1. *Block tokens query* (i): $\mathcal{C}$ responds with the sequence of the random block tokens BT^i;

2. *Decrypt query*(C, P, i): $\mathcal{C}$ decrypts C via running the algorithm *Decrypt*, then responds with the plain message.

- *Challenge*

$\mathcal{A}_1$ submits two equal-length messages M_0^* and M_1^*. $\mathcal{C}$ picks $\rho \in_R \{0,1\}$, and then computes and returns the challenge ciphertext $C^* = Encrypt(pp, M_\rho^*, BT^t)$.

- *Constraints*

1. t is not allowed to appear in the *Block tokens query*;

2. The target data provider's index t and the challenge ciphertext C^* are not allowed to appear in the above *Decrypt query*.

- *Guess*

$\mathcal{A}_1$ can win the game if its output $\rho' \in_R \{0,1\}$ satisfies the condition $\rho = \rho'$.

Now, the advantage of $\mathcal{A}_1$ could be defined as:

$$Adv_{\mathcal{A}_1}^{IND-CCA}(\lambda) = Pr[\rho = \rho'] - \tfrac{1}{2},$$

where λ is the security parameter. Note that the probability analysis is presented after the *Guess* phase in the formal confidentiality proof of our GAA-FQ scheme.

Definition 2 (IND-CCA Security). *The GAA-FQ scheme is IND-CCA secure if the advantage* $Adv_{\mathcal{A}_1}^{IND-CCA}(\lambda) > 0$ *of any probabilistic polynomial-time adversary* $\mathcal{A}_1$ *is negligible.*

B.2.2.2 Proof of IND-CCA confidentiality

Theorem 2. *According to Definition 2 above, the proposed GAA-FQ scheme is IND-CCA secure based on the Lagrange interpolating polynomial against the type-IND adversary in the random oracle model.*

To be specific, let $\mathcal{O}_{BT}$ *be a random oracle and* $\mathcal{A}_1$ *be a Type-IND adversary with*

the advantage $Adv_{\mathcal{A}_1}^{IND-CCA}$ *against our proposed scheme. Hypothetically,* $\mathcal{A}_1$ *requests a total of* $Q_{\mathcal{O}_{BT}} > 0$ *queries to the oracle* $\mathcal{O}_{BT}$*; then there is an algorithm* $\mathcal{C}$ *that can determine all the correct coefficients for the given Lagrange interpolating polynomial with the advantage of at least* $\frac{2}{Q_{\mathcal{O}_{BT}}} Adv_{\mathcal{A}_1}^{IND-CCA}$.

Proof. A polynomial $f(x)$ with the sequence of the block tokens $BT^i = \{BT_1^i, AT_2^i, ..., AT_n^i\}$ $(1 \leqslant i \leqslant m)$ consists of an instance of the Lagrange interpolating polynomial, where $f(x) = a_0^i + H(AT_1^{v_i})x + H(AT_2^{v_i})x^2 + ... + H(AT_n^{v_i})x^n$. Note that a_0^i represents the AES encryption/decryption key generated in the *Encrypt* phase. The target block tokens' index is denoted by t $(1 \leqslant t \leqslant m)$. The algorithm $\mathcal{C}$ aims to determine BT^t through performing $\mathcal{A}_1$ as the subroutine. Next, $\mathcal{C}$ and $\mathcal{A}_1$ play the game defined in Appendix B.2.2.1.

- *Setup*

$\mathcal{C}$ first generates the public parameter $pp = (q, BC_{AT}, H, AES)$ and then sends pp to $\mathcal{A}_1$. After that, $\mathcal{C}$ generates m indexes of the sequences including many block tokens $\{i | 1 \leqslant i \leqslant m\}$ and the target index and sequence are denoted by $\{t, BT^t\}$. Note that all the generated i $(1 \leqslant i \leqslant m)$ are given to the adversary $\mathcal{A}_1$. Finally, $\mathcal{C}$ initialises one empty lists $List_\gamma$ and updates it continuously in the random oracle query *Block tokens query*. If the same input is asked multiple times, the same answer will be returned.

- *Queries*

$\mathcal{C}$ can respond to the queries requested by $\mathcal{A}_1$ polynomial times in the following ways.

1. *Block tokens query* (i): $\mathcal{C}$ asks $\mathcal{O}_{BT}$ to generate the sequence of the block tokens $BT^i = \{BT_1^i, BT_2^i, ..., BT_n^i\}$ randomly and saves (i, BT^i) in $List_{\mathcal{O}_{BT}}$ if it is the first time that i is queried. Then, $\mathcal{C}$ respond with the sequence BT^i. Otherwise, $\mathcal{C}$ should retrieve the sequence BT^i from $List_{\mathcal{O}_{BT}}$ and return it to $\mathcal{A}_1$ directly.

2. *Decrypt query*(C, P, i): $\mathcal{C}$ tries to decrypt C via running $Decrypt(pp, C, P, BT^i)$

then responds with the plain message. Note that there is a conditional branch caused by i to be discussed.

If $i = t$, for each item (i, BT^i) in $List_{\mathcal{O}_{BT}}$, $\mathcal{C}$ executes the operations.

– Reconstruct the Lagrange interpolating polynomial $g^*(x)$ with BT^i and P to determine the secret key $k = a_0$ for AES decryption.

– Recover (M, H_M) by computing $AES'_k(C) = AES'_k(AES_k(M, H_M))$.

– If $H(M) = H_M$ holds, $\mathcal{C}$ returns M to $\mathcal{A}_1$. If there is no item in the $List_{\mathcal{O}_{BT}}$ that satisfies the condition, $\mathcal{C}$ returns $\perp$ to $\mathcal{A}_1$.

If $i \neq t$, $\mathcal{C}$ runs the $Decrypt(pp, C, P, BT^i)$ algorithm directly and then sends the output to $\mathcal{A}_1$ as the answer.

• *Challenge*

$\mathcal{A}_1$ submits two messages $M_1^*, M_2^* \in \{0,1\}^\lambda$ with the same length to $\mathcal{C}$, then $\mathcal{C}$ picks one random bit ρ from the set $\{0, 1\}$. Finally, $\mathcal{C}$ computes the ciphertext C^* of M_ρ^* via the following steps:

1. Choose a secret key $k \in \mathbb{Z}_q$ for AES encryption and decryption;

2. Determine $f^*(x) = k + (BT_1^t)x + (BT_2^t)x^2 + ... + (BT_n^t)x^n \ (mod\ q)$;

3. Pick a random point $P^*(x^*, f^*(x^*))$ on $f^*(x)$;

4. Compute $C^* = AES_k(M_\rho^*, H(M_\rho^*))$.

Finally, $\mathcal{C}$ sends the ciphertext C^* and the point P^* to the adversary $\mathcal{A}_1$.

• *Constraints*

1. t is not allowed to appear in *Identity attributes query*;

2. The target index t and the challenge ciphertext C^* are not allowed to appear in

the above *Decrypt query*.

- *Guess*

$\mathcal{A}_1$ outputs one bit ρ' from the set $\{0, 1\}$. At the same time, $\mathcal{C}$ picks a random element (i, BT^i) from $List_{\mathcal{O}_{BT}}$ as the answer to the above given instance of the Lagrange interpolating polynomial.

- *Probability analysis*

An event $\mathcal{E}$ is defined as that the adversary $\mathcal{A}_1$ issues a query to generate the target sequence BT^t in the *Block tokens query* at a time point during the described game above. If the event $\mathcal{E}$ has happened, BT^t appears in at least one item of $List_{\mathcal{O}_{BT}}$ at the end of this game.

However, if $\mathcal{E}$ does not happen, it means that $Pr[\rho^* = \rho^{*'}|\neg\mathcal{E}] = \frac{1}{2}$ holds. Meanwhile, the condition $Adv_{\mathcal{A}_1}^{IND-CCA} \leqslant Pr[\rho = \rho'] - \frac{1}{2}$ holds because of the definition of the type-IND adversary ($\mathcal{A}_1$). Based upon the above analysis, the following derivation can be illustrated,

$$\begin{aligned} Pr[\varphi = \varphi'] &= Pr[\varphi = \varphi'|\mathcal{E}]Pr[\mathcal{E}] + Pr[\varphi = \varphi'|\neg\mathcal{E}]Pr[\neg\mathcal{E}] \\ &\leqslant Pr[\mathcal{E}] + Pr[\varphi = \varphi'|\neg\mathcal{E}]Pr[\neg\mathcal{E}] \\ &= Pr[\mathcal{E}] + \tfrac{1}{2}Pr[\neg\mathcal{E}] \\ &= \tfrac{1}{2} + \tfrac{1}{2}Pr[\mathcal{E}]. \end{aligned}$$

Hence, I can deduce that the following derivation holds:

$$Adv_{\mathcal{A}_1}^{IND-CCA} \leqslant Pr[\rho = \rho'] - \tfrac{1}{2} \leqslant \tfrac{1}{2}Pr[\mathcal{E}].$$

I can simplify this derivation such that $Pr[\mathcal{E}] \geqslant 2Adv_{\mathcal{A}_1}^{IND-CCA}$.

In conclusion, at the end of the game between the algorithm $\mathcal{C}$ and the adversary $\mathcal{A}_1$,

the probability of the target sequence BT^t being in $List_{\mathcal{O}_{BT}}$ is at least $2Adv_{\mathcal{A}_1}^{IND-CCA}$. Hence, the probability of generating the correct answer $\rho = \rho'$ is at least $\frac{2}{Q_{\mathcal{O}_{BT}}} Adv_{\mathcal{A}_1}^{IND-CCA}$, where $Q_{\mathcal{O}_{BT}}$ represents the total number of the queries to the random oracle $\mathcal{O}_{BT}$. □

B.2.3 Theoretical unforgeability

B.2.3.1 EUF-CMA unforgeability model

The security model I use to prove the integrity of our scheme GAA-FQ is *existentially unforgeable under adaptively chosen-message attacks* (EUF-CMA).

Definition 3 (EUF-CMA Security). *The scheme GAA-FQ is called EUF-CMA security, if for the adversary $\mathcal{A}_2$ can access to an Encrypt oracle $\mathcal{O}_{enc}(M, i)$ $(1 \leqslant i \leqslant m)$, the advantage*

$$Adv_{\mathcal{A}_2}^{EUF-CMA}(\lambda) = Pr[Decrypt(C^*, BT^t) = M^* | M^* \notin \mathcal{M} \wedge C^* \leftarrow \mathcal{A}_2^{\mathcal{O}_{enc}(M^*,i)}],$$

is negligible, where $\mathcal{M}$ is the set of plain messages that $\mathcal{A}_2$ has known the corresponding ciphertext from the oracle $\mathcal{O}_{enc}$, λ is the security parameter, and t is the index of the specified block tokens' sequence (BT^t) by the challenger $\mathcal{C}$ with m sequences in total.

To be specific, the adversary $\mathcal{A}_2$ can play the following game with the challenger $\mathcal{C}$ and win the game with the negligible probability ϵ.

Game 3. *$\mathcal{A}_2$ is a given adversary that can obtain valid signatures from queries. The game between the challenger $\mathcal{C}$ and $\mathcal{A}_2$ is operated as follows.*

- *Setup*

$\mathcal{C}$ first generates the public parameter *pp* via running the algorithm *Setup*. The generated *pp* and all indexes $1 \leqslant i \leqslant m$ are given to the adversary $\mathcal{A}_2$. Note that all the queries are assumed to be block query.

- *Queries*

The following queries can be requested by $\mathcal{A}_2$ for polynomial times. If the information of the same index i is queried multiple times, the same outputs should be offered to $\mathcal{A}_2$.

1. *Message query* : $\mathcal{C}$ responds with a random message M.

2. *Block tokens query*(i) : $\mathcal{C}$ responds with the sequence of the queried block tokens $BT^i = \{BT_1^i, BT_2^i, ..., BT_n^i\}$ and saves this sequence in the list $\mathcal{L}_{BT}$.

3. *Encrypt query*(M, i) : $\mathcal{C}$ executes the algorithm *Encrypt*(M, BT^i) then responds with the ciphertext C for the message M to the adversary $\mathcal{A}_2$. Note that the pair of the message and ciphertext is appended to the list $\mathcal{M}$.

- *Guess*

The adversary $\mathcal{A}_2$ first picks a random message M^* from *Message query* and then chooses a BT^i. Then, $\mathcal{A}_2$ uses the algorithm *Encrypt* to compute the ciphertext C^* for M^*. Finally, $\mathcal{A}_2$ submits M^*, C^* to the challenger $\mathcal{C}$. $\mathcal{A}_2$ can win this game if $\mathcal{C}$ runs the algorithm *Decrypt* and outputs M^*.

- *Constrains*

1. t is not allowed to appear in *Block tokens query*;

2. $\mathcal{A}_2$ cannot submit the message that $\mathcal{A}_2$ has known its ciphertext for BT^t.

I can state that GAA-FQ is EUF-CMA security if the probability $Adv_{\mathcal{A}_2}^{EUF-CMA}(\lambda)$ is negligible with polynomial-time queries in the *Queries* phase under the mentioned constrains.

B.2.3.2 Proof of EUF-CMA unforgeability

Theorem 1. *According to Definition 3, our GAA-FQ scheme is EUF-CMA secure based upon Lagrange interpolation polynomials against the adversary $\mathcal{A}_2$ in the random oracle model.*

To be specific, let $\mathcal{O}_M$ *and* $\mathcal{O}_{BT}$ *be two random oracles;* $\mathcal{O}_{enc}$ *be a real oracle;* $\mathcal{A}_2$ *be the adversary with a non-negligible advantage* ϵ *against the scheme GAA-FQ. Hypothetically,* $\mathcal{A}_2$ *requests a total of* $Q_{BT} > 0$ *queries to the random oracle* $\mathcal{O}_{BT}$*, then there is an challenger* $\mathcal{C}$ *that can determined all the coefficients of the specified Lagrange interpolation polynomial with the advantage at least* $\frac{1}{Q_{BT}} \frac{2^{\lambda}\epsilon - 1}{2^{\lambda} - 1}$.

Proof. Let $H : \{0,1\}^* \rightarrow \{0,1\}^{\lambda}$ be a secure hash function. There are m sequences of block tokens $BT^i = \{BT_1^i, BT_2^i, ..., BT_n^i\}$ $(1 \leqslant i \leqslant m)$. The target sequence's index is defined as t $(1 \leqslant t \leqslant m)$. The algorithm $\mathcal{C}$ aims to figure out BT^t via executing $\mathcal{A}_1$ as the subroutine. Next, $\mathcal{C}$ and $\mathcal{A}_2$ play the following game.

- *Setup*

$\mathcal{C}$ firstly use the algorithm *Setup* in GAA-FQ to generate the public parameter $pp = (q, BC_{AT}, AES)$. The generated pp and all indexes $1 \leqslant i \leqslant m$ are given to the adversary $\mathcal{A}_2$. Note that all the queries are assumed to be block query.

- *Queries*

The following queries can be requested by $\mathcal{A}_2$ for polynomial times. If the information of the same index i is queried multiple times, the same outputs should be offered to $\mathcal{A}_2$.

1. $Message\ query : \mathcal{C}$ responds with a random message M from $\mathcal{O}_M$.

2. *Block tokens query*$(i) : \mathcal{C}$ queries $\mathcal{O}_{BT}$ and then responds with the sequence of the queried block tokens $BT^i = \{BT_1^i, BT_2^i, ..., BT_n^i\}$. Finally, $\mathcal{C}$ saves this sequence in the list $\mathcal{L}_{BT}$.

3. *Encrypt query*$(M, i) : \mathcal{C}$ asks the real oracle $\mathcal{O}_{enc} : Encrypt(M, BT^i)$ then responds with the ciphertext C for the message M to the adversary $\mathcal{A}_2$. Note that the pair of the message and ciphertext $((M, i), C)$ is appended to the list $\mathcal{M}$.

- *Guess*

The adversary $\mathcal{A}_2$ first picks a random message M^* by querying $\mathcal{O}_M$ from *Message query*. Then, after utilising all the ciphertext from $\mathcal{O}_{enc}$ and $\mathcal{L}_{BT}$ to determine a sequence BT^i, $\mathcal{A}_2$ uses the real oracle $\mathcal{O}_{enc}$ to compute the ciphertext C^* for M^*. Finally, $\mathcal{A}_2$ submits M^*, C^* to the challenger $\mathcal{C}$. $\mathcal{A}_2$ can win this game if $\mathcal{C}$ runs the algorithm $Decrypt(C^*, BT^t)$ and outputs M^*.

- *Constrains*

1. $BT^t \notin \mathcal{L}_{BT}$ i.e., t is not allowed to appear in $\mathcal{O}_{BT}$;

2. $(M^*, t) \notin \mathcal{M}$ in *Guess*, i.e., $\mathcal{A}_2$ cannot submit the message that $\mathcal{A}_2$ has known its ciphertext from $\mathcal{O}_{enc}$.

- *Probability analysis*

I define the event E is that BT^t appears in $\mathcal{L}_{BT}$. If E does not happen, $Pr[Decrypt(C^*, BT^t) = M^* | \neg E] = \frac{1}{2^\lambda}$. Based upon Definition. 3, the adversary $\mathcal{A}_2$ has an advantage:

$$\begin{aligned} Adv_{\mathcal{A}_2}^{EUF-CMA} &\leqslant Pr[Decrypt(C^*, BT^t) = M^*] \\ &\leqslant Pr[E] + Pr[Decrypt(C^*, BT^t) = M^* | \neg E] Pr[\neg E] \\ &= Pr[E] + \frac{1}{2^\lambda} Pr[\neg E] \\ &= \frac{1}{2^\lambda} + (1 - \frac{1}{2^\lambda}) Pr[E], \end{aligned}$$

to win the game. It means $\mathcal{A}_2$ has the advantage $Pr[E] \geqslant \frac{2^\lambda Adv_{\mathcal{A}_2}^{EUF-CMA} - 1}{2^\lambda - 1}$ to figure out BT^t and hence to determine the specified Lagrange interpolation polynomial with the *Decrypt* algorithm in GAA-FQ. Hence, the advantage of $\mathcal{C}$ using $\mathcal{A}_2$ as the subroutine to determine the specified Lagrange interpolation polynomial is at least

$$\frac{1}{Q_{BT}} \frac{2^\lambda \epsilon - 1}{2^\lambda - 1}.$$

Therefore, I can state that the scheme GAA-FQ can satisfy EUF-CMA security, i.e., the probability of $Adv_{\mathcal{A}_2}^{EUF-CMA}$, ϵ is negligible if the Lagrange interpolation polynomial

is intact. □

Appendix C

Theoretical Security Analysis for the Proposed Scheme in Chapter 5

C.1 Correctness

In the *Authorise* phase, if the data requester has the correct sequence of the identity attributes AT'^v, the condition $AT^v = AT'^v$ holds,

$AT^v = AT'^v$

$\Leftrightarrow \{H(AT_1^v), H(AT_2^v), ..., H(AT_n^v)\} = \{H(AT_1'^v), H(AT_2'^v), ..., H(AT_n'^v)\}$

$\Leftrightarrow f(x) = g(x)$

$\Leftrightarrow P_y = P'_y$ $(for\ the\ given\ P_x)$

$\Leftrightarrow PH_y = PH'_y$.

This means that the data requester can pass the *Authorise* phase if and only if this requester has the correct corresponding sequence of the identity attributes for the

requested blocks.

To satisfy the condition in the correctness definition, the authorised data requester should retrieve the key $k \in \mathbb{Z}_q$ for AES decryption with the given point $P(x_p, y_p)$ on the polynomial $f(x)$ in the *Decrypt* phase. Meanwhile, P should present on the correct reconstructed polynomial as well. Since the condition $\{H(AT_1^v), H(AT_2^v), ..., H(AT_n^v)\} = \{H(AT_1'^v), H(AT_2'^v), ..., H(AT_n'^v)\}$ holds after the *Authorise* phase, the reconstructed polynomial $g(x)$ is the same as the original polynomial $f(x)$ except for the unknown first coefficient $a_0 = k$. Therefore, determining the secret key $g(0) = a_0 = k \in \mathbb{Z}_q$ for AES decryption requires only one point (shareholder) $P(x_p, y_p)$:

$$k = a_0$$

$$= g(x_p) - AT_1'^v x_p - AT_2'^v x_p^2 - ... - AT_n'^v x_p^n$$

$$= f(x_p) - AT_1'^v x_p - AT_2'^v x_p^2 - ... - AT_n'^v x_p^n$$

$$= f(x_p) - AT_1^v x_p - AT_2^v x_p^2 - ... - AT_n^v x_p^n$$

$$= k \ (mod\ q).$$

Hence, the authorised data requester can reconstruct the polynomial $g(x)$ and restore the correct key k in the *Decrypt* phase to ensure $Decrypt(pp, C, AT'^v) = M$ holds, where $C = Encrypt(pp, M, AT^v)$.

C.2 Theoretical confidentiality

C.2.1 IND-CCA confidentiality model

In this section, the adversary and the definition of IND-CCA (i.e., indistinguishability under chosen-ciphertext attack) confidentiality model are defined for our CRA^3 scheme.

Formally, the adversary defined to prove the theoretical security of our proposed CRA^3 scheme is: *Type-IND adversary.*

• *Type-IND adversary*: In the *Authorise* phase, the adversary cannot determine the message that the given challenge ciphertext is encrypted from, without knowing the target identity attributes AT^{v_t}.

The definition of IND-CCA confidentiality model with the *Type-IND adversary* for the *Authorise* phase in CRA^3 scheme is described in following *Game 4*.

• IND-CCA model

Game 4. *There are m data providers involved in this game. Let $\mathcal{A}_1$ be the given Type-IND adversary, and the index of the target data provider be t $(1 \leqslant t \leqslant m)$. The game played by the algorithm $\mathcal{C}$ and the adversary $\mathcal{A}_1$ is described with the following five phases:*

• *Initialise*

$\mathcal{C}$ first generates the public parameter pp via running the algorithm *Setup*. Then, $\mathcal{C}$ generates m data providers $\{Rn^i | (1 \leqslant i \leqslant m)\}$ and the target data provider is Rn^t. The generated pp and all Rn^i $(1 \leqslant i \leqslant m)$ are given to the adversary $\mathcal{A}_1$.

• *Queries*

The following queries can be requested by $\mathcal{A}_1$ for polynomial times in the game:

1. *Identity attributes query* (i): $\mathcal{C}$ responds with the sequence of the random identity attributes AT^{v_i};

2. *Encrypt query*(M, i): $\mathcal{C}$ outputs the ciphertext $C = Encrypt(pp, M, AT^{v_i})$ and the point P on the constructed polynomial $f(x)$ in the *Encrypt* phase;

3. *Decrypt query*(C, P, i): $\mathcal{C}$ decrypts C via running the algorithm *Decrypt*, then responds with the plain message.

• *Challenge*

$\mathcal{A}_1$ submits two equal-length messages M_0^* and M_1^*. $\mathcal{C}$ picks $\rho \in_R \{0, 1\}$, and then

computes and returns the challenge ciphertext $C^* = Encrypt(pp, M^*_\rho, AT^{v_t})$.

- *Constraints*

1. t is not allowed to appear in the *Identity attributes query*;
2. (M^*_0, t) and (M^*_1, t) are not allowed to appear in the above *Encrypt query*;
3. The target data provider's index t and the challenge ciphertext C^* are not allowed to appear in the above *Decrypt query*.

- *Guess*

$\mathcal{A}_1$ can win the game if its output $\rho' \in_R \{0, 1\}$ satisfies the condition $\rho = \rho'$.

Now, the advantage of $\mathcal{A}_1$ could be defined as:

$$Adv^{IND-CCA}_{\mathcal{A}_1}(\lambda) = |Pr[\rho = \rho'] - \tfrac{1}{2}|,$$

where λ is the security parameter. Note that the probability analysis is presented after the *Guess* phase in the formal confidentiality proof of our CRA^3 scheme.

Definition 4 (IND-CCA Security). *The CRA^3 scheme is IND-CCA secure if the advantage $Adv^{IND-CCA}_{\mathcal{A}_1}(\lambda)$ of any probabilistic polynomial-time adversary $\mathcal{A}_1$ is negligible.*

C.2.2 Proof of IND-CCA confidentiality

Theorem 4. *According to Definition 4 above, the proposed CRA^3 scheme is IND-CCA secure based on the Lagrange interpolating polynomial against the type-IND adversary in the random oracle model.*

To be specific, let γ be a random oracle and $\mathcal{A}_1$ be a Type-IND adversary with the advantage $Adv^{IND-CCA}_{\mathcal{A}_1}$ against our proposed scheme. Hypothetically, $\mathcal{A}_1$ requests a total of $Q_\gamma > 0$ queries to the oracle γ; then there is an algorithm $\mathcal{C}$ that can determine all the correct coefficients for the given Lagrange interpolating polynomial with the advantage of

at least $\frac{2}{Q_\gamma} Adv_{\mathcal{A}_1}^{IND-CCA}$.

Proof. A polynomial $f(x)$ with the sequence of the identity attributes $AT^{v_i} = \{AT_1^{v_i}, AT_2^{v_i}, ..., AT_n^{v_i}\}$ $(1 \leqslant i \leqslant m)$ and a secure hash function $H : \{0,1\}^* \rightarrow \{0,1\}^\lambda$ consist of an instance of the Lagrange interpolating polynomial, where $f(x) = a_0^i + H(AT_1^{v_i})x + H(AT_2^{v_i})x^2 + ... + H(AT_n^{v_i})x^n$ and a_0^i denotes the encryption key randomly generated in the algorithm *Encrypt* of CRA[3]. The target data provider's index is defined as t $(1 \leqslant t \leqslant m)$. The algorithm $\mathcal{C}$ aims to determine AT^{v_t} via executing $\mathcal{A}_1$ as the subroutine. Next, $\mathcal{C}$ and $\mathcal{A}_1$ play the game defined in Appendix C.2.1.

- *Initialise*

$\mathcal{C}$ first generates the public parameter $pp = (q, H, AES)$ and then sends pp to $\mathcal{A}_1$. After that, $\mathcal{C}$ generates n data providers $\{Rn^i | 1 \leqslant i \leqslant m\}$ and the target data provider and its identity attributes are denoted by $\{Rn^t, AT^{v_t}\}$. Note that all the generated Rn^i $(1 \leqslant i \leqslant m)$ are given to the adversary $\mathcal{A}_1$. Finally, $\mathcal{C}$ initialises one empty lists $List_\gamma$ and updates it continuously in the random oracle query *Identity attributes query*. If the same input is asked multiple times, the same answer will be returned.

- *Queries*

$\mathcal{C}$ can respond to the queries requested by $\mathcal{A}_1$ polynomial times in the following ways.

1. *Identity attributes query* (i): $\mathcal{C}$ generates the sequence of the identity attributes $AT^{v_i} = \{AT_1^{v_i}, AT_2^{v_i}, ..., AT_n^{v_i}\}$ randomly and saves (i, AT^{v_i}) in $List_\gamma$ if it is the first time that i is queried. Then, $\mathcal{C}$ respond with the sequence AT^{v_i}. Otherwise, $\mathcal{C}$ should retrieve the sequence AT^{v_i} from $List_\gamma$ and return it to $\mathcal{A}_1$ directly.

2. *Encrypt query*(M, i): $\mathcal{C}$ uses the algorithm *Encrypt* to output the ciphertext $C = Encrypt(pp, M, AT^{v_i})$ and the point P (P should be on the polynomial constructed with AT^{v_i} in the algorithm *Encrypt*).

3. *Decrypt query*(C, P, i): $\mathcal{C}$ tries to decrypt C via running $Decrypt(pp, C, P, AT^{v_i})$

then responds with the plain message. Note that there is a conditional branch caused by i to be discussed.

If $i = t$, for each item (i, AT^{v_i}) in $List_\gamma$, $\mathcal{C}$ executes the operations.

– Reconstruct the Lagrange interpolating polynomial $g^*(x)$ with AT^{v_i} and P to determine the secret key $k = a_0$ for AES decryption.

– Recover (M, H_M) by computing $AES'_k(C) = AES'_k(AES_k(M, H_M))$.

– If $H(M) = H_M$ holds, $\mathcal{C}$ returns M to $\mathcal{A}_1$. If there is no item in the $List_\gamma$ that satisfies the condition, $\mathcal{C}$ returns $\perp$ to $\mathcal{A}_1$.

If $i \neq t$, $\mathcal{C}$ runs the $Decrypt(pp, C, P, AT^{v_i})$ algorithm directly and then sends the output to $\mathcal{A}_1$ as the answer.

• *Challenge*

$\mathcal{A}_1$ submits two messages $M_1^*, M_2^* \in \{0,1\}^\lambda$ with the same length to $\mathcal{C}$, then $\mathcal{C}$ picks one random bit ρ from the set $\{0, 1\}$. Finally, $\mathcal{C}$ computes the ciphertext C^* of M_ρ^* via the following steps:

1. Choose a secret key $k \in \mathbb{Z}_q$ for AES encryption and decryption;
2. Determine $f^*(x) = k + H(AT_1^{v_t})x + H(AT_2^{v_t})x^2 + ... + H(AT_n^{v_t})x^n$;
3. Pick a random point $P^*(x^*, f^*(x^*))$ on $f^*(x)$;
4. Compute $C^* = AES_k(M_\rho^*, H(M_\rho^*))$.

Finally, $\mathcal{C}$ sends the ciphertext C^* and the point P^* to the adversary $\mathcal{A}_1$.

• *Constraints*

1. t is not allowed to appear in *Identity attributes query*;
2. (M_0^*, t) and (M_1^*, t) are not allowed to appear in the above *Encrypt query*;

3. The target data provider's index t and the challenge ciphertext C^* are not allowed to appear in the above *Decrypt query*.

- *Guess*

$\mathcal{A}_1$ outputs one bit ρ' from the set $\{0, 1\}$. At the same time, $\mathcal{C}$ picks a random element (i, AT^{v_i}) from $List_\gamma$ as the answer to the above given instance of the Lagrange interpolating polynomial.

- *Probability analysis*

An event $\mathcal{E}$ is defined as that the adversary $\mathcal{A}_1$ issues a query to generate the target sequence AT^{v_t} in the *Identity attributes query* during the described game above. If the event $\mathcal{E}$ has happened, AT^{v_t} occurs in at least one item of $List_\gamma$ at the end of this game.

However, if $\mathcal{E}$ does not happen, it means that $Pr[\rho^* = \rho^{*'}|\neg\mathcal{E}] = \frac{1}{2}$ holds. Meanwhile, the condition $Adv_{\mathcal{A}_1}^{IND-CCA} \leqslant |Pr[\rho = \rho'] - \frac{1}{2}|$ holds because of the definition of the type-IND adversary ($\mathcal{A}_1$). Based upon the above analysis, the next two derivations can be illustrated.

$$\begin{aligned}
Pr[\varphi = \varphi'] &= Pr[\varphi = \varphi'|\mathcal{E}]Pr[\mathcal{E}] + Pr[\varphi = \varphi'|\neg\mathcal{E}]Pr[\neg\mathcal{E}] \\
&\leqslant Pr[\mathcal{E}] + Pr[\varphi = \varphi'|\neg\mathcal{E}]Pr[\neg\mathcal{E}] \\
&= Pr[\mathcal{E}] + \tfrac{1}{2}Pr[\neg\mathcal{E}] \\
&= Pr[\mathcal{E}] + \tfrac{1}{2}(1 - Pr[\mathcal{E}]) \\
&= \tfrac{1}{2} + \tfrac{1}{2}Pr[\mathcal{E}] \\
Pr[\varphi = \varphi'] &\geqslant Pr[\varphi = \varphi'|\neg\mathcal{E}]Pr[\neg\mathcal{E}] \\
&= \tfrac{1}{2}Pr[\neg\mathcal{E}] \\
&= \tfrac{1}{2} - \tfrac{1}{2}Pr[\mathcal{E}]
\end{aligned}$$

Hence, I can deduce that the following derivation holds:

$$Adv_{\mathcal{A}_1}^{IND-CCA} \leqslant |Pr[\rho = \rho'] - \tfrac{1}{2}| \leqslant \tfrac{1}{2} Pr[\mathcal{E}].$$

I can simplify this derivation such that $Pr[\mathcal{E}] \geqslant 2Adv_{\mathcal{A}_1}$.

In conclusion, at the end of the game between the algorithm $\mathcal{C}$ and the adversary $\mathcal{A}_1$, the probability of the target sequence AT^{v_t} being in the element(s) of $List_\gamma$ is at least $2Adv_{\mathcal{A}_1}^{IND-CCA}$. Hence, the probability of generating the correct answer $\rho = \rho'$ is at least $\frac{2}{Q_\gamma} Adv_{\mathcal{A}_1}^{IND-CCA}$, where Q_γ represents the total number of the elements in the list $List_\gamma$. □