BLOCKCHAIN APPLICATIONS BEYOND CRYPTOCURRENCY

WANG FAT LAU

Mphil

The Hong Kong Polytechnic University

2023

The Hong Kong Polytechnic University

Department of Computing

Blockchain Applications beyond Cryptocurrency

Wang Fat Lau

A thesis submitted in partial fulfillment of the
requirements for the degree of Master of Philosophy

March 2023

CERTIFICATE OF ORIGINALITY

I hereby declare that this thesis is my own work and that, to the best of my knowledge and belief, it reproduces no material previously published or written, nor material that has been accepted for the award of any other degree or diploma, except where due acknowledgement has been made in the text.

Signature: _____

Student Name: Wang Fat Lau

ABSTRACT

Blockchain technology ensures immutability, transparency, and decentralization of transaction information. Originally designed to support cryptocurrency, blockchain technology has been integrated into applications in various domains. Yet, native adoption of blockchain technology may not fulfill all security requirements of the target applications. We realize that by integrating proper cryptographic primitives on top of the blockchain technology to some applications, the security and performance can be improved.

In this thesis, we explore three different blockchain applications beyond cryptocurrency. We propose new designs with enhanced security and performance for each scenario. Specifically, we made the following contributions:

(1) We proposed a blockchain-based e-voting system which does not rely on a single trusted party. The system achieves verifiability, eligibility, fairness, and anonymity. Specifically, we combined threshold cryptographic technique, ElGamal decryption scheme and blind signature to address the issues of single trusted party and efficiency. In addition, the time complexity of all voter functions is constant, which is efficient and practical for real-world scenario.

(2) We proposed an efficient message authentication system for vehicular ad-hoc network (VANET) which allows revokable transparency, without requiring an online security mediator or an additional secure communication channel. In particular, we combined a new revocable signature scheme, KUNodes algorithm, cuckoo filter and blockchain technology to achieve the security, efficiency and functional requirements of our system. The proposed digital signature supports batch and online/offline verification. It also avoids the of digital use certificate and bilinear pairing for efficiency concerns. Besides, our experiment results also showed that our roadside

unit (RSU) assisted signature verification protocol had a significant improvement on the overall efficiency.

(3) We proposed an efficient blockchain-based supply chain management system for regulated industries. Our proposal scheme was based on practical settings and designs from the literature. In particular, we support four logistic behavioral patterns, quality management and different scenarios of tracing for regulation enforcement and anti-counterfeiting. In addition, we introduced a threshold twisted ElGamal decryption scheme for maintaining better privacy and auditability. Currently, our system has been deployed to and applied in pharmaceutical companies in China.

PUBLICATIONS ARISING FROM THE THESIS

[1] Borui Gong, Xingye Lu, Wang Fat Lau, and Man Ho Au. "Blockchain-based threshold electronic voting system". In: *Security and Privacy in Social Networks and Big Data*. Ed. by Weizhi Meng and Steven Furnell. Springer Singapore, 2019, pp. 238–250. ISBN: 978-981-15-0758-8.

[2] Wang Fat Lau, Dennis Y. W. Liu, and Man Ho Au. "Blockchain-based supply chain system for traceability, regulation and anti-counterfeiting". In: *2021 IEEE International Conference on Blockchain (Blockchain)*. 2021, pp. 82–89. DOI: 10.1109/Blockchain53845.2021.00022.

[3] Kang Li, Wang Fat Lau, and Man Ho Au. "A secure and efficient privacy-preserving authentication scheme for vehicular networks with batch verification using cuckoo filter". In: *Network and System Security*. Ed. by Joseph K. Liu and Xinyi Huang. Springer International Publishing, 2019, pp. 615–631. ISBN: 978-3-030-36938-5.

[4] Kang Li, Wang Fat Lau, Man Ho Au, Ivan Wang-Hei Ho, and Yilei Wang. "Efficient message authentication with revocation transparency using blockchain for vehicular networks". In: *Computers & Electrical Engineering* 86 (2020), p. 106721. ISSN: 0045-7906. DOI: https://doi.org/10.1016/j.compeleceng.2020.106721. URL: http://www.sciencedirect.com/science/article/pii/S0045790620305760.

OTHERS PUBLICATIONS

[1] Shabnam Kasra Kermanshahi, Shi-Feng Sun, Joseph K. Liu, Ron Steinfeld, Surya Nepal, Wang Fat Lau, and Man Ho Allen Au. "Geometric Range Search on Encrypted Data With Forward/Backward Security". In: *IEEE Transactions on Dependable and Secure Computing* 19.1 (2022), pp. 698–716. DOI: 10.1109/TDSC.2020.2982389.

[2] Xiao Yang, Wang Fat Lau, Qingqing Ye, Man Ho Au, Joseph K. Liu, and Jacob Cheng. "Practical Escrow Protocol for Bitcoin". In: *IEEE Transactions on Information Forensics and Security* 15 (2020), pp. 3023–3034. DOI: 10.1109/TIFS.2020.2976607.

[3] Zuoxia Yu, Man Ho Au, Jiangshan Yu, Rupeng Yang, Qiuliang Xu, and Wang Fat Lau. "New Empirical Traceability Analysis of CryptoNote-Style Blockchains". In: *Financial Cryptography and Data Security*. Ed. by Ian Goldberg and Tyler Moore. Cham: Springer International Publishing, 2019, pp. 133–149. ISBN: 978-3-030-32101-7.

ACKNOWLEDGMENTS

First and foremost, I would like to express my sincere gratitude to Prof. Man Ho Au, Prof. Daniel Xiapu Luo and Dr. Yan Wang Dennis Liu for offering me this precious opportunity to join the research team. In these years, the works I have done in Hong Kong Polytechnic University is one of the largest achievements in my life. I deeply appreciate and thank for their continue supports and guidance. In particular, I would like to thank Prof. Au, who guided my research and taught me knowledge of cryptographic and blockchain technologies for many years. Also, I would like to give thanks to Dr. Liu who was also my diploma instructor long time ago and who enlightened my interest of computer science.

I would also thank my colleagues and the whole research team, especially who worked with me before, including Prof. Haiyang Xue, Zuoxia Yu, Borui Gong, Xinyu Li, Kang Li and Xiao Yang. I love the supportive and friendly working environment there.

Lastly, I would like to thank my wife and my parents for their unconditional supports over the years.

TABLE OF CONTENTS

CHAPTER PAGE

LIST OF TABLES

LIST OF FIGURES

FIGURE PAGE

LIST OF ABBREVIATIONS

ABS	Attribute-based Signature
CDH	Computational Diffie-Hellman
CL-PKC	Certificateless Public Key Cryptographic
CL-PKS	Certificateless Public Key Signature
DAG	Directed Acyclic Graph
DDH	Decision Diffie-Hellman
EC	Elliptic Curve
ECS	Elastic Compute Service
EDI	Electronic Data Interchange
ERP	Enterprise Resource Planning
GDH	Gap Diffie-Hellman
IOS	Inter-organizational System
IoT	Internet of Things
ITS	Intelligent Transportation System
KGC	Key Generation Center
NFT	Non-fungible Token
OBU	On-board Unit
ORCLS	Outsourced Revocable Certificateless Signature
P2P	Peer to Peer
PID	Pseudo Identity
PKE	Public Key Encryption
PoA	Proof of Authorities
PoS	Proof of Stake
PoW	Proof of Work

PUF	Physically Unclonable Function
RFID	Radio Frequency Identification
RPC	Remote Procedure Call
RSU	Roadside Unit
SEM	Online Security Mediator
tps	Transactions per Second
TX	Transaction
UID	Unique Identifier
UTXO	Unspent Transaction Output
V2V	Vehicle to Vehicle
VANET	Vehicular Ad-hoc Network
ZKP	Zero-knowledge Proof

CHAPTER 1

INTRODUCTION

The concept of blockchain can be traced back to 2008 when Satoshi Nakamoto [74] proposed a new distributed digital currency and payment system called Bitcoin. The term blockchain is named because of its data structure in which data are divided into blocks and linked as a data chain through cryptographic hashes. This design can ensure data is immutable in the system. In other words it provides data integrity which is required in many applications. After the success of Bitcoin, many digital currency projects based on blockchain technology were launched. Since these systems rely on cryptographic algorithms to maintain their security, nowadays they usually referred to as cryptocurrencies. They formed the mainstream application of blockchain.

In 2014, a new blockchain system, Ethereum, was proposed by Vitalik Buterin [10]. It demonstrated how blockchain can serve as a generic platform and support applications in different fields besides cryptocurrency. According to the latest review [47], blockchain applications can be classified to 23 different fields, including finance, government, transportation, healthcare and supply chain. Until 2018, 151 literatures on blockchain applications appear in top conferences/journals [47]. Besides, the surveys from PwC [79, 80] also showed the increasing interest of blockchain applications from the business and industry.

Undoubtedly, research in blockchain applications is popular and has a huge impact to the world. According to [47], cryptocurrency and financial applications draw the most attention in blockchain research, although other areas can also create huge value by adopting blockchain technology [79, 80]. Therefore, we would like to investigate specifically on blockchain applications in non-financial fields. We will present three applications, namely, electronic voting (e-voting) system, message authentica-

tion scheme for vehicular ad-hoc network (VANET) and supply chain management system, in this thesis.

1.1 Threshold Electronic Voting System

Government services are very important functions for the society. It is in the top 3 most valuable fields of blockchain application developments [79]. E-voting, as an essential field of government services, has gained considerable amount of interest over the years. In 1981, Chaum [12] introduced the idea and proposed a concrete design. The proposed scheme empowers voters to cast their vote remotely and ballots can be transmitted securely through the internet. Due to the remote nature and the importance of e-voting, it is easily a target of attackers. In detail, a secure e-voting system should satisfy soundness, completeness, eligibility, unreusability, verifiability, fairness and privacy [32]. Usually, they are achieved by using cryptographic techniques such as mix-net [12], homomorphic encryption [81], linkable ring signature [64] and blind signature [13]. A brief review of these techniques is given below.

Mix-net is a technique to remove linkage between the voters' identities and their ballots by means of shuffling the data. Homomorphic encryption enables user to perform arithmetic operations directly in ciphertext domain, which provides privacy and speed up vote tally in an e-voting scheme. Linkable ring signature allows the system distinguishes the unauthorized voters while at the same time hides the identity of voter among all legitimate voters. However, efficiency is a common issue from these approaches. Mix-net and homomorphic encryption have high computational costs due to the fact that the participants are required to generate additional proof of correctness for their operations. In linkable ring signature scheme, at least one of the operations, the $sign$ and $verify$ algorithm, is linear to the anonymity set. Thus, it is not suitable for voting system with a large number of legitimate voters.

In addition, the complexity of the proof generation also limits homomorphic encryption to only support small number of choices in a voting. Enabling an authority to certify voters' ballots in a privacy-preserving manner and with constant cost, blind signatures are usually adopted in e-voting systems. Nevertheless, the system relies on a single trust party. Consequently, a compromised authority is able to create unlimited fake ballots, which is a huge problem.

The aforementioned approaches can achieve different security requirements and could be combined to establish a secure e-voting system. In addition to the cryptographic approaches using in the voting process, a secure e-voting system requires a trusted public bulletin board for publishing the final result. In order to construct a public bulletin board which is trusted by all participants, integration with blockchain technology can be one of the possible solutions. McCorry et al. [72] introduced a new decentralized e-voting system in 2017. The system built on Ethereum network and achieved self-tallying. Due to the concern of platform dependency, Yu et al. [99] also proposed a platform-independent secure blockchain-based voting system recently. Although these solutions provided self-tallying protocol for better data integrity during vote counting, they still depend on a single trusted party during voters registration and not efficient enough for practical scenario.

In our study [36], we propose a novel and efficient e-voting system with reduced trust. We adopted blind signature, public-key encryption, blockchain and threshold cryptographic techniques to implement our secure e-voting system. By applying threshold blind signature scheme, it increases the overall system efficiency while at the same time mitigates the risk of authorities being compromised. In comparison, it is more efficient than linkable ring signature, especially when there is a large number of legitimate voters. Compared to a typical blind signature, at least a threshold number t of authorities, not only a single authority, must sign together for grant-

ing a right to an eligible voter during registration phase. To achieve fairness, all ballots are encrypted by ElGamal encryption, and the decryption key is distributed through threshold techniques so that it requires a threshold number t^* of authorities to decrypt the ballots and perform counting together. Similarly, blockchain technology is adopted to further reduce the trust between participants, all messages in the system are required to transmit over a blockchain network. Besides, our system achieves the fast-tallying property as homomorphic encryption does, while our system has no limit on the number of ballots choices. Finally, we conduct an experiment to measure the efficiency of the system. The result shows that the efficiency is competitive.

1.2 Message Authentication for Vehicular Networks

Transportation is another very important social function in our life. Due to the rapid development of wireless technologies, intelligent transportation system (ITS) is proposed in the metropolitan cities for enhancing the safety and efficiency of transportation system. Vehicular ad-hoc network (VANET), as a key component in ITS, facilitates vehicles on the roads to share information with each other in real-time. Since the wireless nature of VANET, adversary could easily monitor, alter and forge messages over the wireless network. Therefore data authenticity of messages is the main security requirement in VANET. Besides, the privacy of message senders, the vehicles, should be protected. However, the trusted authority is responsible for investigations and revocations of any malicious or misbehaving vehicle, trusted authority should be capable of revealing the real identity of message sender and vehicle when needed.

In recent years, certificateless signature schemes are being used to build a secure

and privacy-preserving VANET [41, 63, 25, 52]. But they suffer from two major issues. Firstly, efficiency in message authentication is low. Either the scheme implemented by map-to-point hash functions or bilinear pairings has high computational cost. On the contrary, the communication device of vehicles, also known as on-board units (OBUs), only have limited computational power. Secondly, the existing schemes lack efficient revocation mechanisms. In order to achieve revocation, one of the existing approaches is to introduce an online security mediator (SEM) [49]. SEM is used to manage part of the private key and issuing tokens to vehicles. But this approach adds additional communication cost and requires setting up a secure channel between SEM and vehicles. Another approach requires a key generation center (KGC) to maintain a revocation list [101], but its computational complexity is linear, which is not scalable if there is many of vehicles in the networks.

In our research [59, 60], we aim to improve the overall efficiency and reducing the trust of message authentication scheme for VANET. Our proposed scheme has a number of improvements on existing schemes. Firstly, we employed an efficient signature scheme to the protocol. Our scheme is implemented without the computation intensive elements, such as bilinear pairing. Also, part of the signature generation steps can be precomputed offline during the vehicle idle time, which can increase the overall efficiency. Secondly, considering the messages may overload the OBU when the vehicles' density is too high, we also introduced a roadside unit (RSU) assisted approach. The RSU, with relatively more computational capacity, executes a batch verification algorithm of the signature scheme, then it converts the result to cuckoo filter [31] and broadcasts it to the nearby vehicles. OBU can retrieve the validity of those signatures by very efficient assertion tests, instead of verifying each signature by itself.

In our scheme, OBU on a vehicle is responsible to sign and verify messages on

VANET. A full private key is formed by initial partial secret key isk, time key and a random secret value. KGC would maintain a revocation list and refresh the time key periodically. Vehicles would not receive any new time key if they are revoked. We further improve the efficiency for key update process by adopting KUNodes algorithm, which is a node selection algorithm. As a result, revocation achieves logarithmic complexity to the number of vehicles in the VANET. Besides, the time key is directly updated by KGC and broadcasted through public channel, no additional trusted SEM or secure channel is required. Since the VANET relies on a secure KGC, we introduce blockchain to record the revocation activities of KGC, which can provide revocation accountability and enhance the transparency and accountability of the KGC.

1.3 Supply Chain Management System

Due to globalization, manufacturing process becomes more complex nowadays. The study of supply chain management system has gained considerable amount of interest over the years. It is especially important in food and drug manufacturing industries, since it may cause huge impact on human life and they have to be regulated. However, due to the complex logistic flows in the supply chain, it dramatically increases the difficulty for government to enforces regulation and for customers to determine counterfeit goods. Thus, supply chain management system is essential to providing traceability for the products and accountability for the manufacturing and logistic processes.

In supply chain management, one of the biggest challenges is the communications among a huge number of stakeholders, including suppliers, manufacturers, wholesalers, distributors and retailers. This kind of large-scale inter-organizational

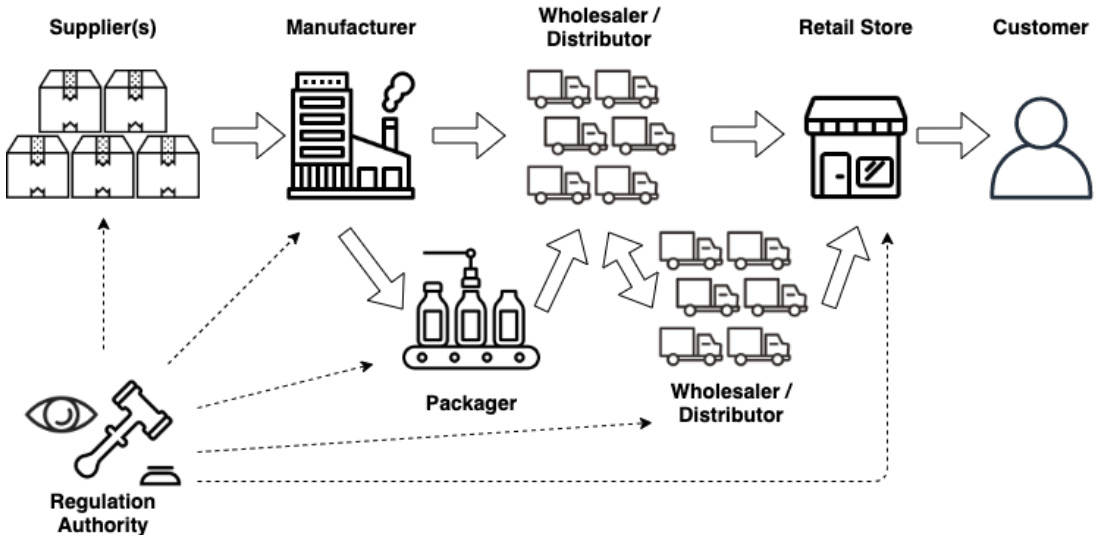


Figure 1.1 An example of a supply chain in regulated manufacturing industry

systems (IOSs) was originally considered as a market strategic decision for reducing transaction cost [70]. Choudhury [17] investigated the problem and classified three patterns of IOSs. Based on this, Benchini et al. [5] modeled the behavioral patterns and proposed a supply chain traceability system based on electronic data interchange (EDI). However, when the system is designed for achieving cost efficiency, the traceability is highly dependent on the trust of all participants. As such, there is a risk of data being tampered during investigation by regulation authority. Figure 1.1 provides an example of a supply chain in regulated manufacturing industry. Ensuring data integrity in each step of the supply chain can be challenging for regulatory authority.

Blockchain-based supply chain systems are being proposed to address the issue. Furthermore, according to a recent work [35], blockchain-based solutions have a positive impact on the customer trust, especially on unfamiliar retailers. As a new distributed ledger technology, blockchain is known for its decentralization and immutability properties. As such, it appears to be an effective solution for regulation authorities and general customers to detect data tampering in the above-mentioned

data integrity issue in supply chain systems. Early efforts [42, 43] discussed how blockchain can be applied to food and drug traceability. However, these proposals are at the conceptual state and its efficiency are evaluated based on simulations or prototyping. Recently, comprehensive solutions [85, 95, 73] were developed. However, these solutions focus mainly on products tracking and do not consider other essential features [14] of practical relevance, such as digital identity management and quality monitoring. Moreover, these solutions do not consider the existing logistic behavioral patterns of real-world scenarios, where supply chain flows form a complex directed acyclic graph (DAG). To the best of our knowledge, only [43] and [95] in the existing blockchain-based solutions consider a supply chain with a DAG structure. Besides, none of them explicitly implements multi-hop tracing and considers how to trace product without unique identifier (UID). These features are essential in anti-counterfeiting scenarios in reality.

We worked closely with drug manufacturers and wholesalers in China to develop a blockchain-based supply chain system for real-world usage. We realize that the common assumption in existing solutions may not apply to drug manufacturing. Specifically, many existing solutions assume the existence of a UID for each product, which is often unavailable in practice. The Chinese government has strict regulations on the production of packing materials of drug products [75]. Each modification of the packing contents, including description text, batch number, serial number and expiry date, requires an explicit approval by relevant authorities. Indeed, the smallest unit of "tracking" is often a product batch (a product batch is a group of identical items produced together; each batch goes through the same stages in the production process). In case of fault, it is often the smallest unit to be recalled. Besides the pharmaceutical industry, we note that batch production is also common in the production of electrical goods, clothing, and fast food, etc. Furthermore,

there are numerous procedures and regulations [89, 76] for the practitioners to follow during the production and logistic processes. A robust system for such settings is required to record all essential activities, timestamps and participants for auditing purposes. However, there is no existing solution considering this scenario.

In our study [56], we proposed a blockchain-based supply chain management system for traceability, regulation and anti-counterfeiting, which has a number of improvements when compared with existing proposals. Firstly, we proposed three logarithmic traceability algorithms and event-logs-query-free approach to enhance supply chain tracing efficiency for anti-counterfeiting scenario. Secondly, considering the needs of regulation enforcement, we also introduced supply chain quality management in our system. Apart from the logistic information in a supply chain, all activities of regulatory procedures, such as manufacturing logs, are recorded in our system. In addition, actor-based authentication is employed for accountability of the quality management. Thirdly, our proposed smart contracts and traceability algorithms support all four basic logistic behavioral patterns and multi-hop routing, which can handle the network graph created by logistic activities. Furthermore, composed identifier is used in one of the tracing algorithm for addressing the scenario if unique identifier does not exist. Finally, based on our evaluation of the implementation and comparison of existing works, the system is efficient for real-world settings and provides a more complete solution to supply chain management.

We further enhance our system by introducing a threshold twisted ElGamal decryption scheme for maintaining better privacy and auditability. The scheme enables participants to encrypt their sensitive information, such as price and quantity of sale, before recording them in blockchain. Ciphertext could be decrypted only if t decryption keys owners, typically regulation authorities, working together. The scheme also supports homomorphic addition, and it is zero-knowledge-proof (ZKP)

friendly. Thus, it can improve privacy and security of the system.

1.4 Related Works

1.4.1 Related Works of Electronic Voting System

Four cryptographic approaches are commonly adopted in the e-voting system for achieving anonymity and verifiability, namely, mix-net [12],homomorphic encryption [81], linkable ring signature [64] and blind signature [13]. Recently, blockchain was also adopted in e-voting system to address the trust issue on vote result publishing.

Mix-net was firstly proposed by Chaum [12] in 1981. It uses a number of mixes to shuffle and remask the ballots in e-voting system. As a result, it breaks the linkage between the voters' identities and their ballots. In 1995, Sako and Kilian [84] developed a concept called universal verifiability, which requires each mix to generate proof for the complete correctness of its operation, to reduce the trust assumption. Afterward, a new technique called randomized partial checking mix-net was proposed by Jakobsson et al. [46] for building a robust e-voting systems in 2002. Instead of requiring complete correctness proofs, it uses randomized partial checking. However, the proofs are still hard to generate and verify due to its computation complexity, and its first implementation had not been made until [34]. The efficiency of mix-net based e-voting scheme was further improved in [33], although the proofs generation is still a bottleneck.

Homomorphic encryption was proposed by Rivest el al. [81] in 1978. It is another widely adopted approach for privacy-preserving in e-voting system. It allows teller aggregates voters' encrypted ballots in ciphertext domain during vote counting. To

be more specifically, Paillier encryption [83, 96] and ElGamal decryption [54, 57] are adopted in existing e-voting systems. Based on the models introduced in [18, 21, 6], Cramer et at. [21] proposed a distributed scheme for multi-authorities in 1996. It is further enhanced in [20] and became the first optimal scheme for large scale e-voting system by means of threshold cryptographic techniques. Nevertheless, this scheme only supports binary (yes or no) voting and requires additional ZKP for the correctness of each undecrypted ballot processing. In general, it is hard to implement multiple choices e-voting systems with homomorphic encryption approach due to the computational complexity of the proof generation, while it is more suitable for e-voting system with small number of candidates when comparing with mix-net approach [1]. However, it can be combined with mix-net [87] as a mix-type e-voting scheme to improve the overall efficiency.

Linkable ring signature was proposed by Liu et al. [64] in 2004. It can be used in e-voting system to ensure the authenticity and anonymity of ballots. The signature size is originally linear to the anonymity group size. In spite of the improvement that signature size can be reduced to constant size in some schemes [27, 93], the time complexity of either the signature generation or verification algorithm is still linear to the group size. This makes linkable ring signature not suitable for e-voting system with large number of legitimate voters. On the contrary, blind signature [13] is another commonly adopted signature scheme which is efficient and independent to the number of voters. It is often applied in voter registration and vote casting. It can prove the ballot authenticity while still protect the voter privacy. In 1992, Fujioka et al. [32] introduced blind signature into e-voting system. In addition, Okamoto [77] proposed the first practical receipt-free e-voting scheme for large-scale election in 1997. Although blind signature enables e-voting system to support large number of voters, it requires a trusted authority to be the signer. If the signer is compromised

by attacker, it will be able to counterfeit as many ballots as its wish. To address the issue, Juang et al. [50] proposed an e-voting scheme which supports distributed authorities by means of applying threshold cryptographic techniques [26]. Recently, Mateu et al. [71] proposed a new method which achieved public verifiability in their threshold e-voting system.

Recently, blockchain technology is adopted to construct e-voting system. In 2015, Zhao et al. [102] proposed the first blockchain-based e-voting system based on Bitcoin [74]. This system adds ZKP to fulfill additional security requirements, but it only supports binary vote. In 2017, Tarasov [91] proposed a blockchain-based e-voting system based on the Zcash payment protocol [40]. The anonymity of voters can be ensured by Zcash protocol itself, however it assumes there is a trusted third-party to guarantee the correctness of voting result. In the same year, McCorry [72] proposed the first decentralized self-tallying e-voting protocol using Ethereum [10] smart contract, with the constraint that it can only work with binary vote and 50 voters as maximum. More recently, Yu et al. [99] pointed out that existing blockchain-based e-voting systems highly depend on the underlying blockchain protocol which caused correctness and receipt-freeness were hard to achieve. So they demonstrated the construction of a secure blockchain-based e-voting system without any platform dependencies. In 2019, Lin and Zhang [62] proposed a new "efficient one-out-of-T" ZKP to construct a blockchain-based self-tallying e-voting protocol. Although the efficiency of prove and verify operations was improved compared to typical ZKPs, it was still linear with respect to the number of voters, and therefore only suitable for small-scale anonymous voting.

1.4.2 Related Works of Message Authentication for Vehicular Networks

Due to the nature of VANET, both security and efficiency of the network is important to the drivers' safety. In order to secure the message being broadcasted in the VANET, message authentication is required. Because of the performance consideration, certificateless signature scheme is commonly used in VANET. In 2003, Al-Riyami and Paterson [82] proposed the certificateless public key cryptographic (CL-PKC). In the next year, Lee [100] introduced a generic construction of certificateless public key signature (CL-PKS) scheme by transforming from any identity-based signature scheme. In 2007, Au et al. [4] defined a new security model for CL-PKS, which takes a new malicious KGC attack into consideration. In general, many CL-PKS relies on bilinear pairings, which has relatively high computational cost when comparing with scalar multiplication over elliptic curve (EC) group. So He et al. [37] proposed the first pairing-free CL-PKS in 2012. More recently, Yeh et al. [98] proposed a CL-PKS scheme for the smart object in the internet-of-things (IoT) network. However, Jia et al. [48] discovered the security flaws of it, thus they proposed another CL-PKS solution for IoT deployment.

In order to support revocation in CL-PKC, there are several approaches. The first approach is mediated certificateless scheme which is proposed by Ju et al. [49] in 2005. In this scheme, the private key is divided into two parts, the SEM can manage the revocation by rejecting token requests from revoked users. However, this approach requires SEM to maintain secret values of all users, which increases the risk to be targeted by attackers. The second approach is proposed by Sun et al. [90] in 2014. The private key in this scheme is composed by several parts. One of the parts, time key, is refreshed and distributed by KGC periodically. To perform revocation,

KGC should refuse the key refresh for any revoked users. In 2015, Zhang and Zhao [101] proposed to use revocation list to construct an efficient revocable CL-PKS scheme. Recently, Du et al. [29] proposed an outsourced revocable certificateless signature (ORCLS) scheme, which outsources the computational expensive process to the cloud.

Considering the large number of messages being broadcasted in the VANET, batch signature verification is adopted in some works. In 2015, He et al. [38] proposed an efficient identity-based VANET authentication scheme which adds batch signature verification as a component. In addition, bloom filter [7], a probabilistic data structure for set-membership assertion, is added to assist the batch verification [16, 55]. In 2016, Malhi and Batra [68] proposed an authentication framework which uses two bloom filters to perform MAC address and pseudonym check. More recently, Cui et al. [23] introduced cuckoo filter, another set-membership assertion structure, to the message authentication scheme for VANET. However, Limbasiya [61] stated that [23] still have the key escrow problem from the identity-based cryptographic.

Recently, blockchain technology is being adopted to VANET, because it can provide decentralization, high availability and data integrity preserving. In 2018, Malik et al. [69] integrated blockchain to VANET for reducing dependence of KGC. The proposed system can reduce communication overhead between KGC and RSUs, also it improves the efficiency of revocation. In 2019, Ali et al. [2] proposed a blockchain-based CL-PKS scheme, which uses two blockchains to store the pseudo-identities of revoked and non-revoked users. As a result, it achieves the revocation transparency over the networks.

1.4.3 Related Works of Supply Chain Management System

The history of supply chain management system can be traced back to 1985 when Cash [11] reviewed on how inter-organizational systems (IOSs) influenced the manufacturing industry. In its early stage, supply chain management systems are designed for reducing transaction cost in the value-added chain of manufacturing [70]. In 1997, Choudhury [17] proposed the three common patterns of IOSs, namely, electronic monopolies, multilateral IOSs and electronic dyads. The electronic dyads pattern is a type of peer-to-peer (P2P) network, where each organization directly connects with each of its business partner through independent electronic links. It is common that electronic data interchange (EDI) is adopted as a message standard in electronic dyads.

The word traceability is added into ISO8420:1994 [45] for quality management and quality assurance. In 2002, van Dorp [28] summarized the development of existing traceability systems. In 2008, Benchini et al. [5] proposed a new supply chain system and modeled the behavioral patterns in logistics. Since then, many works [92, 39] focused on the food and drug supply chain system. Recently, due to the improvement in wireless communication, radio frequency identification (RFID) and the internet-of-things (IoT) are introduced to supply chain systems [53, 65, 39] as well. Although there are more data to assist the quality management in supply chain system, there is a trust issue on these centralized systems. Data tampering can be easily conducted if a malicious organization wants to circumvent government regulations.

Recently, blockchain-based supply chain systems were proposed to address this issue. In 2017, Chen et al. [14] proposed a blockchain-based supply chain quality management framework. They described different modules required in the system, including digital identity management, quality monitoring and control, logistics

planning and demand analysis. In 2018, Hua et al. [42] proposed a system for agricultural products. However, these systems remain conceptual and no actual implementation is conducted. Drugledger [43] is a blockchain-based system, extended from Bitcoin [74]. The system provides traceability and identity management. However, its design relies on the unspent transaction output (UTXO) model and the proposed data structure does not cater for tracking the internal process of the organization down to the employee level. Since the personnel in the manufacturing process may change, it is hard to define the owner of a transaction output. In 2019, Aniello et al. [3] proposed to adopt physically unclonable function (PUF) in blockchain-based supply chain system for anti-counterfeiting purpose. This technique requires special hardware, which may hinder its deployment in practice. More recently, a blockchain token based solution was proposed [95]. It used an Ethereum [10] token standard (ERC721 [30]), known as non-fungible token (NFT), to keep track of various logistic activities. However, the solution misuses ERC721 and adds quantity field to smart contract, where quantity field is removed from ERC721 intentionally. PharmaCrypt [85] is another recent blockchain-based drug supply chain system. It requires the manufacturer to scan barcode for each product and upload the data of the shipping box to blockchain. This process is inherently inefficient and not practical when the batch size is large. In 2021, Musamih et al. [73] proposed another Ethereum-based solution with implementation details, with testing and validation. However, both [85] and [73] did not consider all logistic behavioral patterns in their design, hereby could not handle integration and division lots where it is very common in reality. Currently, most of the existing blockchain-based solutions are still in prototyping phase and most of them focus on traceability only, with less efforts spent on other essential features, like identity management, quality monitoring and privacy preserving, for supply chain systems.

1.5 Thesis Organization

In the next chapter, Chapter 2, the preliminaries of the report is given. Followed in Chapter 3, we present our contributions to blockchain-based threshold e-voting system. In Chapter 4, we present our design of efficient message authentication with revocation transparency using blockchain for VANET. After that, in Chapter 5, we present our contributions to blockchain-based supply chain system for traceability, regulation and anti-counterfeiting. Finally, in Chapter 6, we give a conclusion of this thesis.

CHAPTER 2

PRELIMINARY

In this chapter, we give the preliminaries which will be used in the later chapters. We aim to show some background information on the notation, definitions and assumptions of some cryptographic primitives, algorithms and techniques which are used in the following chapters. In particular, we will introduce the computational assumptions used in our schemes. We also give the syntax and definitions of *threshold blind signature* scheme, *threshold ElGamal decryption* scheme, *Twisted ElGamal encryption* scheme, *threshold twisted ElGamal decryption* scheme, *Cuckoo filter* algorithm and *KUNodes* algorithm. Lastly, we will describe the blockchain technology and models we used in our works.

2.1 Notations

We use $x \xleftarrow{\$} S$ to denote that x is randomly and uniformly picked form set S. We use $y \leftarrow A(x)$ to denote that output y is generated from algorithm A with input x. We use $A(x) \rightarrow y$ to denote that algorithm A with input x will output y.

We let $\mathbb{G}$ be a cyclic group of prime order q with generator P. We let $\mathbb{Z}_q$ denote integer set $\{0, 1, \ldots, q-1\}$ while $\mathbb{Z}_q^*$ denote integer set $\{1, 2, \ldots, q-1\}$. We let $\mathbb{PG} = (\mathbb{G}, \mathbb{G}_T, q, P, \hat{e})$ to denote that a pairing group $\mathbb{PG}$ composes of two groups $\mathbb{G}, \mathbb{G}_T$ of the same prime order q with a generator P of $\mathbb{G}$, and $\hat{e}$ denote the bilinear mapping $\hat{e} : (\mathbb{G}, \mathbb{G}) \rightarrow \mathbb{G}_T$.

We use $\oplus$ to denote the *exclusive OR* operation.

2.2 Assumptions and Hard Problems

In our study, security of our design relies on the computational complexity of three well-known cryptographic problems, namely, Computational Diffie-Hellman (CDH) Problem, Decision Diffie-Hellman (DDH) Problem and Gap Diffie-Hellman (GDH) Problem. We assume these problems are hard. The descriptions of each cryptographic problem are given below.

- **Computational Diffie-Hellman (CDH) Problem**. Given a triple of group elements $(P, P^a, P^b) \in \mathbb{G}$, where $a, b \in \mathbb{Z}_q^*$, find an element $C \in \mathbb{G}$ such that $C = P^{ab}$.
- **Decision Diffie-Hellman (DDH) Problem**. Given a quadruple of group elements $(P, P^a, P^b, P^c) \in \mathbb{G}$, where $a, b, c \in \mathbb{Z}_q^*$, decide whether $c = ab$.
- **Gap Diffie-Hellman (GDH) Problem**. Given a triple of group elements $(P, P^a, P^b) \in \mathbb{G}$, where $a, b \in \mathbb{Z}_q^*$, and a DDH oracle, which answers DDH problem, find an element $C \in \mathbb{G}$ such that $C = P^{ab}$.

2.3 Threshold Blind Signature

The (t, n)-threshold blind signature scheme [94] contains four algorithms, namely, *Setup* algorithm (TBU), $KeyGeneration$ algorithm (TBK), $SignatureGeneration$ algorithm (TBS) and $SignatureVerification$ algorithm (TBV). Its security is based on GDH hard problem. In this scheme, TBK and TBS are two interactive protocols. We denote n players in this protocol as $\{L_1, L_2, \ldots, L_n\}$.

1. *Setup* algorithm (TBU)

   With the input security parameter 1^λ, this algorithm outputs public parameters $param = (\mathbb{PG}, H)$, where $\mathbb{PG}$ is a pairing group such that $\mathbb{PG} = (\mathbb{G}, \mathbb{G}_T,$

$q, P, \hat{e})$. Let $H : \{0,1\}^* \rightarrow \mathbb{G}$ denotes a one-way function. In the following algorithms, *param* is an implicit input.

2. *KeyGeneration* protocol (TBK)

   In this interactive protocol the secret key is distributed to the n players but does not appear explicitly in the protocol. Let $\{L_i\}_{i=1}^n$ denotes the players. For each L_i computes the followings:

   (a) Randomly picks parameters $a_{ij} \xleftarrow{\$} \mathbb{Z}_q^*$ for $j = 0, 1, \ldots, t-1$ and forms polynomial $f_i(x) = a_{i0} + a_{i1}x + \ldots + a_{i,t-1}x^{t-1}$.

   (b) Computes and broadcasts $P^{a_{ij}}$ for $j = 0, 1, \ldots, t-1$ and sends $f_i(j)$ to player L_j for $j = 1, 2, \ldots, n; j \neq i$.

   (c) Player L_i will receive $f_j(i)$ from L_j for $j = 1, 2, \ldots, n; j \neq i$, L_i verifies if the equation $P^{f_j(i)} = \prod_{k=0}^{t-1} P^{a_{jk} \cdot i^k}$ holds. If the check fails, L_i broadcasts a complaint against L_j.

   (d) L_i computes the secret share $s_i = \sum_{k=1}^n f_k(i)$ and public share $Q_i = P^{s_i}$, which will be broadcasted to all other players.

   After executing the protocol, the secret key is set as $s = \sum_{i=1}^n a_{i0}$, the public key is set as $Q = P^s$, which can be computed by $Q = \prod_{i=1}^n P^{a_i 0}$.

3. *SignatureGeneration* protocol (TBS)

   This protocol allows user A to obtain a blind signature on the message m from t signers. Let $S = \{L_i | 1 \leq i \leq t\}$ denotes the set of t signers. For the ease of presentation, we use w_i to denote $\prod_{j \in S, j \neq i} \frac{i}{j-1}$. The protocol is runs as the followings:

   (a) User A randomly picks $r \xleftarrow{\$} \mathbb{Z}_q^*$ and blind the message m by computing $m' = H(m)^r$. User A sends m' to every signer $L_i \in S$.

(b) Each signer L_i computes $\sigma_i = m'^{w_i s_i}$ and sends σ_i to user A.

(c) User A validates σ_i by checking whether the equation $\hat{e}(\sigma_i, P) = \hat{e}(m'^{w_i}, Q_i)$ holds. If it does not hold, user A sends m' to the signer L_i to request a valid σ_i again. Otherwise, user A computes the signature σ on message m as $\sigma = (\prod_{i \in S} \sigma_i)^{-r}$.

4. *SignatureVerification* algorithm (TBV)

   The algorithm accepts the signature σ on the message m if the equation $\hat{e}(\sigma, P) = \hat{e}(H(m), Q)$ holds, otherwise it rejects.

2.4 Threshold ElGamal Decryption

The (t^*, n^*)-threshold ElGamal decryption scheme [36] contains four algorithms, namely, *Setup* algorithm (TEU), *KeyGeneration* algorithm (TEK), *Encryption* algorithm (TEC) and *Decryption* algorithm (TED). Its security is based on DDH hard problem. In this scheme, TEK and TED are two interactive protocols. We denote n^* players in this protocol as $\{L_1^*, L_2^*, \ldots, L_n^*\}$.

1. *Setup* algorithm (TEU)

   With the input security parameter 1^λ, this algorithm outputs public parameters $param = (\mathbb{PG})$, where $\mathbb{PG}$ is a pairing group such that $\mathbb{PG} = (\mathbb{G}, \mathbb{G}_T, q, P, \hat{e})$. In the following algorithms, $param$ is an implicit input.

2. *KeyGeneration* protocol (TEK)

   Same as the TBK protocol of the threshold blind signature scheme, each player L_i^* randomly picks $a_{ij} \xleftarrow{\$} \mathbb{Z}_q^*$ in the polynomial $f_i(x) = a_{i0} + a_{i1}x + \ldots + a_{i,t-1}x^{t-1}$ and sends $f_i(j)$ to L_i^*. The secret share of L_i^* is $s_i^* = \sum_{k=1}^{n^*} f_k(i)$. The corresponding public share is $Q_i^* = P^{s_i^*}$, which will be broadcasted. As

a result, the secret key is set as $s^* = \sum_{i=1}^{n^*} a_{i0}$ and the public key is set as $Q^* = P^{s^*} = \prod_{i=1}^{n^*} P^{a_{i0}}$. The whole algorithm can be written as $TEK \rightarrow (Q^*, s^*, Q_i^*, s_i^*)$, for $i \in \{1, 2, \ldots, n^*\}$.

3. *Encryption* algorithm (TEC)

   With the input message m and public key Q^*, randomly pick a number $k \xleftarrow{\$} \mathbb{Z}_q$ and computes the ciphertext as $C = (C_1, C_2) = (P^k, mQ^{*k})$.

4. *Decryption* protocol (TED)

   This protocol requires t^* decrypters to decrypt the ciphertext C together. Let $S^* = \{L_i^* | 1 \leq i \leq t\}$ denotes the set of t^* decrypters. The protocol is run as the followings:

   (a) Each decrypter L_i^* computes $w_i = \prod_{j \in S^*, j \neq i} \frac{j}{j-1}$ and $m_i = C_1^{-w_i s_i^*}$, then broadcasts m_i to others.

   (b) After receiving m_i, verifies if the equation $\hat{e}(m_i, P) = \hat{e}(C_1^{-w_i}, Q_i)$ holds. If it does not hold, broadcasts a complaint on L_i^*.

   (c) Finally, the decrypted message m' can be computed as $m' = C_2 \cdot \prod_{i=1}^{t^*} m_i$.

Correctness. Let $f(x)$ be a polynomial, where $f(x) = a_0 + a_1 x + \ldots + a_{t-1} x^{t-1} = \sum_{i=1}^{n^*} f_i(x)$. Therefore, s^* can be expressed as $s^* = \sum_{i=1}^{n^*} a_{i0} = a_0$. We use the Lagrange interpolation, s^* can be derived from s_i^*. The decrypted message m' can

be expressed as,

$$
\begin{aligned}
m' &= C_2 \cdot \prod_{i=1}^{t^*} m_i = C_2 \cdot \prod_{i=1}^{t^*} C_1^{-w_i s_i^*} \\
&= C_2 \cdot \prod_{i=1}^{t^*} C_1^{-\prod_{j \in S^*, j \neq i} \frac{j}{j-i} \cdot \sum_{k=1}^{n^*} f_k(i)} \\
&= C_2 \cdot \prod_{i=1}^{t^*} C_1^{-\prod_{j \in S^*, j \neq i} \frac{j}{j-i} \cdot f(i)} \\
&= C_2 \cdot C_1^{-f(0)} = C_2 \cdot C_1^{-s^*} \\
&= mP^{s^* k} \cdot P^{-s^* k} = m
\end{aligned}
$$

2.5 Twisted ElGamal Encryption

The twisted ElGamal encryption scheme [15] is a ZKP-friendly additive homomorphic public key encryption (PKE) scheme. It contains four algorithms, namely, *Setup* algorithm (EU), *KeyGeneration* algorithm (EK), *Encryption* algorithm (EC) and *Decryption* algorithm (ED). Its security is based on DDH hard problem.

1. *Setup* algorithm (EU)

   With the input security parameter 1^λ, this algorithm outputs public parameters $param = (\mathbb{G}, g, h)$, where $\mathbb{G}$ is a cyclic group such that $\mathbb{G} = (\mathbb{G}, q, P)$ and (g, h) are two random generators of $\mathbb{G}$. In the following algorithms, $param$ is an implicit input.

2. *KeyGeneration* protocol (EK)

   Randomly pick secret key $s \xleftarrow{\$} \mathbb{Z}_q^*$ and output public key $Q = g^s$.

3. *Encryption* algorithm (EC)

   With the input message m and public key Q, randomly pick a number $r \xleftarrow{\$} \mathbb{Z}_q$ and computes the ciphertext as $C = (C_1, C_2) = (Q^r, h^m g^r)$.

4. *Decryption* protocol (ED)

   With input secret key s and ciphertext C, compute $R = C_1^{s^{-1}}$ and $m' = C_2 R^{-1} = h^m$, where m could be recovered by table lookup when m is small (i.e. 32-bits).

Additive homomorphic. The scheme allows to perform addition on ciphertext domain, such that $ED_s(EC_Q(m_1) + EC_Q(m_2)) = m_1 + m_2$.

2.6 Threshold Twisted ElGamal Decryption

The (t', n')-threshold twisted ElGamal decryption scheme contains four algorithms, namely, *Setup* algorithm (TTU), *KeyGeneration* algorithm (TTK), *Encryption* algorithm (TTC) and *Decryption* algorithm (TTD). Its security is based on DDH hard problem. In this scheme, TTK and TTD are two interactive protocols. We denote n' players in this protocol as $\{L'_1, L'_2, \ldots, L'_n\}$.

1. *Setup* algorithm (TTU)

   With the input security parameter 1^λ, this algorithm outputs public parameters $param = (\mathbb{PG})$, where $\mathbb{PG}$ is a pairing group such that $\mathbb{PG} = (\mathbb{G}, \mathbb{G}_T, q, g, \hat{e})$ and h is a random generator of $\mathbb{G}$. In the following algorithms, $param$ is an implicit input.

2. *KeyGeneration* protocol (TTK)

   In this interactive protocol the secret key is distributed to the n' players and KGC assists the generation of the system shared public key. Let $\{L'_i\}_{i=1}^{n'}$ denotes the players. The interactions between n' players are described as follows:

   (a) Each player L'_i conducts the following computations.

i. Randomly picks parameters $a_{ij} \xleftarrow{\$} \mathbb{Z}_q^*$ for $j = 0, 1, \ldots, t'-1$ and forms polynomial $f_i(x) = a_{i0} + a_{i1}x + \ldots + a_{i,t'-1}x^{t'-1}$.

ii. Computes and broadcasts $g^{a_{ij}}$ for $j = 0, 1, \ldots, t'-1$ and sends $f_i(j)$ to player L'_j for $j = 1, 2, \ldots, n'; j \neq i$.

iii. Player L'_i will receive $f_j(i)$ from L'_j for $j = 1, 2, \ldots, n'; j \neq i$, L'_i verifies if the equation $g^{f_j(i)} = \prod_{k=0}^{t'-1} g^{a_{jk} \cdot i^k}$ holds. If the check fails, L'_i broadcasts a complaint against L'_j.

iv. L'_i computes the partial secret $s'_i = \prod_{k=1}^{n'} f_k(i)$ and public share $Q'_i = g^{s'_i}$, which will be broadcasted to all other players. Then sends a_{i0} to KGC.

(b) After KGC receives all a_{i0} from L'_i for $i = 1, 2, \ldots, n$, KGC computes $s' = (\sum_{i=1}^{n} a_{i0})^{-1}$ and $Q' = g^{s'}$, then broadcast Q'.

After executing the protocol, the system shared public key and private key is set as Q' and s' respectively, which the decryption key s'^{-1} is distributed to the n' players.

3. *Encryption* algorithm (TTC)

With the input message m and public key Q', randomly pick a number $r \xleftarrow{\$} \mathbb{Z}_q$ and computes the ciphertext as $C = (C_1, C_2) = (Q'^r, h^m g^r)$.

4. *Decryption* protocol (TTD)

This protocol requires t' decrypters to decrypt the ciphertext C together. Let $S' = \{L'_i | 1 \leq i \leq t'\}$ denotes the set of t' decrypters. The protocol is run as the followings:

(a) Each decrypter L'_i computes $w_i = \prod_{j \in S', j \neq i} \frac{j}{j-1}$ and $m_i = C_1^{w_i s'_i}$, then broadcasts m_i to others.

(b) After receiving m_i, verifies if the equation $\hat{e}(m_i, g) = \hat{e}(C_1^{w_i}, Q_i')$ holds. If it does not hold, broadcasts a complaint on L_i'.

(c) Finally, compute $m' = C_2 \cdot (\prod_{i=1}^{t'} m_i)^{-1} = h^m$ and the original message m could be recovered by table lookup when m is small (i.e. 32-bits).

Correctness. We use the Lagrange interpolation to derive s' from s_i' and h^m can be expressed as,

$$\begin{aligned} m' &= C_2/(\prod_{i=1}^{t'} m_i) = C_2/(\prod_{i=1}^{t'} C_1^{w_i s_i'}) \\ &= C_2/(\prod_{i=1}^{t'} C_1^{\prod_{j \in S', j \neq i} \frac{j}{j-i} \cdot \sum_{k=1}^{n'} f_k(i)}) \\ &= C_2/(\prod_{i=1}^{t'} C_1^{\prod_{j \in S', j \neq i} \frac{j}{j-i} \cdot f(i)}) \\ &= C_2/(C_1^{f(0)}) = C_2/(C_1^{s'^{-1}}) \\ &= h^m g^r/(g^{s' r \cdot s'^{-1}}) = h^m \end{aligned}$$

2.7 Cuckoo Filter

Cuckoo filter [31] is a data structure for approximate set membership test. It is a variant of cuckoo hash table [78]. It has high query performance with low space complexity which also supports dynamic item addition and deletion. The basic construction of cuckoo hash table and cuckoo filter are illustrated in Figure 2.1. It contains three algorithms namely, *insert*, *delete* and *query*, the details are described in the following.

The fingerprints of items are stored in a hash tables. When we perform the set membership check, *query* algorithm search for the fingerprint of the searching item. If the fingerprint is found, it represents the searching item exists and vice versa. As

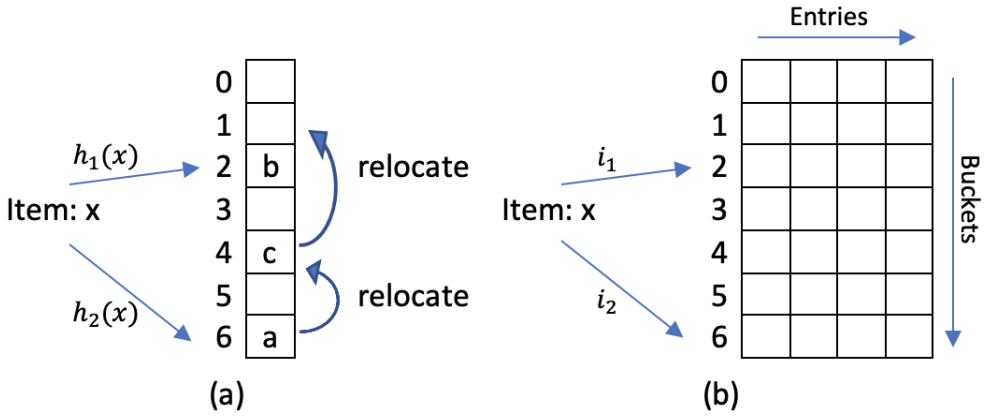


Figure 2.1 (a) Insertion and relocation of cuckoo filter (b) The hash table structure of cuckoo filter

Figure 2.1(a) shown, each item x can have two candidate buckets for the fingerprint storage. The indexes of these buckets are determined by two hash functions. During insertion, the fingerprint of item is stored to one of these candidate buckets when it is empty. If both buckets are filled, one of the existing fingerprint a in the two buckets is kicked out and be replaced by the new fingerprint of x. This triggers a relocation and re-insertion of the kicked out fingerprint a as shown in Figure 2.1(a). This relocation process can trigger multiple times during one insertion.

Cuckoo filter composes of a set of cuckoo hash tables, each bucket allows storing multiple entries. In our construction, the indexes of the candidate buckets i_1 and i_2 are calculated by $i_1 = hash(x)\ mod\ M$ and $i_2 = (i_1 \oplus hash(Fingerprint(x)))\ mod\ M$. The overall structure of cuckoo filter is illustrated in Figure 2.1(b). The formal specifications of the algorithms are listed in Algorithm 1 and Algorithm 2.

Algorithm 1 Insert(x)

$f \leftarrow Fingerprint(x)$
$i_1 \leftarrow hash(x)\ mod\ M$
$i_2 \leftarrow (i_1 \oplus hash(f))\ mod\ M$
if bucket[i_1] or bucket[i_2] has an empty entry **then**
    add f to that bucket
    return $Done$
else
    $i \xleftarrow{\$} \{i_1, i_2\}$
    for $n \leftarrow 1$ to $MaxNumKicks$ **do**
        randomly select an entry e from bucket[i]
        swap f and the fingerprint stored in entry e
        $i \leftarrow i \oplus hash(f)$
        if bucket[i] has an empty entry **then**
            add f to bucket[i]
            return $Done$
        end if
    end for
    return $Failure$
end if

Algorithm 2 Query(x)

$f \leftarrow Fingerprint(x)$
$i_1 \leftarrow hash(x)\ mod\ M$
$i_2 \leftarrow (i_1 \oplus hash(f))\ mod\ M$
if bucket[i_1] or bucket[i_2] has f **then**
    return $True$
else
    return $False$
end if

2.8 Binary Tree and KUNodes Algorithm

$KUNodes$ algorithm [8] is a scalable and efficient algorithm works on a binary tree BT. The algorithm can be expressed as $KUNodes : (BT, RL, t) \rightarrow Y$. It computes the minimum set Y of nodes which does not contain any descendants of revoked user. Thus, the KGC can perform key update on time t only to non-revoked users with minimum computation power. In the binary tree BT, a leaf node represents a

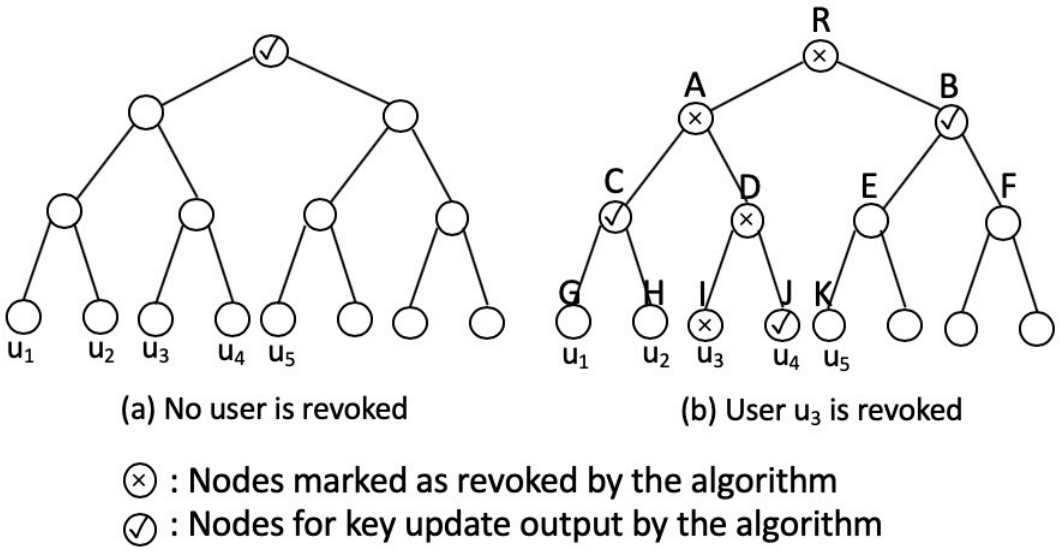


Figure 2.2 The KUNodes algorithm

user and a parent node represents all its children in that subtree. We denote v as a node and *root* as the root node of BT. v_l and v_r denote the left and right child of v if v is not a leaf node. $Path(v)$ denotes the inclusive set of nodes connecting v and *root*. We denote RL as the revocation list, where it contains a list of revoked user's leaf node v_i and its revocation or key expiration time t_i. Figure 2.2 shows the overview of $KUNodes$ algorithm in revocation.

As in Figure 2.2 (a), when *root* is returned from the algorithm, which means all nodes in BT are still valid and all users are non-revoked. Refer to the example of Figure 2.2 (b), suppose users $\{u_1, u_2, u_3, u_4, u_5\}$ correspond to nodes $\{G, H, I, J, K\}$ in BT. Revocation of user u_3 means nodes $\{R, A, D, I\}$, which can be written as $Path(u_3)$, will mark as revoked. The minimum set of nodes which does not contain any descendants of revoked user is $\{C, J, B\} \leftarrow KUNodes(BT, RL, t)$. The complexity of key update is reduced from linear to logarithmic when this algorithm is adopted. The pseudocode of $KUNodes$ is stated in Algorithm 3.

Algorithm 3 KUNodes(BT, RL, t)

```
1: X, Y ← φ
2: for all (v_i, t_i) ∈ RL do
3:     if t_i ≤ t then
4:         add Path(v_i) to X
5:     end if
6: end for
7: for all x ∈ X do
8:     if x_l ∉ X then
9:         add x_l to Y
10:    end if
11:    if x_r ∉ X then
12:        add x_r to Y
13:    end if
14: end for
15: if Y = φ then
16:    add root to Y
17: end if
18: return Y
```

2.9 Blockchain Technology

A blockchain is a distributed and append-only ledger which maintains a growing list of data *blocks*. Each block contains an ordered set of *transactions* (TXs) data, and typically links to its predecessor through a cryptographic hash pointer. The machines of the blockchain network are commonly called the *nodes*. The ledger data is distributed and replicated across the network. Nodes that store a full replica of ledger data is named *full nodes*. The machines only interact with the blockchain but do not store any local copy are called *clients*. The authenticity of transactions is ensured by digital signature. Public keys of the users are commonly encoded to *addresses*, which are also used as user identifiers. Blockchain data is distributed and synchronized among all full nodes through a *consensus protocol* such as proof-of-work (PoW), proof-of-authorities (PoA) and proof-of-stake (PoS). Blocks and transactions are validated by all *validator nodes*, usually the full nodes, before being added to the

chain. Based on the distributed architecture and cryptographic techniques adopted, blockchain can provide five properties [97], namely, immutability, non-repudiation, integrity, transparency and equal rights. It helps the implementation of a secure tamper-proof system.

Besides the basic assets transfer functionality, blockchain can have *smart contract* for building generic applications. It is a programming script deployed to blockchain and executed among all network nodes. A smart contract can express conditions, iterations and complex business logic [97] which enables developers to implement programmable transactions and develop decentralized applications on top of blockchain.

Based on access control model, blockchain can be classified to *public blockchain* and *private blockchain*; the former is an open network that allows any nodes to join and participate in the consensus, and the latter is a close network where a node requires permission to be granted before joining the network. Blockchain efficiency is highly dependent on the consensus protocol employed, and it is common that permissioned blockchain can achieve higher transaction throughput. For instance, Bitcoin [74] can only support a maximum of 7 transactions per second (tps) [88]. Ethereum [10] supports around 15 tps [88] in public network, while it can reach over 100 tps in private network setting [86].

In our works [36, 60], the applications are platform independent, they are constructed by an append-only blockchain ledger. The blockchain requires only three basic functions, namely, *initialize*, *store* and *view*. In our work [56], account-based smart contract capability is required by our design, thus Ethereum [10] is adopted in our implementation.

CHAPTER 3

THRESHOLD ELECTRONIC VOTING SYSTEM

In this chapter, we will describe our blockchain-based threshold e-voting system (published in [36]). We apply several techniques including threshold blind signature scheme, threshold decryption scheme and blockchain network to construct our proposed system. Thanks to the adoption of threshold schemes and blockchain technology, our system does not rely on a single trusted third party which may be an issue of some existing e-voting system. In addition, our system supports distributed voters registration and votes tallying. To ensure our system is efficient and practical for real-life scenarios, we also give an implementation of our proposed system. The experiment shows the time complexity in voter perspective is constant, which is scalable.

Chapter Organization. In Section 3.1, we define the entities, phases and security requirements of an e-voting system. The overview of our design and the details of construction are described in Section 3.2. The analysis of our system security is conducted in Section 3.3. Lastly, we give our implementation details and performance results in Section 3.4.

3.1 Definitions

Entities. An e-voting system contains the following three entities, namely, *Voter*, *Organizer* and *Teller*. Descriptions of the entities are given below.

- **Voter**. The entity casts a ballot. Each eligible voter can only cast one ballot during a voting.
- **Organizer**. The entity which is responsible for granting rights to voters.

- **Teller**. The entity counts the ballots after voting.

Phases. An e-voting system contains the following five phases, namely, *Setup*, *KeyGen*, *Register*, *VoteCasting* and *VoteCounting*.

- **Setup**. Voters, organizers and tellers confirm the common parameters of the system.
- **KeyGen**. Voters, organizers and tellers generate their own key pairs, i.e. the secret key and public key, respectively.
- **Register**. Voters get voting rights from organizers through registering.
- **VoteCasting**. Each voter casts their ballot through the system.
- **VoteCounting**. Tellers count the ballots and publish the voting result.

Security Requirements. The required security properties of an e-voting system are described as follows.

- **Verifiability**. Voters can verify whether their ballots are counted correctly or not.
- **Eligibility**. Only eligible voters, who had registered their identities and got permissions from organizers, can vote. Each eligible voter is allowed to vote once and only valid ballots will be counted in the system.
- **Fairness**. The result is counted and reviewed only at the end of the voting. In particular, no intermediate result of the voting, or trends, can be inferred during the voting since it might affects decision of voters before they cast their vote. In other words, fairness guarantees that the choice of the voters will not be influenced by intermediate information available from the voting system.
- **Anonymity**. Voters can cast their ballots anonymously. In particular, no one can reveal the ownership of a ballot.

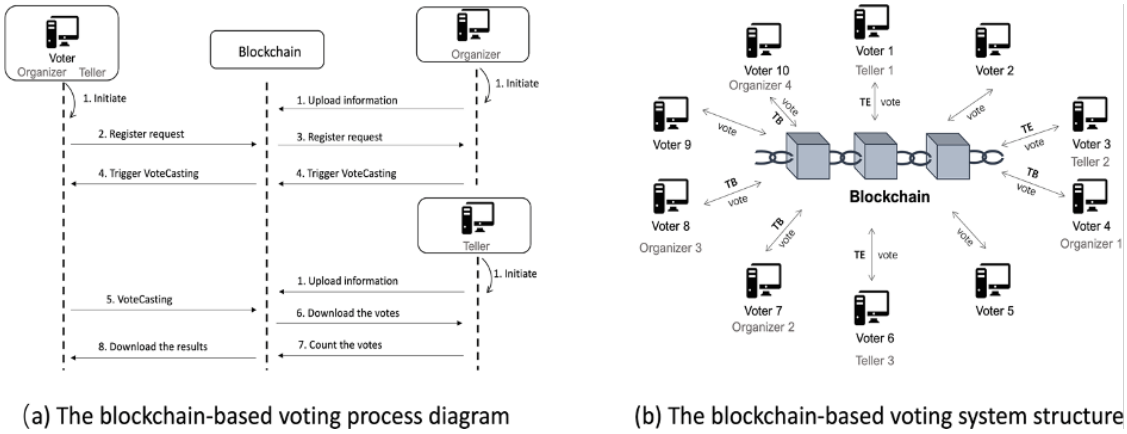
3.2 Our Construction

3.2.1 Design Overview

We apply two cryptographic techniques, i.e. (t, n)-threshold blind signature scheme $\mathcal{TB}$ and (t^*, n^*)-threshold ElGamal decryption scheme $\mathcal{TE}$, and combine them with blockchain technology to construct our system. The conceptual design is outlined as follows.

All eligible voters are required to register in the system. Firstly, voter generate a one-time keys pair, then sends a registration request to organizers. After verification of the voter's eligibility, t organizers compute $\mathcal{TB}$ together for generating a threshold blind signature on voter's public key, which grants the voting right of the eligible voter. When an eligible voter casts a ballot in the system, it is encrypted by threshold ElGamal scheme $\mathcal{TE}$, whose encryption key is publicly available. After that the encrypted ballot, signature issued by organizers, voter's public key and signature of ballot are broadcasted to blockchain. During vote counting, t^* tellers validate the signatures and execute threshold decryption of $\mathcal{TE}$ together. The decrypted ballots is then being published to blockchain. The overview of our design is shown in Figure 3.1.

The system is secure and all ballots are guaranteed to be cast by legitimate voters as long as there are less than t malicious organizers and less than t^* malicious tellers in the system. The linkage of a ballot and its voter is protected by the blindness property of $\mathcal{TB}$, so voters can vote anonymously. Since each voter can only register once, and the signature on voter's public key can be only obtained from organizers, signing ballots with same public key means double vote. The system can easily detect double vote and discard the invalid ballots. In addition, the threshold



(a) The blockchain-based voting process diagram (b) The blockchain-based voting system structure

Figure 3.1 The overview of our blockchain-based e-voting system

techniques adopted are both one-round protocols with low complexity, which make the system efficient and practical.

3.2.2 Construction Details

To construct our platform independent blockchain-based e-voting system, we adopt the following approaches, including blockchain technology, (t, n)-threshold blind signature scheme $\mathcal{TB} : (TBU, TBK, TBS, TBV)$, (t^*, n^*)-threshold ElGamal decryption scheme $\mathcal{TE} : (TEU, TEK, TEC, TED)$ and signature scheme $\Pi^{sig} : (KeyG, Sign, Verify)$.

We assume there are l eligible voters, n organizers and n^* tellers. V_i, L_i and T_i denote an eligible voter, an organizer and a teller respectively. $\mathcal{V}$, $\mathcal{L}$ and $\mathcal{T}$ denote set of voters, set of organizers and set of tellers respectively, where $\mathcal{V} = \{V_1, V_2, ..., V_l\}$, $\mathcal{L} = \{L_1, L_2, ..., L_n\}$ and $\mathcal{T} = \{T_1, T_2, ..., T_{n^*}\}$. In general, organizers and tellers are played by eligible voters.

There are five phases in the system, namely, *Setup*, *KeyGen*, *Register*, *VoteCasting* and *VoteCounting*. The details are listed as follows.

1. **Setup**: On input security parameter 1^λ, it outputs $param = (\mathbb{G}, q, P, H)$ and broadcasts on blockchain. $\mathbb{G}$ be a GDH group with prime order q and generator P. $H : \{0,1\}^* \to \mathbb{G}$ is a one-way function. In the following algorithms, $param$ is an implicit input.

2. **KeyGen**: Each voter V_i generates its public key and secret key pair $(pk_i, sk_i) \leftarrow KeyG$ in Π^{sig} scheme. n eligible voters are randomly selected to be organizers $\mathcal{L} = \{L_1, L_2, ..., L_n\}$ and participate as n players in $\mathcal{TB}$ scheme. Then $\mathcal{L}$ run TBK together to compute the followings.

   - Each player L_i gets its public share Q_i and secret share s_i, where $Q_i = P^{s_i}$. Intermediate values $P^{a_{ij}}$ and Q_i will be broadcasted to blockchain.
   - In $\mathcal{TB}$ scheme, the secret key and public key are set to s and $Q = P^s$ respectively.

   Similarly to organizers, n^* eligible voters are randomly selected to be tellers $\mathcal{T} = \{T_1, T_2, ..., T_{n^*}\}$ and participate as n^* players in $\mathcal{TE}$ scheme. Then $\mathcal{T}$ run TEK together to compute the followings.

   - Each player T_i gets its public share Q_i^* and secret share s_i^*, where $Q_i^* = P^{s_i^*}$. Intermediate values $P^{a_{ij}^*}$ and Q_i^* will be broadcasted to blockchain.
   - In $\mathcal{TE}$ scheme, the secret key and public key are set to s^* and $Q^* = P^{s^*}$ respectively.

3. **Register**: By executing TBS protocol, every eligible voter V_i can obtain a blind signature σ_i on its public key pk_i by $\sigma_i \leftarrow TBS_s(pk_i)$. Each voter V_i needs to interact with t players in $\mathcal{L}$ in order to execute TBS protocol.

4. **VoteCasting**: Each eligible voter V_i encrypts its ballot b into ciphertext C_i with the public key Q^* of $\mathcal{TE}$ scheme, i.e. $C_i \leftarrow TEC_{Q^*}(b)$. Then V_i generates

signature σ_i' of ciphertext C_i using Π^{sig} scheme with its registered secret key sk_i, i.e. $\sigma_i' \leftarrow Sign_{sk_i}(C_i)$. Then broadcast data quadruple $(pk_i, \sigma_i, C_i, \sigma_i')$ to blockchain.

5. **VoteCounting**: During vote counting, a teller T_i can check the authenticity of ciphertext by verifying signatures σ_i and σ_i'. σ_i is verified by TBV algorithm in $\mathcal{TB}$ scheme, it returns valid if the equation $\hat{e}(\sigma_i, P) = \hat{e}(H(pk_i), Q)$ holds. σ_i' is verified by $Verify$ algorithm in Π^{sig} scheme, it check the validity of signature σ_i' on ciphertext C_i using public key pk_i, i.e. $Verify(C_i, \sigma_i') \stackrel{?}{=} valid$. If all verifications are passed, t^* tellers decrypt the ciphertext C_i together by running TED protocol in $\mathcal{TE}$ scheme. The decrypted ballot b is appended to the data quadruple, finally the result data $(pk_i, \sigma_i, C_i, \sigma_i', b)$ is broadcasted to blockchain.

3.3 Security Analysis

Our system is able to achieve the four security properties, namely, *verifiability*, *eligibility*, *fairness* and *anonymity*.

- **Verifiability**. The opened ballot data quintuple $(pk_i, \sigma_i, C_i, \sigma_i', b)$ is publicly broadcast to blockchain network. Each voter V_i can verify whether its ballot has been counted correctly or not.
- **Eligibility**. This property is guaranteed by the unforgeability in threshold blind signature scheme $\mathcal{TB}$ and the typical signature scheme Π^{sig}. With unforgeability, malicious adversary cannot forge a new valid signature, thus he cannot forge the signature σ_i and σ_i' in the ballot data quadruple $(pk_i, \sigma_i, C_i, \sigma_i')$. This guarantees every valid ballots must be cast by an eligible voter. Since public key pk_i can be retrieved in ballot data quadruple, tellers can easily

recognize any ballot are cast by the same voter. Thus, the system can prevent double voting and enforce each eligible voter can only vote once.

- **Fairness**. Threshold ElGamal decryption scheme $\mathcal{TE}$ guarantees there must be at least t^* tellers to open the encrypted ballots for reviewing voting result. When there are less than t^* malicious tellers, no one can obtain the result before vote counting phase, which maintains the fairness of the voting.
- **Anonymity**. This property is guaranteed by the blindness in threshold blind signature scheme $\mathcal{TB}$. Each eligible voter V_i can obtain its blind signature σ_i on its public key pk_i to proof its identity during vote casting phase. Although the blind signature is issued by organizers, due to the blindness property, even the organizers cannot retrieve the identity. In addition to all information are broadcasted through blockchain network, the system achieves highly anonymity.

3.4 Performance Evaluation

3.4.1 Performance Evaluation

As a blockchain platform independent design, the actual system performance would vary, based on the underlying blockchain network and consensus algorithm. The blockchain transaction throughput can be varied from 7 tps to over 100 tps, or even higher for a permissioned blockchain. Thus, we conducted our experiment to focus on performance evaluation of the e-voting cryptographic protocol. We use C++ to implement our cryptographic library with PBC library [66] and Crypto++ library [22]. We choose the parameters suggested in Type A internals [67] for our

Number of Voters l / Time Consumed t'	1000	2000	3000	4000	5000
Average Time in *Register* = Total Time / l (ms)	18.007	17.987	18.001	18.255	17.919
Average Time in *VoteCasting* = Total Time / l (ms)	5.829	5.835	5.838	5.874	5.792
Average Time in *VoteCounting* = Total Time / l (ms)	11.058	11.011	11.060	11.084	10.973
Total Time in *Register* (min)	0.3	0.6	0.9	1.22	1.49
Total Time in *VoteCasting* (min)	0.1	0.19	0.29	0.39	0.48
Total Time in *VoteCounting* (min)	0.18	0.37	0.55	0.74	0.91

Table 3.1: Measure of time consumption in proposed e-voting design with $(7, 10)$-threshold settings

pairing group. Our experiment runs on MacBook Pro with 16 GB memory and 3.1 GHz Intel Core i5 processor.

In our proposed system, it contains five phases, namely, *Setup*, *KeyGen*, *Register*, *VoteCasting* and *VoteCounting*. Since *Setup* and *KeyGen* are preparation process of a voting, we only consider the performance of *Register*, *VoteCasting* and *VoteCounting* phases. The experiment results are described as follows.

Refer to Table 3.1, we set the threshold to $(7, 10)$ in both $\mathcal{TB}$ scheme and $\mathcal{TE}$ scheme in our system, then we measure the time consumed t' in each phase versus the number of voters l in the system. The average time consumed in each phase for each voter is a roughly constant value. For example, *Register* phase takes roughly 18 ms, *VoteCasting* phase takes roughly 5.8 ms and *VoteCounting* phase takes roughly 11 ms. Therefore, the total time consumed in each stage is linear to the number of voters l in the system.

In addition, we test on the relationship between the performance and different threshold parameters (t, n) in both threshold blind signature scheme $\mathcal{TB}$ and threshold ElGamal decryption scheme $\mathcal{TE}$. The average time consumed in each phase is

	[72]	[62]	[99]	Ours
Cryptographic	ZKP	ZKP	Ring signature, homomorphic encryption	Threshold blind signature, threshold encryption
Self-tallying	Yes	Yes	No	No
Support multi-candidate	No	Yes	Yes	Yes
Support large number of voters	No	No	Yes	Yes
Efficiency	Slow	Slow	Fast	Fast[1]

Table 3.2: Comparison between existing blockchain-based e-voting systems

slightly linear to the threshold size t, it only increases little additional execution time when compares with the increment of t. On the other hand, the size of players n does no impact on the time in all *Register*, *VoteCasting* and *VoteCounting* phases.

In particular, a system with $(7, 10)$-threshold takes roughly 11 ms to decrypt and count one ballot. Our e-voting protocol enables each teller to complete vote counting for one million votes on a laptop in about 26.19 minutes. The efficiency can be further improved by deploying the system to a real server cluster and executing the vote counting parallelly. Furthermore, the efficiency can be further optimized by moving the process of signature validation into *VoteCasting* phase. Thus, our whole system is efficient and practical enough to be adopted in the real-world.

3.4.2 Comparison

In Table 3.2 summarizes the comparison between the existing blockchain-based e-voting systems [72, 99, 62] and our proposed design. The work by McCorry et al. [72] had a limitation in that it only supported binary vote, while other works supported multiple candidates. Both [72] and [62] used ZKP to implement their systems, and suppored self-tallying. However, the computational demands of ZKP

[1]Our design has the best efficiency on voter side when threshold t was small.

approach made them unsuitable for large-scale voting. Comparing to the work by Yu et al. [99] and our work, we both targeted to handle large-scale voting that had one million voters. The major difference was that they adopted ring signature and homomorphic encryption, while we used threshold blind signature and threshold encryption. Their design provided a technique that pre-computes and groups the key accumulations in a setup phase, before vote casting. The computation effort for voters was still linear to the voter size n and slower than our approach if the threshold size t was small. With the same setup time (around 40s) in their suggested settings, we are able to generate 140-200-threshold keys, where the average time for register, vote casting and vote counting operation are 367ms, 5.8ms and 360ms respectively. Moreover, our system allows voters, organizers, and tellers to run the system without the need for a system administrator to host the voting process.

CHAPTER 4

MESSAGE AUTHENTICATION FOR VEHICULAR NETWORKS

In this chapter, we will describe our contribution (published in [59, 60]) in efficient message authentication with revocation transparency using blockchain for VANET. We apply several techniques including pairing-free revocable certificateless digital signature scheme, KUNodes algorithm, cuckoo filter and blockchain network to construct our proposed system.

Thanks to the use of pairing-free signature scheme, our system can achieve higher efficiency when comparing to some existing bilinear paring based schemes. To further improve the revocation efficiency, our system adopted KUNodes algorithm for time key update to reduce the complexity to logarithmic. In addition, we combine batch verification technique and cuckoo filter to construct a roadside unit (RSU) assisted signature verification, hence it can achieve even higher overall performance to meet the practical needs of VANET. Besides, blockchain takes a role for ensuring the security and accountability of the VANET. To further analyze the performance of our scheme, we also give a complexity analysis on the signature scheme and perform an experiment to measure the efficiency of cuckoo filter.

Chapter Organization. In Section 4.1, we define the entities, threats and security requirements of a message authentication scheme for VANET. The overview of our design and the details of construction are described in Section 4.2. The analysis of our system security is conducted in Section 4.3. Lastly, we give performance analysis and experiment results in Section 4.4.

4.1 Definitions

Entities. A vehicle-to-vehicle (V2V) message authentication scheme of VANET contains the following three entities, namely, *Key Generation Center (KGC)*, *Roadside Unit (RSU)* and *Vehicle*. Descriptions of the entities are given below.

- **Key Generation Center (KGC)**. A trusted authority who assists vehicle key generation and revocation.
- **Roadside Unit (RSU)**. A device installed in the critical point of road for assisting the message exchange of vehicles. The computational power of RSU is more powerful than the on-board unit (OBU) of a vehicle.
- **Vehicle**. A vehicle would broadcast and receive traffic related information to assist driving. On-board unit (OBU) is the communication device installed in a vehicle. It has limited computational power.

Threats. There are numerous attacks on VANET [51]. For a message authentication scheme adopted in VANET, it is mainly under the following four threats.

- **Bogus Information Attack**. Adversary broadcasts fake information to network on purpose. For instance, an adversary can mislead other vehicles to other direction by broadcasting fake traffic condition warning to VANET.
- **Impersonation Attack**. Adversary pretends to be another vehicle and sends malicious messages on behalf of it.
- **Man in the Middle Attack**. Messages may be modified by an adversary during the middle of transmission.
- **Message Replay Attack**. Adversary monitors the network traffic and resends a legitimate message later, which misleads other vehicles about the traffic conditions.

Security Requirements. By considering the possible attacks, the required security properties of a message authentication scheme in VANET are described as follows.

- **Identity Privacy Preserving**. The real identity of each vehicle should be hidden. In particular, it cannot be extracted from the transmitted messages in VANET by adversary.
- **Message Integrity and Authentication**. Each user in the network is allowed to verify the authenticity of messages, including to check whether a message is sent by legitimate user and to ensure a message is not modified by adversary.
- **Non-repudiation**. Sender of a message cannot deny the fact that it has done so.
- **Traceability and Revocation**. The trusted authority, i.e. KGC, is capable to reveal the real identity of sender from a message. Once a misbehaving or compromised user is detected, KGC can revoke its credential in the network.
- **Revocation Accountability and Enhance Transparency**. Since KGC is the only trusted authority for credential issuance and revocation over the VANET, KGC's activities about user revocation should be transparent and accountable for inspection purpose.

4.2 Our Construction

4.2.1 Design Overview

Our message authentication scheme for VANET is constructed by a batch verifiable certificateless signature CLS scheme, $KUNodes$ algorithm, cuckoo filter and blockchain technology. The conceptual design is outlined as follows.

As the illustration shown in Figure 4.1, the message authentication flows in VANET has eight phases. In phase 1, a vehicle submits an identity registration request to KGC. After verification, in phase 2, KGC generates a pseudo identity (PID) and initial partial secret key in CLS scheme and sends to the vehicle. Then KGC would update the time key for non-revoked vehicles periodically which is noted as phase 3 in the figure, the non-revoked vehicles are queried by $KUNodes$ algorithm. In phase 4, the vehicle can generate its full secret key for CLS using the initial partial secret key, the time key and random secret value. Afterward, the vehicle can sign messages with the full secret key. In phase 5, the vehicle broadcasts signed messages on VANET to provide useful traffic information to other vehicles. The last three phases, i.e. phases 6-8, are related to signature verification. When a vehicle receives messages in the network, it can verify the signatures locally or use RSU assisted information for verification. In the latter scenario, RSU listens to the messages in the vehicle network and performs batch signatures verification, which is noted as phase 6 in the figure. Afterward, in phase 7, RSU broadcasts a notification message, which is a cuckoo filter, to surrounding vehicles for assisting the verification. Finally, in phase 8, vehicle can perform efficient assertion test on notification messages instead of running signature verification.

During revocation, KGC adds the revoked vehicle to a revoked list and broadcasts the updated list on blockchain network, which is jointly maintained by KGC and RSUs. KGC would no longer send any new time key to revoked vehicles. Thus, after the existing time key expired, a revoked vehicle cannot generate valid signature anymore.

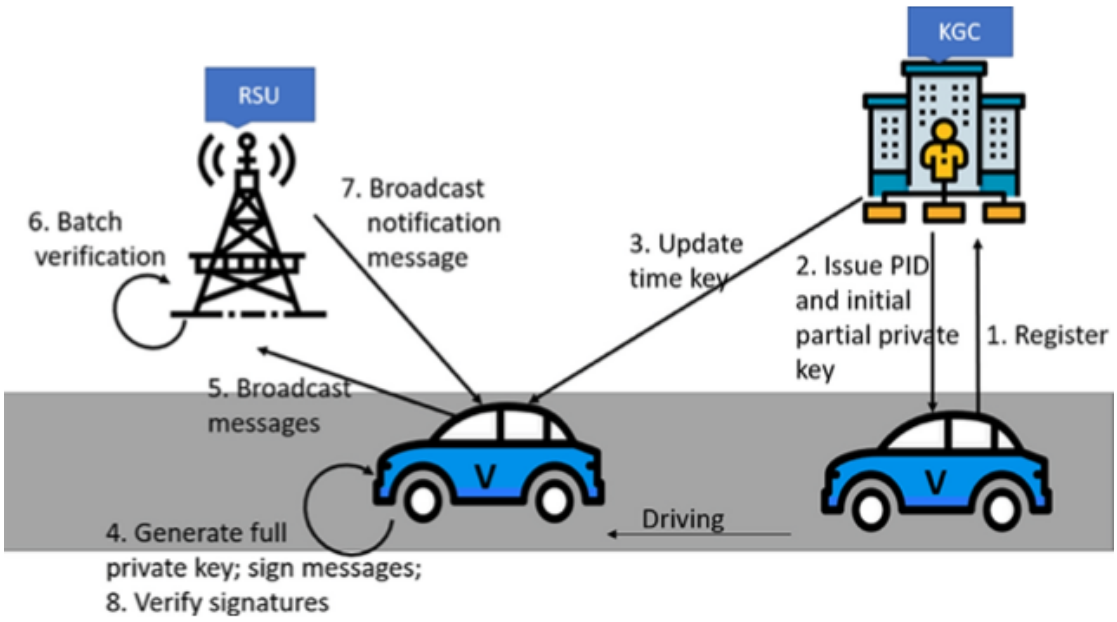


Figure 4.1 The overview of message authentication process in a VANET

4.2.2 Basic Construction

To construct our message authentication scheme for VANET, we adopt the following techniques, blockchain technology, batch verifiable certificateless signature CLS scheme, $KUNodes$ algorithm and cuckoo filter.

We assume there are maximum N users in the system. We denote V_i as the i-th vehicle in the network. We use RID_i and PID_i to denote the real identity and pseudo identity of V_i. psk_i, s_{PID_i} and vpk_{PID_i} denote the partial secret key, random secret and vehicle public key of V_i in CLS scheme respectively. We let H_1, H_2, H_3 and H_4 be four secure hash functions used in CLS scheme. Δ_t, t_r and t_i denote the valid time period of pseudo identity PID_i, revocation time and message timestamp respectively.

There are eleven basic algorithms and protocols in the scheme, namely, *System-Parameter-Setup*, *Pseudo-Identity-Generation*, *Initial-Partial-Secret-Key-Generation*, *Time-Key-Generation*, *Partial-Secret-Key-Generation*, *Full-Key-Generation*, *Offline-*

Sign, *Online-Sign*, *Individual-Verification*, *Batch-Verification* and *Revocation*. The details are listed as follows.

1. **System-Parameter-Setup**. On input security parameter 1^{λ}, KGC outputs the system parameter $param = (\mathbb{G}, q, P, H_1, H_2, H_3, H_4)$. $\mathbb{G}$ be a cyclic group with prime order q and generator P. H_1, H_2, H_3, H_4 are four secure hash functions with domain $\{0,1\}^*$ and range $\mathbb{Z}_q^*$. Then KGC randomly select a number $\alpha \in \mathbb{Z}_q^*$ as its master secret key and sets the corresponding master public key to $P_{pub} = \alpha \cdot P$. The tuple of system parameter and KGC's master public key $(param, P_{pub})$ are pre-loaded by RSUs and vehicles. Besides, KGC prepares an empty revocation list RL and a binary tree BT with N leave nodes. Each node of BT is set to a random polynomial. Moreover, KGC defines a genesis block and initializes a permissioned blockchain with it.

2. **Pseudo-Identity-Generation**. Vehicle V_i randomly picks $k_i \in \mathbb{Z}_q^*$ and computes $PID_{i,1} = k_i \cdot P$. With its real identity RID_i, V_i sends tuple $(RID_i, PID_{i,1})$ to KGC. With the tuple as input, KGC computes $PID_{i,2} = RID_i \oplus H_1((\alpha \cdot PID_{i,1}), \Delta t_i, P_{pub})$ and sends $PID_{i,2}$ to V_i if RID_i does not exist in revocation list RL. Otherwise, KGC rejects the registration request. The pseudo identity PID_i of V_i is set to $PID_i = (PID_{i,1}, PID_{i,2}, \Delta t_i)$, where Δt_i is the expiration time of PID_i. To reveal the real identity from PID_i, KGC can compute $RID_i = PID_{i,2} \oplus H_1((\alpha \cdot PID_{i,1}), \Delta t_i, P_{pub})$.

3. **Initial-Partial-Secret-Key-Generation**. KGC generates the initial partial secret key isk_{PID_i} of V_i by Algorithm 4. isk_{PID_i} is the identity-component of the full secret key of CLS scheme. The algorithm randomly chooses an empty leaf node η_{PID_i} from binary tree BT then stores PID_i into this node. Then it searches all nodes from η_{PID_i} to the root node and computes their correspond-

ing d_θ and D_θ. Finally, KGC sends the result $isk_{PID_i} = \{(\theta, d_\theta, D_\theta)\}_{\theta \in Path(\eta_{PID_i})}$ to vehicle V_i.

Algorithm 4 iskGen(PID_i, BT)

random choose an empty leaf node η_{PID_i} from BT
store PID_i in η_{PID_i}
for all $\theta \in Path(\eta_{PID_i})$ **do**
    if r_θ is $undefined$ **then**
        $r_\theta \xleftarrow{\$} \mathbb{Z}_q^*$
        store r_θ in node θ
    end if
    $d_\theta \leftarrow r_\theta \cdot P$
    $D_\theta \leftarrow r_\theta + \alpha \cdot H_2(PID_i, d_\theta)(mod\ q)$
end for
return $isk_{PID_i} \leftarrow \{(\theta, d_\theta, D_\theta)\}_{\theta \in Path(\eta_{PID_i})}$

4. **Time-Key-Generation**. KGC periodically updates and broadcasts the time keys tk_t to non-revoked users. tk_t is computed by Algorithm 5. The algorithm computes $KUNodes$ to query the minimal set of nodes to represent all non-revoked users before or at time t_r. For each node in that set, the time-component (e_μ and E_μ) is refreshed. Lastly, KGC computes tk_t and broadcasts tk_t to VANET.

Algorithm 5 tkGen(BT, RL, t_r)

for all $\mu \in KUNodes(BT, RL, t_r)$ **do**
    $m_\mu \xleftarrow{\$} \mathbb{Z}_q^*$
    $e_\mu \leftarrow m_\mu \cdot P$
    $E_\mu \leftarrow m_\mu + \alpha \cdot H_3(t_r, e_\mu)(mod\ q)$
end for
return $tk_t \leftarrow \{(\mu, e_\mu, E_\mu)\}_{\mu \in KUNodes(BT, RL, t_r)}$

5. **Partial-Secret-Key-Generation**. After receiving $isk_{PID_i} = \{(\theta, d_\theta, D_\theta)\}_{\theta \in Path(\eta_{PID_i})}$ and $tk_t = \{(\mu, e_\mu, E_\mu)\}_{\mu \in KUNodes(BT, RL, t_r)}$, a non-revoked user of V_i can compute the partial secret key by Algorithm 6. The algorithm

checks whether there is any node in common between θ and μ, which indicates that the leaf node corresponding to pseudo identity PID_i itself or has an ancestor in the non-revoked users set $KUNodes(BT, RL, t_r)$. Thus, PID_i is a non-revoked user before or at time t_r. We denote the partial secret key of PID_i as $(D^{(PID_i)}, E^{(PID_i)})$, where $D^{(PID_i)} = r + \alpha \cdot H_2(PID_i, d^{(PID_i)})(mod\ q)$; $d^{(PID_i)} = r \cdot P$; $E^{(PID_i)} = m + \alpha \cdot H_3(tr, e^{(PID_i)})(mod\ q)$ and $e^{(PID_i)} = m \cdot P$. To check the validity of partial secret keys, V_i can verify the following two equation: $D^{(PID_i)} \cdot P \stackrel{?}{=} d^{(PID_i)} + H_2(PID_i, d^{(PID_i)}) \cdot P_{pub}$ and $E^{(PID_i)} \cdot P \stackrel{?}{=} e^{(PID_i)} + H_3(t_r, e^{(PID_i)}) \cdot P_{pub}$. Note that P_{pub}, PID_i, t_r, $d^{(PID_i)}$ and $e^{(PID_i)}$ are public.

Algorithm 6 pskGen(isk_{PID_i}, tk_t)

1: **for all** $(\theta, d_\theta, D_\theta) \in isk_{PID_i}$, $(\mu, e_\mu, E_\mu) \in tk_t$ **do**
2: **if** $\exists(\theta, \mu)$ s.t. $\theta = \mu$ **then**
3: **return** $psk_{PID_{i,t}} \leftarrow (D_\theta, E_\mu)$
4: **end if**
5: **end for**
6: // isk_{PID_i}, tk_t do not have any node in common
7: **return** $psk_{PID_{i,t}} \leftarrow \perp$

6. **Full-Key-Generation**. To compute the full key pair, vehicle V_i randomly chooses $s_{PID_i} \in \mathbb{Z}_q^*$ as its secret value and computes corresponding public value by $vpk_{PID_i} = s_{PID_i} \cdot P$. A non-revoked user with pseudo identity PID_i before or at time t_r, its full secret key for signature generation is $(D^{(PID_i)}, E^{(PID_i)}, s_{PID_i})$ and the corresponding public key for signature verification is $(vpk_{PID_i}, d^{(PID_i)}, e^{(PID_i)}, t_r)$.

7. **Offline-Sign**. A vehicle V_i picks a random value $w_i \in \mathbb{Z}_q^*$ and computes $W_i = w_i \cdot P$. Then it set offline signature ϕ_i as tuple (w_i, W_i) and stores it locally. Since ϕ_i is independent to the message in signature scheme, it can be pre-computed when OBU is idle.

8. **Online-Sign**. On input message m_i, current timestamp t_i, partial secret key $(D^{(PID_i)}, E^{(PID_i)})$, secret value s_{PID_i} and offline signature tuple ϕ_i, vehicle V_i with pseudo identity PID_i generates the signature σ_i as follows. Firstly, it parses ϕ_i as (w_i, W_i) then computes $h_{4i} = H_4(m_i, PID_i, d^{(PID_i)}, e^{(PID_i)}, vpk_{PID_i}, W_i, t_i)$ and $s_i = w_i + h_4 i \cdot (D^{(PID_i)} + E^{(PID_i)} + s_{PID_i})(mod\ q)$. The output signature is set to $\sigma_i = (W_i, s_i, d^{(PID_i)}, e^{(PID_i)})$. Finally, vehicle V_i broadcasts message quintuple $(m_i, PID_i, \sigma_i, t_i, vpk_{PID_i})$ over the vehicle network.

9. **Individual-Verification**. Upon receiving message $(m_i, PID_i, \sigma_i, t_i, vpk_{PID_i})$, verifier can firstly verify the expiration time of timestamp t_i. If t_i is expired, it drops the message. Then, it computes hashes $h_{2i} = H_2(PID_i, d^{(PID_I)})$, $h_{3i} = H_3(t_r, e^{(PID_I)})$ and $h_{4i} = H_4(m_i, PID_i, d^{(PID_I)}, e^{(PID_I)}, vpk_{PID_i}, W_i, t_i)$ then checks the equation $s_i \cdot P = W_i + h_{4i} \cdot (d^{(PID_I)} + h_{2i} \cdot P_{pub} + e^{(PID_I)} + h_{3i} \cdot P_{pub} + vpk_{PID_i})$. The message is treated as valid when the equation holds, otherwise it is being rejected.

10. **Batch-Verification**. Our certificateless signature scheme supports batch verification. Multiple messages of $(m_i, PID_i, \sigma_i, t_i, vpk_{PID_i})$, where $i = 1, 2, 3, ..., n$, can be verified all the signature together. Firstly we verify the expiration times of all timestamps t_i before proceeding. Then it randomly selects a vector $v = v_1, v_2, v_3, ..., v_n$, where the value of each v_i is small. Finally, it validates the equation $(\sum_{i=1}^{n} s_i \cdot v_i) \cdot P = \sum_{i=1}^{n}(W_i \cdot v_i) + \sum_{i=1}^{n}(h_{4i} \cdot d^{(PID_i)} \cdot v_i) + (\sum_{i=1}^{n}(h_{4i} \cdot h_{2i} \cdot v_i + h_{4i} \cdot h_{3i} \cdot v_i)) \cdot P_{pub} + \sum_{i=1}^{n}(h_{4i} \cdot e^{(PID_i)} \cdot v_i) + \sum_{i=1}^{n}(h_{4i} \cdot vpk_{PID_i} \cdot v_i)$. The messages are treated as valid when the equation holds, otherwise they are being rejected.

11. **Revocation**. When a user with pseudo identity PID_i should be revoked, KGC finds out its corresponding leaf node η_{PID_i} and revocation time t_r. KGC updates the revocation list $RL \leftarrow RL \cup \{\eta_{PID_i}, t_r\}$ then the latest RL is

broadcasted and stored on blockchain network. Records on the blockchain are publicly accessible, any user can query the revocation list at anytime.

4.2.3 RSU-assisted Verification

In order to assist the nearby vehicles to perform signature verification, RSU is responsible for running batch signatures verification and generating notification message using cuckoo filter. Thus, the nearby vehicles only need to perform efficient assertion test over cuckoo filter instead running signature verification. The details about the notification message generation and verification are described as follows.

Generating Notification Message.

In general, batch verification algorithm accepts or rejects messages as a whole. A single invalid signature would cause the algorithm reject the whole batch. To prevent this, RSU could apply binary search as in [23, 25, 31] to filter all invalid signatures. After running the signatures extract, i.e. Algorithm 7, RSU obtains a list of invalid signatures in *Result*. RSU can arrange the valid and invalid signatures with the corresponding pseudo identities into two list $validList(V_i)$ and $invalidList(V_i)$. Then, RSU runs Algorithm 8 to generate a notification message. RUS creates two cuckoo filters, namely, posFilter and negFilter to store the concatenation of the messages, pseudo identities and timestamps corresponding to the valid and invalid signatures respectively. Finally, RSU signs on the notification message and broadcasts $\{posFilter, negFilter, \sigma_{RSU}\}$ over VANET.

Signature Verification base on Notification Message.

Using notification message from a nearby RUS, a vehicle V_i can verify a signature σ_j of a message m_j quickly. The protocol of message authentication using cuckoo filter is shown in Algorithm 9. Specifically, vehicle V_i computes the

Algorithm 7 signatureExtract($List$, $Result$ low, $high$)

if $batchVerify(List, low, high)$ = true **then**
    return
else
    if low = high **then**
        $Result.add(List[low])$
        return
    else
        $mid = (low + high)/2$
        $signatureExtract(List, Result, low, mid)$
        $signatureExtract(List, Result, mid + 1, high)$
        return
    end if
end if

Algorithm 8 notifyMsg($validList(V_i)$, $invalidList(V_i)$, sk_{RSU})

for $PID_i \in validList(V_i)$ **do**
    $x_i \leftarrow (PID_i||t_i||m_i)$
    $posFilter.Insert(x_i)$
end for
for $PID_i \in invalidList(V_i)$ **do**
    $x_i \leftarrow (PID_i||t_i||m_i)$
    $negFilter.Insert(x_i)$
end for
$\sigma_{RSU} \leftarrow \Pi^{sig}_{sk_{RSU}}(posFilter, negFilter)$
return $\{posFilter, negFilter, \sigma_{RSU}\}$

fingerprint of cuckoo filter $f_j = Fingerprint(x_j)$, where $x_j = (PID_j||t_j||m_j)$. V_i calculates the indexes for item x_j as $i_1 = hash(x_j) mod\ M$ and $i_2 = (i_1 \oplus hash(Fingerprint(x_j)))mod\ M$, then it queries the positive and negative cuckoo filters in the notification message with item x_j to decide accept or reject. Since cuckoo filter inherently has false positive, there are four possible scenarios as follows.

1. **posFilter outputs true and negFilter outputs false**. It indicates σ_j is valid.

2. **posFilter outputs false and negFilter outputs true**. It indicates σ_j is invalid.

3. **Both filters outputs true**. This means false positive occurs. V_i can either resend the signature σ_j back to the nearby RSU then wait for re-confirmation or verify the signature by itself.

4. **Both filters outputs false**. σ_j has not been verified by RSU. V_i may wait for the next notification message from RUS or execute individual signature verification by itself.

The case 3 is a false positive scenario in the RSU-assisted verification, which requires an additional re-confirmation process. The probability of this false positive is very low, and we will give an analysis in the later section. To further improve the efficient during re-confirmation process, RSU can cache the signature validation results for a short period of time. Thus, once received the re-confirmation request from V_i, RSU can check the validity of signature immediately and insert to the corresponding cuckoo filter in the next notification message.

Algorithm 9 V_i verifies σ_j of V_j

$x_j \leftarrow (PID_j||t_j||m_j)$
while t_j is not expired **do**
    V_i queries $posFilter$, $negFilter$ on f_j
    if $posFilter.Query(x_j) = true$ **then**
        if $negFilter.Query(x_j) = false$ **then**
            V_i accepts the validity of σ_j; **break**
        else
            V_i resends σ_j to RSU or V_i verifies σ_j by itself; **break**
        end if
    else
        if $negFilter.Query(x_j) = true$ **then**
            V_i rejects the validity of σ_j; **break**
        else
            V_i waits for next notification broadcast or V_i verifies σ_j by itself; **break**
        end if
    end if
end while

4.3 Security Analysis

Our system is able to achieve the four security properties, namely, *identity privacy preserving*, *message authentication and integrity*, *non-repudiation*, *traceability and revocation* and *revocation accountability and enhanced transparency*.

- **Identity Privacy Preserving**. Each user in the VANET uses a pseudo identity instead of real identity to communicate with each others. The pseudo identity is generated by KGC in registration phase, while the real identity can be only retrieved by using master secret key α of KGC. In particular, KGC uses equation $RID_i = PID_{i,2} \oplus H_1((\alpha \cdot PID_{i,1}), \Delta t_i, P_{pub})$. when misbehaving vehicle is found. Since it is infeasible to extract the master secret key α from the master public key P_{pub} or other messages, identity privacy of users can be preserved in our scheme.

- **Message Authentication and Integrity**. All messages in the vehicle network are signed, only legitimate user, who has registered with KGC and not yet revoked, can generate a valid signature. When a message is modified by adversary, it will be rejected during signature verification.
- **Non-repudiation**. Each message is signed by a registered vehicle before broadcasting to the vehicle network. Hence, once a vehicle generates a signature on a message, it cannot deny the fact later.
- **Traceability and Revocation**. The pseudo identity of a vehicle is set to $PID_i = (PID_{i,1}, PID_{i,2}, \Delta t_i)$, where $PID_{i,1} = k_i \cdot P$ and $PID_{i,2} = RID_i \oplus H_1((\alpha \cdot PID_{i,1}), \Delta t_i, P_{pub})$. With the master secret key α of KGC, the real identity of a vehicle can be computed by $RID_i = PID_{i,2} \oplus H_1((\alpha \cdot PID_{i,1}), \Delta t_i, P_{pup})$. To mitigate the damage from malicious or compromised users, KGC is able to revoke misbehaving users by adding the misbehaving pseudo identities into revocation list and updating the time key for all non-revoked users. Therefore, our proposed scheme can achieve both traceability and scalable revocation.
- **Revocation Accountability and Enhanced Transparency**. In our system, whenever the revocation list is updated, KGC broadcasts the activity to a blockchain network. Since it is empowered by the immutability of blockchain network, all revocation actions of KGC are accountable, transparent and available for inspection.

4.4 Performance Analysis

To analyze the performance of our proposed message authentication scheme for VANET, we analyze two key components of our construction: our certificateless

signature CLS scheme and the cuckoo filter adopted in RSU-assisted verification. The details are described in the following.

4.4.1 Complexity of our Signature Scheme

To evaluate our signature scheme, we consider the computation time for signature generation and verification and signature size. Let T_{bp}, T_{bp-m}, T_{ecc-m}, T_H and T_h be the execution time of bilinear paring, scalar multiplication in a pairing-friendly group, scalar multiplication in elliptic curve group, a map-to-point hash function and an ordinary hash function respectively. Using the parameters and benchmark results from [38], T_{bp}, T_{bp-m}, T_{ecc-m}, T_H and T_h are 4.2110 ms, 1.7090ms, 0.4420 ms, 4.406 ms and 0.0001 ms respectively. We let $|G_{pr}|$, $|G_{ecc}|$ and $|\mathbb{Z}_q^*|$ be the size of a pairing-base group element, an elliptic curve group element and group element of $\mathbb{Z}_q^*$ respectively. In particular, $|G_{pr}|$, $|G_{ecc}|$ and $|\mathbb{Z}_q^*|$ are set to 128 bytes, 40 bytes and 20 bytes respectively. The comparison between existing message authentication schemes in VANETs is listed in Table 4.1.

For pairing-based signature scheme like [90, 44], it has higher computation and communication cost when comparing to pairing-free schemes, such as [29] and ours. The signature generation and verification are roughly 28 times and 10 times faster in pairing-free schemes respectively. While comparing our scheme to [29], we slightly improve the complexity of the scheme and has a number of advantages. Firstly, our scheme supports online/offline signature generation to reduce the runtime computational cost. Secondly, we adopt KUNodes algorithm for time key update, which reduces the revocation burden of KGC from linear to logarithmic complexity. Lastly, our scheme supports an efficient RSU-assisted verification where user can perform efficient assertion test on cuckoo filter instead of signature verification.

Schemes	Sign(ms)	Verify(ms)	Signature Size (bytes)
[90]	$2T_{bp-m} + 2T_H$ ≈ 12.23	$3T_{bp} + 2T_H$ ≈ 21.445	$2\|G_{pr}\| = 256$
[44]	$2T_{bp-m} + 2T_H$ ≈ 12.23	$4T_{bp} + 3T_H + T_{bp-m}$ ≈ 31.771	$3\|G_{pr}\| = 384$
[29]	$T_{ecc-m} + 2T_h$ ≈ 0.4422	$5T_{ecc-m} + 4T_h$ ≈ 2.2104	$3\|G_{ecc}\| + \|\mathbb{Z}_q^*\| = 140$
Our scheme	$T_{ecc-m} + T_h$ ≈ 0.4421	$4T_{ecc-m} + 3T_h$ ≈ 1.7683	$3\|G_{ecc}\| + \|\mathbb{Z}_q^*\| = 140$

Table 4.1: Comparison between existing message authentication schemes in VANETs

4.4.2 Analysis of Cuckoo Filter

In our scheme, RSU performs batch signatures verification and broadcasts the results to VANET for shortening total verification time of each OBU. RSU uses a cuckoo filter to contain the signature verification results. As mentioned in Section 4.2.3, cuckoo filter has false positive which will trigger re-confirmation. However, its false positive rate is very small, the upper bound of the total probability of a false fingerprint collision is $1-(1-\frac{2}{2^f})^{2b} \approx \frac{2b}{2^f}$, where f is the length of fingerprint in bits and b the number of entries per bucket in the hash table. We set the parameters $f = 13$ and $b = 4$ to achieve the best or close-to-best space efficiency to false positive rate which fit our needs. The false positive rate is calculated to be 0.000976. Thus, it is expected that the chance of re-confirming a signature is very low. Moreover, we can suppress the false positive rate exponentially by increasing the fingerprint size f and achieve negligible false positive.

In addition, we conduct an experiment to measure the performance of cuckoo filter. We implement a simple program which utilizes the C++ library in [31] and runs over a MacBook Pro notebook with a 3.1GHz Intel i5 processor and 16 GB memory. We configure similar parameters as in [24], bucket size, fingerprint length, load factor and filter capacity are set to 4, 12 bits, 95.36% and one million respec-

tively. We perform one million executions on each basic operation of a cuckoo filter. The measured total execution time of one million executions of insertion, deletion and query are 75 ms, 67 ms and 56 ms respectively. The false positive rate of our experiment is 0.0944%. While comparing to the time of signature verification, the overhead of cuckoo filter is very small, hence using cuckoo filter for message authentication can improve the overall efficiency.

CHAPTER 5

SUPPLY CHAIN MANAGEMENT SYSTEM

In this chapter, we will describe our contribution (published in [56]) in blockchain-based supply chain system for traceability, regulation and anti-counterfeiting, and an extended research for enhancing privacy. We apply several techniques including logistic behavioral patterns, blockchain network and threshold twisted ElGamal decryption scheme to construct our proposed design.

Thanks to data immutability of blockchain technology, our system is able to prevent data tempering, which is an essential element for regulation enforcement and anti-counterfeiting. By considering literature of supply chain management [17, 28, 92] and real-world industrial requirements, we construct a more comprehensive solution, which supports quality management, multi-hop routing and tracing without unique identifier. Unlike most existing works, we require every actor, such as employee or device of a company, has its own key pair, this allows regulation authority to audit the activities in supply chain with improved accountability. In addition, complexity of our traceability algorithm is logarithmic and measure of our implementation shows that it is efficient and scalable. Finally, we introduce a threshold twisted ElGamal decryption scheme for increase privacy preserving of our system. User can encrypt sensitive information before submitting to blockchain, while the decryption key is distributed to multiple authorities. Thus, this prevents privacy leakage when part of the decryptors being compromised and limited the power of decryptors.

Chapter Organization. In Section 5.1, we define the traceability, logistic behavioral patterns, entities and security requirements of a supply chain management system. The construction of our basic design and details of extended privacy preserv-

ing mechanism are described in Section 5.2. The evaluation of our system security is conducted in Section 5.3. Lastly, we give system evaluation, comparison between existing solutions and performance analysis of our implementation in Section 5.4.

5.1 Definitions

Traceability. In a supply chain management system, the tracing unit is commonly called *lot*, which can be a single unit of product or a batch of products. Traceability can be further divided into the abilities of lot tracking and lot tracing [28]. The flows of tracking and tracing are shown in Figure 5.1 and described below:

- **Tracking**: Ability of keeping track of the flows of lots transporting from upstream to downstream in a supply chain. It is especially important during products recall when a fault in the manufacturing process is discovered.
- **Tracing**: Ability to follow the supply chain upward and determine the source of a lot. This enables customers to have capability to distinguish counterfeit products

Logistic Behavioral Patterns. The basic lot behaviors can be modeled by four patterns [19], namely, *integration*, *division*, *alteration* and *movement*. They are illustrated in Figure 5.2 and described below:

- **Integration**: A number of lots are combined into a new lot. It is a general representation of mixing, assembling and packing during manufacturing. The relationship is instantiated by $sentTo$ and $madeBy$ fields of Lot record in our system.
- **Division**: A lot is split into multiple new lots. It represents the splitting of a batch and distributing them to different parties. The fields $receivedFrom$ and $sentTo$ are used to record this pattern in our system.

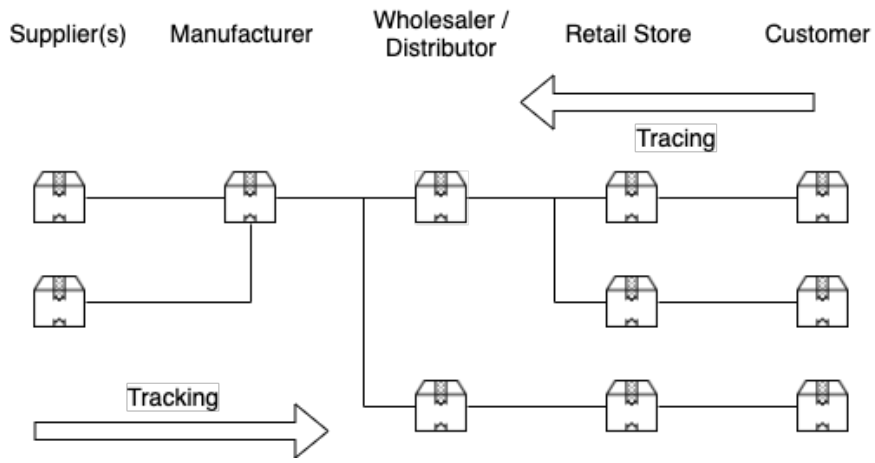


Figure 5.1 Flows of tracking and tracing

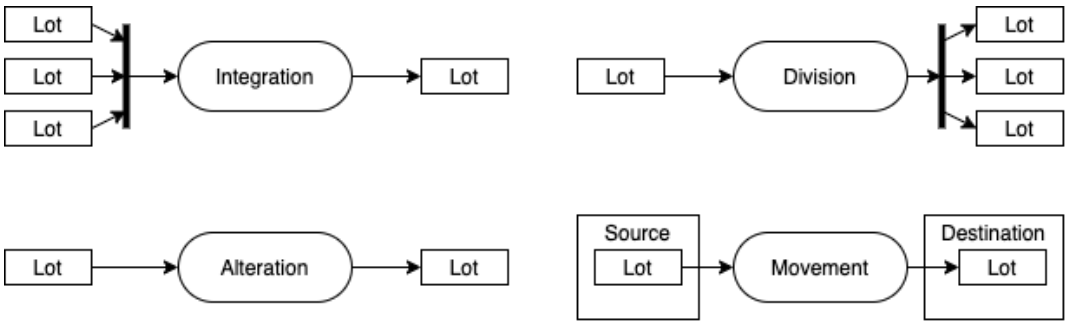


Figure 5.2 Four logistic behavioral patterns in supply chain

- **Alteration**: A lot is altered and processed. Most of the manufacturing processes belong to this pattern. In our system, we do not create a new batch. Instead, it is recorded in the procedure logs of a lot due to performance concern.
- **Movement**: A lot is moved from a source site to a destination site. It represents the internal transfer within a company between warehouses or the shipment between buyer to seller. The relationship is also recorded by $receivedFrom$ and $sentTo$ in our system.

Entities Definitions. A supply chain system contains the following six key entities, namely, *Regulation Authority*, *Company*, *Product*, *Lot*, *Procedure* and *Actor*. An

overview of the entities' relationship is shown in Figure 5.3. Descriptions of the entities are given below.

- **Regulation Authority**: An organization with regulatory authority. Responsible for certifying companies and products, monitoring the quality of lots and manufacturing procedures.
- **Company**: An organization that interacts with the product lots in a supply chain. It is governed by a regulation authority.
- **Product**: A type of goods for sale. Need to follow compliance during the manufacturing and logistic processes.
- **Lot**: A tracing unit of product in supply chain.
- **Procedure**: An activity associated to a production lot, which is recorded by an actor and monitored by the regulation authority.
- **Actor**: A person or device of a company who executes, records and signs procedure logs for a lot.

Security Requirements. The required security requirements of a supply chain system are described as follows.

- **Accountability**: Responsible parties of all logistic activities and regulatory procedures can be traced in the system.
- **Integrity**: All logistic data and regulatory procedures logs in the system is trusted, immutable, and able to prevent data tampering from adversary or malicious user.
- **Verifiability and Transparency**: All users are able to verify the logistic activities and regulatory procedures in the system. Especially for downstream

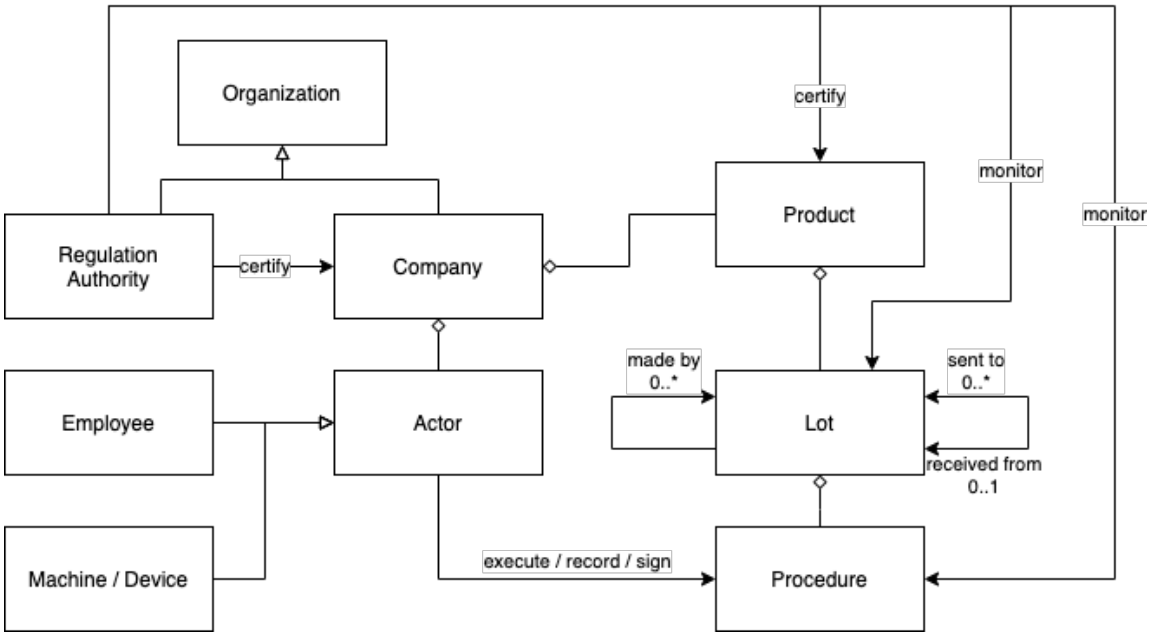


Figure 5.3 Entities relationship of our blockchain-based supply chain management system

users, including consumers and retailers, to trace upstream information in a supply chain.

- **Privacy**: A participant is able to record sensitive information through a privacy preserving mechanism in the system. Only regulation authorities can retrieve the data under particular conditions.

5.2 Our Construction

5.2.1 Overall Design

In order to develop our supply chain management system, we constructed a (t', n')-threshold twisted ElGamal decryption scheme $\mathcal{TT}$ and combined it with blockchain technology, based on common logistic behavioral patterns. The resulting conceptual design is outlined as follows.

Firstly, the platform maintainer, typically the regulatory authority, defines the initial setup of a private blockchain network. Each organization, including manufacturers, distributors, and pharmacies, sets up and connects its full mining node to the private blockchain. To prove their identities, each participant needs to generate its own key pair and register the public key through blockchain. Afterward, the supply chain system operates iteratively with three sub-steps: lot creation, lot procedure logging, and lot movement. Whenever a party creates a new batch of products or receives a lot from upstream, the employee creates a new smart contract and submits it to the blockchain to represent the new lot. The origin smart contract address is recorded in the newly created smart contract when there is any upstream lot, such as raw materials or original transmitting lot. Then, the lot goes through the defined product procedures, and all stages' information and transitions are signed and recorded in the blockchain. When the lot transfers to another location or party, it reduces its quantity and marks the receiver in the smart contract. These three steps repeat until the end of the supply chain. Since each lot's smart contract is linked together by an address pointer, any participants can scan through the smart contract to perform tracing and tracking of a lot.

To protect the privacy, participants can encrypt the data before submitting and updating the smart contract. n participants, usually the regulatory authorities or third-party auditing organizations, are selected to become decryptors. Each decryptor computes its own partial private key, and they compute the system public key together and broadcast it to the blockchain network. Each user can use the public key to encrypt data, but it requires t decryptors to work together during decryption.

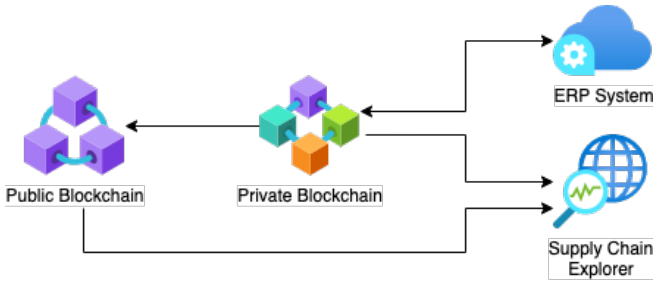


Figure 5.4 Architecture of the system

5.2.2 Basic Construction

System Architecture

While blockchain provides trust and data integrity, its lacks the performance and query handling capability to act as a centralized database replacement. Therefore, our system adopts a hybrid blockchain architecture as shown in Figure 5.4. The system can be divided into four components, namely, a *public blockchain*, a *private blockchain*, an *enterprise resource planning (ERP) system* and a *supply chain explorer*.

As mentioned in section 2.9, private blockchain has a better performance. Our data and smart contracts are mainly stored in a private blockchain, which is hosted by the participating organizations. Companies interact with the private blockchain through a centralized ERP system. All public data are uploaded to the blockchain, while private data are stored in the database of the ERP system on a cloud server or companies' own private server. To prevent collusion among blockchain node participants in a private blockchain network, hashes of block data are uploaded to a public blockchain periodically.

Supply Chain Explorer is a client providing traceability of product lots and showing the certifications of companies. It is a light-weighted client which only read data from the blockchains and smart contracts.

Digital Identities

Each organization and actor must have a digital identity in the system. Each of them must generate its blockchain key pairs and record its public identifier (i.e., account address) in the blockchain. All TXs it makes are signed by the corresponding private key before submitting to the blockchain network. In order to bind the real identity, an organization is responsible to manage its actors' digital identities and records their roles in the blockchain. Also, an organization publishes its blockchain identifier publicly, e.g., by uploading it to its official website address. An organization can also attach a digital certificate, issued by a trusted CA, which is used to generate a signature to its digital identity and uploaded to the blockchain.

Data Model and Smart Contracts

To represent the six entities defined in section 5.1, we define two smart contracts, namely, *SCOrganization* and *SCLot*. The functionality of each smart contract are described below:

- **SCOrganization**: (Refer to Script 5.1) Each deployed *SCOrganization* represents an individual organization. The contract extends $Owned$ and $Certifiable$, which allows a declaration of the owner blockchain address and certification address. There are three mapping fields in the contract, namely, *products*, *employees* and *devices*, which are used to store and allow a quick look-up for the relevant entities. When a product only has a batch ID on its packing, the system uses Algorithm 12 $TraceByBatch$ to trace the origin of a lot. The field $batchTrace$ is used to serve this algorithm. When a company receives a lot $SCLot_{from}$ and creates a contract $SCLot_{new}$, it sets $SCLot_{new}.batch = SCLot_{from}.batch$ and updates the mapping by $batchTrace[SCLot_{new}.batch]$

$.push(SCLot_{new}.address)$. The *lots* fields in the *Product* structure stores all the logs of that product. The *id* field of the *Actor* structure is the blockchain account address of the actor.

- **SCLot**: (Refer to Script 5.2) Each deployed *SCLot* represents an individual lot. This smart contract manages and represents the lot behavioral patterns. It uses the fields *madeBy*, *receivedFrom* and *sentTo* to link other lots and construct a graph of a supply chain. To represent integration, we use *madeBy* and *sentTo*. To represent division and movement, we use *receivedFrom* and *sentTo*. To represent alteration, we append *logs* and change the *status* based on the work flow. The *procedures* field stores the regulation procedures and responsible actors.

As mentioned in section 5.2.2, all TXs are signed, and thus, the sender of each smart contract's function execution is recorded in the blockchain. We use the account address of the organization and the actor for permission checking.

```
1  contract SCOrganization is Owned,
2                          Certifiable {
3    enum ActorType { Employee, Device }
4    struct Product {
5      string id;
6      address[] lots;
7      // ...
8    }
9    struct Actor {
10     address id;
```

```
11      ActorType type;
12      // ...
13    }
14    map<string, Product> products;
15    map<string, Actor> employees;
16    map<string, Actor> devices;
17    map<address, address[]> batchTrace;
18    // ...
19  }
```

Script 5.1: Smart contract of organization SCOrganization

```
1  contract SCLot is Owned {
2    struct Procedure {
3      string id;
4      address primary_actor;
5      address secondary_actor;
6      // ...
7    }
8    struct Log {
9      address actor;
10     string content;
11     unit256 timestamp;
12   }
13   address company;
14   string product;
```

```
15    address batch;
16    address[] madeBy;
17    address receivedFrom;
18    address[] sendTo;
19    uint8 status;
20    unit256 quantity;
21    Procedure[] procedures;
22    Log[] logs;
23    // ...
24  }
```

Script 5.2: Smart contract of lot SCLot

Tracking and Tracing Algorithms

The system provides three algorithms, namely, $Track$, $TraceByLot$ and $TraceByBatch$. All these algorithms only depend on data retrieval from smart contract and details are listed below:

- **Algorithm 10** $Track : ID_{lot} \rightarrow t$ takes a lot identifier ID_{lot} as input and returns a tree t which represents all downstream lots. It recursively calls the $Track$ function and crafts the returned results as a sub-tree of current root node. The algorithm will only traverse each node once. Thus, the complexity is linear to the size of the result set.
- **Algorithm 11** $TraceByLot : ID_{lot} \rightarrow l$ takes ID_{lot} as input and returns a list l of lot identifiers, which represents a path of its upstream lots. This algorithm follows the $receivedFrom$ fields of $SCLot$, recursively appends the result list. The complexity is $O(log(n))$ where n is the size of the supply chain graph.

- **Algorithm 12** $TraceByBatch : (ID_{company}, ID_{batch})$

  $\rightarrow$ m takes a company identifier $ID_{company}$ and a lot identifier ID_{batch} as input. It returns a matrix m, which represents multiple paths that the specific lot takes to reach the company. This algorithm is commonly used when a product is only print its batch identifier ID_{batch} on its packing and customer wants to trace a product which is bought from $ID_{company}$. This algorithm uses the assisting mapping $SCOrganization\#traceBatch$ to query the lots IDs of targeting batch in current company. It runs $TraceByLot$ on each item in IDs to obtain the result. Let $b = |IDs|$ be the branching factor, the complexity of $TraceByBatch$ is $O(b \cdot log(n))$

Algorithm 10 Track(ID_{lot})

```
1: Create a tree T and denote the root node as r
2: c ← SearchContract(ID_lot)
3: r ← ID_lot
4: if c ≠ φ then
5:     for all ID_to in c.sentTo do
6:         T_to ← Track(ID_to)
7:         Graft T_to as a child of r
8:     end for
9: end if
10: return T
```

Algorithm 11 TraceByLot(ID_{lot})

```
1: L ← [ID_lot]
2: c ← SearchContract(ID_lot)
3: if c = φ then
4:     return L
5: else
6:     return Concat(L, TraceByLot(c.receivedFrom))
7: end if
```

Algorithm 12 TraceByBatch($ID_{company}, ID_{batch}$)

```
1: M ← new matrix
2: c ← SearchContract(ID_company)
3: if c = φ then
4:     IDs ← c.batchTrace[ID_batch]
5:     size ← |IDs|
6:     for i ← 0 to size − 1 do
7:         M[i] ← TraceByLot(IDs[i])
8:     end for
9: end if
10: return M
```

External Storage

Currently, most blockchain frameworks do not support TX with large amount of data, meaning that storing of multimedia files on blockchain is infeasible. In our system, the multimedia files, like the certificate of company and laboratory test results of product samples, are uploaded to a separate decentralized storage and the link and a hash of the file(s) are stored in blockchain.

5.2.3 Privacy Preserving Mechanism

In order to provide both privacy preserving and data integrity, (t', n')-threshold twisted ElGamal decryption scheme $\mathcal{TT} : (TTU, TTK, TTC, TTD)$ is adopted. In our proposed design, n' decryptors are selected from the participants. Usually, these decryptors are the regulation authorities, blockchain platform maintainers and third-party auditors. During system setup phase, all decryptors execute TTU and TTK together for computing system parameter $param$, shared system public key Q' and its own partial secret s'_i. Then $param$ and Q' are uploaded to blockchain. When an organization or an actor wants to protect its data privacy in the system, it can

encrypt the sensitive information with the shared system public key by TTC before serialized into a blockchain transaction.

To review an encrypted information on blockchain, a user can send a decrypt request to t' of the decryptors. When the user is permitted to retrieve the encrypted message, such as it is an authority, t' decryptors compute TTD together. Each decryptor performs partially decryption with its s'_i and sends m_i to user. Then user is able use m_i to recovering the original message from ciphertext.

5.3 Security Analysis

Our system is able to achieve the four security requirements, namely, *accountability*, *immutable*, *verifiability and transparency*, *privacy preserving*.

- **Accountability**: Every interaction with the product lot is recorded in blockchain by executing the *log* function in *SCProcedure*. Moreover, all TXs are signed by the private key of actors, entity authentication is guaranteed by the digital signatures. Thus, the responsible party(s) of a particular stage in the manufacturing and logistic work flow can be traced easily, which is typically important for regulated industries.
- **Immutable**: Data immutability and integrity is achieved by the immutability property of blockchain. The hybrid blockchain architecture further reduces the risk of data tampering even there is any collusion among the private blockchain nodes, since hashes of private blockchain blocks are backup to a public blockchain periodically.
- **Verifiability and Transparency**: Our system allows validator nodes to join as parts of the private blockchain network, all blockchain TXs are publicly

verifiable. Besides, all the certifications of products and companies, the procedure logs of production and the logistic information can be retrieved through the smart contract functions, any user can query and verify by itself.

- **Privacy**: Any supply chain participant in our system is able to encrypt sensitive information with the proposed threshold twisted ElGamal decryption scheme. The decryption key is distributed through threshold techniques so that it requires a threshold number t' of decryptors to decrypt the message together. This prevents unauthorized access and avoids a compromised decryptor leak out the protected information. In addition, the scheme is ZKP-friendly and supports homomorphic, which allows user to create confidential transaction with auditability [15].

5.4 Evaluation

5.4.1 System Analysis

Our system is able to achieve the two core system features, namely, *traceability* and *efficient*.

- **Traceability**: In our system, all logistic information is stored in the smart contract in blockchain. A user can use UID to perform lot tracking and tracing by running $Track$ and $TraceByLot$ algorithms respectively. Also, the $TraceByBatch$ algorithm supports multi-hop routing for finding multiple possible paths generated due to lot division and lot integration in the practical settings.
- **Efficient**: We adopt a hybrid blockchain architecture in our system to maintain the efficiency. Typically, most smart contract query and TXs are submit-

ted to the private blockchain layer, which has significant improvement when comparing to directly submit TXs to a public blockchain. Besides, the proposed tracing algorithms ($TraceByLot$ and $TraceByBatch$) have logarithmic complexity that allows customer to trace the product origin efficiently for anti-counterfeiting purpose. Moreover, our implementation and testing show that the system is efficient and practical for real-world settings.

5.4.2 Comparison

Table 5.1 summarizes the comparison between the existing blockchain-based supply chain systems [43, 85, 95, 73] and our proposed design. Most of the existing solutions [85, 95, 73] were built on Ethereum blockchain platform and [43] was built on Bitcoin network with extended capabilities. While our proposed system is also implemented on Ethereum, but the design itself is possible to deploy on any blockchain platforms that supports smart contract. The proposed systems in [85, 73] and ours adopt smart contracts to store the traceability records, while [43] and [95] use UTXO and ERC721 in their design respectively. However, the design of [95] violated the original intention of ERC721 token, since a "quantity" field is added. In order to perform tracing or tracking in the system, [43] would query TXs from the blockchain. Then, trace through the transaction inputs and outputs. In our design, we direct query the smart contract instead. For [85, 95, 73], they use event logs in Ethereum as a tracing media. However, searching in event logs is slower than directly interact with smart contract. Besides, this may affect the availability of systems, since indexing event logs consumes additional computational and storage resources. In a default setting of Ethereum node, it only indexes one year of event logs, which may not be suitable for products with long expiry day. In additional,

	[43]	[85]	[95]	[73]	Ours
Blockchain Platform	Extended Bitcoin	Ethereum	Ethereum	Ethereum	Ethereum[1]
Data Structure	UTXO	Smart Contract	ERC721	Smart Contract	Smart Contract
Tracing and Tracking Media	TX	Event Logs	Event Logs	Event Logs	Smart Contract
Authentication Level	Company	Company	Company	Company	Actor (Employee or Device)
Supports Quality Management	No	No	No	No	Yes
Supports Multi-hop Routing	Yes	No	Yes	No	Yes
Supports Tracing without UID	No	No	No	No	Yes
Privacy Preserving	No	No	No	No	Yes

Table 5.1: Comparison between existing blockchain-based supply chain systems

unlike the existing solutions which authenticate users through validating the signatures of companies, our system authenticates each actor, employee or device. This increases the accountability and transparency of the system, since every action and procedure are logged in a more fine-grained manner. In [85] and [73], the proposed systems do not support multi-hop routing and do not consider all logistics behavioral patterns, while [43, 95] and our solution do. For customers, it is common that they are not in the possession of a unique identifier (UID) for tracing the origin of a product. Our proposed system includes an algorithm (Algorithm 12) in the smart contract to allow efficient tracing with without UID. Furthermore, we introduced a threshold twisted ElGamal decryption scheme for supply chain participants to encrypt sensitive information before submitting to blockchain. This increases privacy of the system while still maintains data integrity and auditability. Therefore, our system is able to provide a more complete solution to supply chain management.

5.4.3 Implementation and Performance Evaluation

Our system is implemented on a private Ethereum blockchain network hosted on Alibaba Cloud. We use Ethereum golang client (geth) version 1.9.14 as Ethereum node client and solidity version 0.4.24 for smart contract implementation. We use swarm version 0.5.7 for distributed storage on top of Ethereum. The ERP system and supply chain explorer are built by Ruby on Rails with Ruby version 2.5.1 and Rails version 5.1.6 and PostgreSQL database version 12.2. The periodic backup process of the hybrid blockchain is implemented by schedule job and a Node.js script with version 10.16.3 which submits TXs to public blockchain network.

The system is deployed to five elastic compute service (ECS) instances. Each of them has a 40 GB system disk in Ubuntu 16.04 64-bit. Each server has an Elastic IP address with a 5 MB/s bandwidth. The mining speed of Ethereum network is configured to 3 s/block. The five ECS instances are located in different physical zone and have the specifications shown in Table 5.2. The responsibility of each server is described as the follows:

- **Server 1**: Host an ERP server, a supply chain explorer server, a PostgreSQL database, a swarm node, a primary Ethereum boot node and an Ethereum remote procedure call (RPC) node.
- **Server 2**: Host a primary Ethereum mining node.
- **Server 3**: Host a secondary Ethereum boot node.
- **Server 4**: Host a secondary Ethereum mining node.
- **Server 5**: Host a secondary Ethereum mining node.

[1]The design applies to blockchain architectures that support smart contract. Our implementation is based on Ethereum.

Name	Zone	Instance Type	CUP	Memory
Server 1	Shenzhen Region Zone A	ecs.sn1ne.2xlarge	8 vCPU	16 GB
Server 2	Shenzhen Region Zone A	ecs.sn1ne.2xlarge	8 vCPU	16 GB
Server 3	Hangzhou Region Zone G	ecs.n4.small	1 vCPU	2 GB
Server 4	Shenzhen Region Zone E	ecs.hfc5.large	2 vCPU	4 GB
Server 5	Shenzhen Region Zone C	ecs.hfc5.large	2 vCPU	4 GB

Table 5.2: Specification of the instances of our supply chain system implementation

Our system's performance is evaluated by the test cases jointly designed with our collaborating industrial partners, based on a drug manufacturing scenario involving manufacturer, distributors and retailers. Each company has multiple work flow stages and processes for a product lot. For example, a manufacturer requires employees to perform eight rounds of quality assurance and quality control testing during production. Since our system works on lot (a product batch) instead of a single product unit, the underlying requirement of the blockchain is light. A typical scenario of drug selling in Shanwei City, a manufacturer sells its products to a sole distributor. The distributor sells the drugs to 103 hospitals and clinics. The monthly production of a drug can reach 20 million product units in peak. It is commonly set to 100,000 product units per batch. Thus, our system needs to handle 7 batches per day. The number of trace and track queries, on the other hand, depends on the number of product units (e.g., each customer may issue a track query when they receive a product unit). Nonetheless, the corresponding query is a read-only transaction, and typically no consensus is needed[2].

Our measurement on the performance of the system is based on the average

[2]A read-only transaction can be handled locally by a full node.

time of common operations. For lot creation, the average time is 17.98s, while for recording a stage in the work flow, the average time is 1.91s. The lot creation time is linear to the number of stages in the work flow. The average time for $TraceByLot$ and $TraceByBatch$ is 236.01ms and 716.33ms respectively. The tracing time is linear to the size of result set. The time for a tracking operation is also linear to the size of the result set, which is 8.35s for our system. Our system's lot creation, recording stages and tracking operations had acceptable average times, which may not execute frequently. However, our system's efficient tracing operations, which were the primary concern to our collaborating industrial partners, demonstrated good performance. The results indicate that the system is capable to handle real-world scenario with high performance and scalability.

CHAPTER 6

CONCLUSION AND FUTURE WORK

This thesis aims to fulfill the security requirements of some real-world applications that cannot be fulfilled by traditional cryptographic primitives, by using the blockchain technology. In particular, the contributions of this thesis is summarized as follows.

Our first contribution in [36] addresses the limitation of existing solutions of e-voting system and proposes a blockchain-based threshold voting system. In our proposed scheme, we combine several cryptographic techniques including (1) threshold cryptographic scheme to address the issue of single trusted third party, (2) blind signature for providing efficiency and anonymity to the system and (3) platform independent blockchain network to integrate with a threshold scheme, providing a trusted public bulletin board for voting. We show that our system achieves the commonly defined security requirements for an e-voting system. In addition, we provide an implementation with performance evaluation. It is notable that the time complexity in user side is constant and independent to the total number of voters in the system. Different form ring signature-based system, we can handle large amount of voters in a real-world scenario, since the execution time of each user is constant and does not scale up with an increase of the number of voters. our experiment results show that it is efficient and practical.

Our second contribution in [59, 60] demonstrates another feasible adoption of blockchain to provide efficient message authentication with revocation transparency for VANET. We combine a revocable pairing-free online/offline certificateless digital signature scheme, KUNodes algorithm, cuckoo filter and blockchain technology to achieve the security, efficiency and functional requirements of our system. Our result shows that our scheme offers better performance in computation complexity and

signature size. In addition, our scheme removes the dependency of online security mediators and dependency of secure communication channel between KGC and vehicles during revocation. Thus, it is more suitable and practical to be applied in VANET. More importantly, our system offers three additional desirable features. First, we boost signature verification efficiency by an RSU-assisted approach. RSU performs batch verification and stores results in cuckoo filters, then broadcast the filters over VANET. This allows vehicles to authenticate messages by efficient cuckoo filter query instead of running individual signature verification. Our experiment results show that it has a significant improvement on the overall efficiency. Second, we adopt KUNodes algorithm for time key refresh during revocation. It reduces the computation and communication cost of KGC logarithmically. Third, blockchain technology is adopted to manage the blacklist of revoked vehicles. It enables all network participants to verify the transaction data and revocation activities. In consequence, it enhances the accountability and transparency of KGC. Lastly, we conduct a security analysis which demonstrates that it can achieve the security requirements of message authentication in VANET.

Our last contribution in [56] addresses the existing limitations of another popular blockchain application, supply chain management system. In our proposed system, combine a threshold twisted ElGamal decryption scheme and blockchain technology to achieve security, real-world requirements of supply chain management system in regulated industries. In particular, our system has four advancements compares to existing solutions. Firstly, by considering all four logistic behavioral patterns, the system is able to support multi-hop routing. Secondly, we employ actor-based authentication and quality management to the system, which enhances system accountability and auditability. Thirdly, a composite key tracing algorithm is proposed to handle scenario of tracing without unique identifier. Lastly, thresh-

old twisted ElGamal decryption provides additional security and privacy, while still maintains limited power for regulation authorities. In addition, we give analysis on system security and system requirements, also compares between different existing blockchain-based solutions with our system. Finally, the measuring results of our implementation shows that it is efficient and practical.

In conclusion, blockchain technology is a promising technique for enhancing the security and transparency of a system. The properties of blockchain, including decentralization, immutability, integrity and transparency, add values to the typical cryptographic protocol of a system. In addition to cryptocurrency, we find that blockchain technology is especially suitable for application which has a large public concern, like government service, public service and supply chain system. Blockchain acts as a building block of the application to reduce the trust of a centralized authority and improve the transparency of a system. Moreover, we find that threshold cryptographic is a proper approach to be adopted in a blockchain application. Because in many scenarios of blockchain applications, reduce the trust and decentralized the system are two major goals. Finally, we discover that privacy of a blockchain application could be a concern, due to the immutability and transparency of blockchain. However, pseudo identity, encryption and zero-knowledge proof can be employed to address the issue.

Future Work. There are many interesting topics we would like to further develop on. For example, how to enhance the privacy in VANETs. Adoption of advanced cryptographic primitives will help to develop a more secure and efficient message authentication protocol for VANET. Attribute-based signature (ABS) [58] is signature scheme which generates signature base on user's attribute instead of identity. We will investigate the possibility of the adoption of ABS in VANET for the enhance-

ment of user privacy. Furthermore, we will investigate if ring signature, threshold cryptographic and blockchain network would improve the transparency or reduce the trust of RSU and KGC in the system.

In addition, we will continuously work on the blockchain based supply chain system. We will extend the current adoption of threshold twisted ElGamal decryption scheme to not only encrypt sensitive information in the system. Thanks to the additively homomorphic and ZKP-friendly properties of twisted ElGamal [15], we will develop a new protocol to support privacy transactions, so that supply chain participants can sell and transfer their products to others without disclosing the number of inventory. We will investigate the adoption of bulletproofs [9] over our threshold twisted ElGamal.

BIBLIOGRAPHY

[1] Riza Aditya et al. "Secure e-voting for preferential elections". In: Electronic Government. Springer Berlin Heidelberg, 2003, pp. 246–249. ISBN: 978-3-540-45239-3.

[2] Ikram Ali et al. "A blockchain-based certificateless public key signature scheme for vehicle-to-infrastructure communication in VANETs". In: *Journal of Systems Architecture* 99 (2019), p. 101636. ISSN: 1383-7621. DOI: https://doi.org/10.1016/j.sysarc.2019.101636. URL: http://www.sciencedirect.com/science/article/pii/S1383762119302103.

[3] Leonardo Aniello et al. "Towards a supply chain management system for counterfeit mitigation using blockchain and PUF". In: *arXiv preprint arXiv: 1908.09585* (2019).

[4] Man Ho Au et al. "Malicious KGC attacks in certificateless cryptography". In: *Proceedings of the 2nd ACM symposium on Information, computer and communications security*. Association for Computing Machinery, 2007, pp. 302–311. DOI: 10.1145/1229285.1266997. URL: https://doi.org/10.1145/1229285.1266997.

[5] Alessio Bechini et al. "Patterns and technologies for enabling supply chain traceability through collaborative e-business". In: *Information and Software Technology* 50.4 (2008), pp. 342–359. ISSN: 0950-5849. DOI: https://doi.org/10.1016/j.infsof.2007.02.017. URL: http://www.sciencedirect.com/science/article/pii/S0950584907000213.

[6] Josh Daniel Cohen Benaloh. "Verifiable secret-ballot elections". Thesis. 1987.

[7] Burton H. Bloom. "Space/time trade-offs in hash coding with allowable errors". In: *Commun. ACM* 13.7 (1970), pp. 422–426. ISSN: 0001-0782. DOI: 10.1145/362686.362692. URL: https://doi.org/10.1145/362686.362692.

[8] Alexandra Boldyreva, Vipul Goyal, and Virendra Kumar. "Identity-based encryption with efficient revocation". In: *Proceedings of the 15th ACM Conference on Computer and Communications Security*. CCS '08. Alexandria, Virginia, USA: Association for Computing Machinery, 2008, pp. 417–426. ISBN: 9781595938107. DOI: 10.1145/1455770.1455823. URL: https://doi.org/10.1145/1455770.1455823.

[9] Benedikt Bünz et al. "Bulletproofs: Short proofs for confidential transactions and more". In: *2018 IEEE Symposium on Security and Privacy (SP)*. 2018, pp. 315–334. DOI: 10.1109/SP.2018.00020.

[10] Vitalik Buterin. "A next-generation smart contract and decentralized application platform". In: *white paper* 3.37 (2014).

[11] James I Cash and Benn R Konsynski. "IS redraws competitive boundaries". In: *Harvard business review* 63.2 (1985), pp. 134–142.

[12] David Chaum. "Untraceable electronic mail, return addresses, and digital pseudonyms". In: *Communications of the ACM* 24.2 (1981), pp. 84–90. ISSN: 0001-0782.

[13] David Chaum. "Blind signatures for untraceable payments". In: *Advances in Cryptology*. Ed. by David Chaum, Ronald L. Rivest, and Alan T. Sherman. Springer US, 1983, pp. 199–203. ISBN: 978-1-4757-0602-4.

[14] S. Chen et al. “A blockchain-based supply chain quality management framework”. In: *2017 IEEE 14th International Conference on e-Business Engineering (ICEBE)*. 2017, pp. 172–176. DOI: 10.1109/ICEBE.2017.34.

[15] Yu Chen et al. “PGC: Decentralized confidential payment system with auditability”. In: *Computer Security – ESORICS 2020*. Ed. by Liqun Chen et al. Cham: Springer International Publishing, 2020, pp. 591–610. ISBN: 978-3-030-58951-6.

[16] T. W. Chim et al. “SPECS: Secure and privacy enhancing communications schemes for VANETs”. In: *Ad Hoc Networks* 9.2 (2011), pp. 189–203. ISSN: 1570-8705. DOI: https://doi.org/10.1016/j.adhoc.2010.05.005. URL: http://www.sciencedirect.com/science/article/pii/S1570870510000648.

[17] Vivek Choudhury. “Strategic choices in the development of interorganizational information systems”. In: *Information Systems Research* 8.1 (1997), pp. 1–24. ISSN: 1047-7047. DOI: 10.1287/isre.8.1.1. URL: https://doi.org/10.1287/isre.8.1.1.

[18] Josh D Cohen and Michael J Fischer. *A robust and verifiable cryptographically secure election scheme*. Yale University. Department of Computer Science, 1985.

[19] Committee for Developing Educational Materials for Food Traceability. *Handbook for introduction of food traceability systems (Guidelines for food traceability)*. [Online]. Available: https://www.maff.go.jp/j/syouan/seisaku/trace/pdf/handbook_en.pdf. Food Marketing Research and Information Center (FMRIC), Japan, 2003. URL: https://www.maff.go.jp/j/syouan/seisaku/trace/pdf/handbook_en.pdf.

[20] Ronald Cramer, Rosario Gennaro, and Berry Schoenmakers. "A secure and optimally efficient multi-authority election scheme". In: *Proceedings of the 16th annual international conference on Theory and application of cryptographic techniques*. Springer-Verlag, 1997, pp. 103–118.

[21] Ronald Cramer et al. "Multi-authority secret-ballot elections with linear work". In: Advances in Cryptology — EUROCRYPT '96. Springer Berlin Heidelberg, 1996, pp. 72–83. ISBN: 978-3-540-68339-1.

[22] *Crypto++*. [Online]. Available: https://www.cryptopp.com/.

[23] J. Cui et al. "SPACF: A secure privacy-preserving authentication scheme for VANET with cuckoo filter". In: *IEEE Transactions on Vehicular Technology* 66.11 (2017), pp. 10283–10295. ISSN: 1939-9359. DOI: 10.1109/TVT.2017.2718101.

[24] J. Cui et al. "An efficient message-authentication scheme based on edge computing for vehicular ad hoc networks". In: *IEEE Transactions on Intelligent Transportation Systems* 20.5 (2019), pp. 1621–1632. DOI: 10.1109/TITS.2018.2827460.

[25] Jie Cui et al. "An efficient certificateless aggregate signature without pairings for vehicular ad hoc networks". In: *Information Sciences* 451-452 (2018), pp. 1–15. ISSN: 0020-0255. DOI: https://doi.org/10.1016/j.ins.2018.03.060. URL: http://www.sciencedirect.com/science/article/pii/S0020025516310519.

[26] Yvo Desmedt and Yair Frankel. "Threshold cryptosystems". In: Advances in Cryptology — CRYPTO' 89 Proceedings. Springer New York, 1990, pp. 307–315. ISBN: 978-0-387-34805-6.

[27] Yevgeniy Dodis et al. “Anonymous identification in ad hoc groups”. In: Advances in Cryptology - EUROCRYPT 2004. Springer Berlin Heidelberg, 2004, pp. 609–626. ISBN: 978-3-540-24676-3.

[28] Kees-Jan van Dorp. “Tracking and tracing:a structure for development and contemporary practices”. In: *Logistics Information Management* 15.1 (2002), pp. 24–33. ISSN: 0957-6053. DOI: 10.1108/09576050210412648. URL: https://doi.org/10.1108/09576050210412648.

[29] H. Du, Q. Wen, and S. Zhang. “A provably-secure outsourced revocable certificateless signature scheme without bilinear pairings”. In: *IEEE Access* 6 (2018), pp. 73846–73855. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2018.2880875.

[30] William Entriken et al. “EIP-721: ERC-721 non-fungible token standard”. In: *Ethereum Improvement Proposals* (2018). [Online]. Available: https://eips.ethereum.org/EIPS/eip-721.

[31] Bin Fan et al. “Cuckoo filter: Practically better than bloom”. In: *Proceedings of the 10th ACM International on Conference on emerging Networking Experiments and Technologies*. Association for Computing Machinery, 2014, pp. 75–88. DOI: 10.1145/2674005.2674994. URL: https://doi.org/10.1145/2674005.2674994.

[32] Atsushi Fujioka, Tatsuaki Okamoto, and Kazuo Ohta. “A practical secret voting scheme for large scale elections”. In: *International Workshop on the Theory and Application of Cryptographic Techniques*. Springer, 1992, pp. 244–251.

[33] Jun Furukawa, Kengo Mori, and Kazue Sako. “An implementation of a mix-net based network voting scheme and its use in a private organization”. In:

Towards Trustworthy Elections: New Directions in Electronic Voting. Ed. by David Chaum et al. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 141–154. ISBN: 978-3-642-12980-3. DOI: 10.1007/978-3-642-12980-3_8. URL: https://doi.org/10.1007/978-3-642-12980-3_8.

[34] Jun Furukawa et al. "An implementation of a universally verifiable electronic voting scheme based on shuffling". In: Financial Cryptography. Springer Berlin Heidelberg, 2003, pp. 16–30. ISBN: 978-3-540-36504-4.

[35] Marion Garaus and Horst Treiblmaier. "The influence of blockchain-based food traceability on retailer choice: The mediating role of trust". In: *Food Control* 129 (2021), p. 108082. ISSN: 0956-7135. DOI: https://doi.org/10.1016/j.foodcont.2021.108082. URL: https://www.sciencedirect.com/science/article/pii/S0956713521002206.

[36] Borui Gong et al. "Blockchain-based threshold electronic voting system". In: *Security and Privacy in Social Networks and Big Data*. Ed. by Weizhi Meng and Steven Furnell. Springer Singapore, 2019, pp. 238–250. ISBN: 978-981-15-0758-8.

[37] D. He, J. Chen, and R. Zhang. "An efficient and provably-secure certificateless signature scheme without bilinear pairings". In: *Int. J. Commun. Syst.* 25.11 (2012), pp. 1432–1442. ISSN: 1074-5351. DOI: 10.1002/dac.1330. URL: https://doi.org/10.1002/dac.1330.

[38] D. He et al. "An efficient identity-based conditional privacy-preserving authentication scheme for vehicular ad hoc networks". In: *IEEE Transactions on Information Forensics and Security* 10.12 (2015), pp. 2681–2691. ISSN: 1556-6021. DOI: 10.1109/TIFS.2015.2473820.

[39] I-Hsuan Hong et al. "An RFID application in the food supply chain: A case study of convenience stores in Taiwan". In: *Journal of Food Engineering* 106.2 (2011), pp. 119–126. ISSN: 0260-8774. DOI: https://doi.org/10.1016/j.jfoodeng.2011.04.014. URL: https://www.sciencedirect.com/science/article/pii/S026087741100210X.

[40] Daira Hopwood et al. "Zcash protocol specification". In: Electric Coin Company, 2016.

[41] Shi-Jinn Horng et al. "An efficient certificateless aggregate signature with conditional privacy-preserving for vehicular sensor networks". In: *Information Sciences* 317 (2015), pp. 48–66. ISSN: 0020-0255. DOI: https://doi.org/10.1016/j.ins.2015.04.033. URL: http://www.sciencedirect.com/science/article/pii/S0020025515003151.

[42] J. Hua et al. "Blockchain based provenance for agricultural products: A distributed platform with duplicated and shared bookkeeping". In: *2018 IEEE Intelligent Vehicles Symposium (IV)*. 2018, pp. 97–101. ISBN: 1931-0587. DOI: 10.1109/IVS.2018.8500647.

[43] Y. Huang, J. Wu, and C. Long. "Drugledger: A practical blockchain system for drug traceability and regulation". In: *2018 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)*. 2018, pp. 1137–1144. DOI: 10.1109/Cybermatics_2018.2018.00206.

[44] Ying-Hao Hung, Yuh-Min Tseng, and Sen-Shan Huang. "A revocable certificateless short signature scheme and its authentication application". In: *Informatica* 27 (2016), pp. 549–572. ISSN: 1822-8844.

[45] ISO. "ISO 8402:1994 Quality management and quality assurance - Vocabulary". In: (1994).

[46] Markus Jakobsson, Ari Juels, and Ronald L Rivest. "Making mix nets robust for electronic voting by randomized partial checking". In: *USENIX security symposium*. San Francisco, USA, 2002, pp. 339–353.

[47] J. Abou Jaoude and R. George Saade. "Blockchain applications – Usage in different domains". In: *IEEE Access* 7 (2019), pp. 45360–45381. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2019.2902501.

[48] Xiaoying Jia et al. "An efficient provably-secure certificateless signature scheme for Internet-of-Things deployment". In: *Ad Hoc Networks* 71 (2018), pp. 78–87. ISSN: 1570-8705. DOI: https://doi.org/10.1016/j.adhoc.2018.01.001. URL: http://www.sciencedirect.com/science/article/pii/S1570870518300015.

[49] Hak Soo Ju et al. "Efficient revocation of security capability in certificateless public key cryptography". In: *Knowledge-Based Intelligent Information and Engineering Systems*. Ed. by Rajiv Khosla, Robert J. Howlett, and Lakhmi C. Jain. Springer Berlin Heidelberg, 2005, pp. 453–459. ISBN: 978-3-540-31986-3.

[50] W. Juang, C. Lei, and H. Liaw. "A verifiable multi-authority secret election allowing abstention from voting". In: *The Computer Journal* 45.6 (2002), pp. 672–682. ISSN: 1460-2067. DOI: 10.1093/comjnl/45.6.672.

[51] Mohammed Saeed Al-kahtani. "Survey on security attacks in Vehicular Ad hoc Networks (VANETs)". In: *2012 6th International Conference on Signal Processing and Communication Systems*. 2012, pp. 1–9. DOI: 10.1109/ICSPCS.2012.6507953.

[52] Ismaila Adeniyi Kamil and Sunday Oyinlola Ogundoyin. "An improved certificateless aggregate signature scheme without bilinear pairings for vehicular ad hoc networks". In: *Journal of Information Security and Applications* 44 (2019), pp. 184–200. ISSN: 2214-2126. DOI: https://doi.org/10.1016/j.jisa.2018.12.004. URL: http://www.sciencedirect.com/science/article/pii/S2214212618305507.

[53] Thomas Kelepouris, Katerina Pramatari, and Georgios Doukidis. "RFID-enabled traceability in the food supply chain". In: *Industrial Management & data systems* (2007).

[54] Aggelos Kiayias and Moti Yung. "Self-tallying elections and perfect ballot secrecy". In: Public Key Cryptography. Springer Berlin Heidelberg, 2002, pp. 141–158. ISBN: 978-3-540-45664-3.

[55] Su-Hyun Kim and Im-Yeong Lee. "A secure and efficient vehicle-to-vehicle communication scheme using bloom filter in vanets". In: *International Journal of Security & Its Applications* 8.2 (2014).

[56] Wang Fat Lau, Dennis Y. W. Liu, and Man Ho Au. "Blockchain-based supply chain system for traceability, regulation and anti-counterfeiting". In: *2021 IEEE International Conference on Blockchain (Blockchain)*. 2021, pp. 82–89. DOI: 10.1109/Blockchain53845.2021.00022.

[57] Byoungcheon Lee and Kwangjo Kim. "Receipt-free electronic voting scheme with a tamper-resistant randomizer". In: Information Security and Cryptology — ICISC 2002. Springer Berlin Heidelberg, 2003, pp. 389–406. ISBN: 978-3-540-36552-5.

[58] Jin Li et al. "Attribute-based signature and its applications". In: *Proceedings of the 5th ACM Symposium on Information, Computer and Communications*

Security. ASIACCS '10. Beijing, China: Association for Computing Machinery, 2010, pp. 60–69. ISBN: 9781605589367. DOI: 10.1145/1755688.1755697. URL: https://doi.org/10.1145/1755688.1755697.

[59] Kang Li, Wang Fat Lau, and Man Ho Au. "A secure and efficient privacy-preserving authentication scheme for vehicular networks with batch verification using cuckoo filter". In: *Network and System Security*. Ed. by Joseph K. Liu and Xinyi Huang. Springer International Publishing, 2019, pp. 615–631. ISBN: 978-3-030-36938-5.

[60] Kang Li et al. "Efficient message authentication with revocation transparency using blockchain for vehicular networks". In: *Computers & Electrical Engineering* 86 (2020), p. 106721. ISSN: 0045-7906. DOI: https://doi.org/10.1016/j.compeleceng.2020.106721. URL: http://www.sciencedirect.com/science/article/pii/S0045790620305760.

[61] Trupil Limbasiya and Debasis Das. "Secure message confirmation scheme based on batch verification in vehicular cloud computing". In: *Physical Communication* 34 (2019), pp. 310–320. ISSN: 1874-4907. DOI: https://doi.org/10.1016/j.phycom.2018.07.015. URL: http://www.sciencedirect.com/science/article/pii/S187449071730352X.

[62] Yikang Lin and Peng Zhang. "Blockchain-based complete self-tallying e-voting protocol". In: *2019 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*. 2019, pp. 47–52. DOI: 10.1109/APSIPAASC47483.2019.9023220.

[63] Dan Liu et al. "Efficient anonymous roaming authentication scheme using certificateless aggregate signature in wireless network". In: *Journal on Communication* 37.7 (2016), pp. 182–192.

[64] Joseph K. Liu, Victor K. Wei, and Duncan S. Wong. "Linkable spontaneous anonymous group signature for ad hoc groups". In: *Information Security and Privacy*. Ed. by Huaxiong Wang, Josef Pieprzyk, and Vijay Varadharajan. Springer Berlin Heidelberg, 2004, pp. 325–335. ISBN: 978-3-540-27800-9.

[65] L. Liu and W. Jia. "Business model for drug supply chain based on the internet of things". In: *2010 2nd IEEE InternationalConference on Network Infrastructure and Digital Content*. 2010, pp. 982–986. DOI: 10.1109/ICNIDC.2010.5657943.

[66] Ben Lynn. *PBC library*. [Online]. Available: https://crypto.stanford.edu/pbc/.

[67] Ben Lynn. "Type A internals". In: *PBC Library Manual* (). [Online]. Available: https://crypto.stanford.edu/pbc/manual/ch08s03.html.

[68] Avleen Malhi and Shalini Batra. "Privacy-preserving authentication framework using bloom filter for secure vehicular communications". In: *International Journal of Information Security* 15.4 (2016), pp. 433–453. ISSN: 1615-5270. DOI: 10.1007/s10207-015-0299-4. URL: https://doi.org/10.1007/s10207-015-0299-4.

[69] N. Malik et al. "Blockchain based secured identity authentication and expeditious revocation framework for vehicular networks". In: *2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/ 12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE)*. 2018, pp. 674–679. ISBN: 2324-9013. DOI: 10.1109/TrustCom/BigDataSE.2018.00099.

[70] Thomas W Malone, Joanne Yates, and Robert I Benjamin. "Electronic markets and electronic hierarchies". In: *Communications of the ACM* 30.6 (1987), pp. 484–497.

[71] V. Mateu, F. Sebé, and M. Valls. "Blind certificates for secure electronic voting". In: *2013 10th International Conference on Information Technology: New Generations*. 2013, pp. 20–26. DOI: 10.1109/ITNG.2013.11.

[72] Patrick McCorry, Siamak F. Shahandashti, and Feng Hao. "A smart contract for boardroom voting with maximum voter privacy". In: *Financial Cryptography and Data Security*. Ed. by Aggelos Kiayias. Springer International Publishing, 2017, pp. 357–375. ISBN: 978-3-319-70972-7.

[73] Ahmad Musamih et al. "A blockchain-based approach for drug traceability in healthcare supply chain". In: *IEEE Access* 9 (2021), pp. 9728–9743. DOI: 10.1109/ACCESS.2021.3049920.

[74] Satoshi Nakamoto. "Bitcoin: A peer-to-peer electronic cash system". In: *Bitcoin* 4 (2008). [Online]. Available: https://bitcoin.org/bitcoin.pdf. URL: https://bitcoin.org/bitcoin.pdf.

[75] National Medical Products Administration, China. "Drug administration law chapter 6 drug packaging management". In: (2002). [Online]. Available: https://www.nmpa.gov.cn/xxgk/zhcjd/20020120134401569.html.

[76] National Medical Products Administration, China. "Drug Production Management Law". In: (2020). [Online]. Available: https://www.nmpa.gov.cn/yaopin/ypfgwj/ypfgbmgzh/20200330182901110.html.

[77] Tatsuaki Okamoto. "Receipt-free electronic voting schemes for large scale elections". In: Security Protocols. Springer Berlin Heidelberg, 1998, pp. 25–35. ISBN: 978-3-540-69688-9.

[78] Rasmus Pagh and Flemming Friche Rodler. "Cuckoo hashing". In: *Journal of Algorithms* 51.2 (2001), pp. 122–144. ISSN: 0196-6774. DOI: https://doi.org/10.1016/j.jalgor.2003.12.002. URL: http://www.sciencedirect.com/science/article/pii/S0196677403001925.

[79] PwC. *2018 market survey report for (non-financial) application of blockchain in China*. Report. PwC, 2018. URL: https://www.pwccn.com/en/risk-assurance/2018-china-blockchain-survery-report-en.pdf.

[80] PwC. *PwC's global blockchain survey 2018*. Report. PwC, 2018. URL: https://www.pwccn.com/en/research-and-insights/publications/global-blockchain-survey-2018/global-blockchain-survey-2018-report.pdf.

[81] Ronald L Rivest, Len Adleman, and Michael L Dertouzos. "On data banks and privacy homomorphisms". In: *Foundations of secure computation* 4.11 (1978), pp. 169–180.

[82] Sattam S. Al-Riyami and Kenneth G. Paterson. "Certificateless public key cryptography". In: Advances in Cryptology - ASIACRYPT 2003. Springer Berlin Heidelberg, 2003, pp. 452–473. ISBN: 978-3-540-40061-5.

[83] P. Y. A. Ryan. "Prêt à voter with paillier encryption". In: *Mathematical and Computer Modelling* 48.9 (2008), pp. 1646–1662. ISSN: 0895-7177. DOI: https://doi.org/10.1016/j.mcm.2008.05.015. URL: http://www.sciencedirect.com/science/article/pii/S0895717708001763.

[84] Kazue Sako and Joe Kilian. "Receipt-free mix-type voting scheme". In: *Advances in Cryptology — EUROCRYPT '95*. Ed. by Louis C. Guillou and Jean-Jacques Quisquater. Springer Berlin Heidelberg, 1995, pp. 393–403. ISBN: 978-3-540-49264-1.

[85] N. Saxena et al. “PharmaCrypt: Blockchain for critical pharmaceutical industry to counterfeit drugs”. In: *Computer* 53.7 (2020), pp. 29–44. DOI: 10.1109/MC.2020.2989238.

[86] Markus Schäffer, Monika di Angelo, and Gernot Salzer. “Performance and scalability of private ethereum blockchains”. In: *Business Process Management: Blockchain and Central and Eastern Europe Forum*. Ed. by Claudio Di Ciccio et al. Cham: Springer International Publishing, 2019, pp. 103–118. ISBN: 978-3-030-30429-4.

[87] Francesc Sebé et al. “Simple and efficient hash-based verifiable mixing for remote electronic voting”. In: *Computer Communications* 33.6 (2010), pp. 667–675. ISSN: 0140-3664. DOI: https://doi.org/10.1016/j.comcom.2009.11.013. URL: http://www.sciencedirect.com/science/article/pii/S0140366409003028.

[88] István András Seres et al. “Mixeth: efficient, trustless coin mixing service for ethereum”. In: *International Conference on Blockchain Economics, Security and Protocols (Tokenomics 2019)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik. 2019.

[89] State Administration for Market Regulation. “Drug registration and management law”. In: (2020). URL: [Online].%20Available:%20%5Curl%7Bhttp://gkml.samr.gov.cn/nsjg/fgs/202003/t20200330_313670.html%7D.

[90] Yinxia Sun, Futai Zhang, and Limin Shen. “A revocable certificateless signature scheme”. In: *Journal of Computers* 9.8 (2014), pp. 1843–1851. ISSN: 1796-203X.

[91] Pavel Tarasov and Hitesh Tewari. *Internet voting using Zcash*. Cryptology ePrint Archive, Paper 2017/585. https://eprint.iacr.org/2017/585. 2017. URL: https://eprint.iacr.org/2017/585.

[92] Maitri Thakur and Charles R. Hurburgh. "Framework for implementing traceability system in the bulk grain supply chain". In: *Journal of Food Engineering* 95.4 (2009), pp. 617–626. ISSN: 0260-8774. DOI: https://doi.org/10.1016/j.jfoodeng.2009.06.028. URL: https://www.sciencedirect.com/science/article/pii/S0260877409003264.

[93] Patrick P. Tsang and Victor K. Wei. "Short linkable ring signatures for e-voting, e-cash and attestation". In: Information Security Practice and Experience. Springer Berlin Heidelberg, 2005, pp. 48–60. ISBN: 978-3-540-31979-5.

[94] Duc Liem Vo, Fangguo Zhang, and Kwangjo Kim. "A new threshold blind signature scheme from pairings". In: (2003).

[95] Martin Westerkamp, Friedhelm Victor, and Axel Küpper. "Tracing manufacturing processes using blockchain-based token compositions". In: *Digital Communications and Networks* 6.2 (2020), pp. 167–176. ISSN: 2352-8648. DOI: https://doi.org/10.1016/j.dcan.2019.01.007. URL: http://www.sciencedirect.com/science/article/pii/S235286481830244X.

[96] Zhe Xia et al. "Analysis, improvement and simplification of Prêt à voter with paillier encryption". In: *Proceedings of the conference on Electronic voting technology*. USENIX Association, 2008, Article 13.

[97] Xiwei Xu, Ingo Weber, and Mark Staples. *Architecture for blockchain applications*. 2019. ISBN: 978-3-030-03034-6. DOI: 10.1007/978-3-030-03035-3.

[98] Kuo-Hui Yeh et al. “A novel certificateless signature scheme for smart objects in the Internet-of-Things”. In: *Sensors (Basel, Switzerland)* 17.5 (2017/05// 2017). DOI: 10.3390/s17051001. URL: https://doi.org/10.3390/s17051001.

[99] Bin Yu et al. “Platform-independent secure blockchain-based voting system”. In: *Information Security*. Ed. by Liqun Chen, Mark Manulis, and Steve Schneider. Springer International Publishing, 2018, pp. 369–386. ISBN: 978-3-319-99136-8.

[100] Dae Hyun Yum and Pil Joong Lee. “Generic construction of certificateless signature”. In: Information Security and Privacy. Springer Berlin Heidelberg, 2004, pp. 200–211. ISBN: 978-3-540-27800-9.

[101] J. Zhang and Zhao Xubing. “An efficient revocable certificateless signature scheme”. In: *2015 12th International Conference on Fuzzy Systems and Knowledge Discovery (FSKD)*. 2015, pp. 1852–1857. DOI: 10.1109/FSKD.2015.7382229.

[102] Zhichao Zhao and T-H Hubert Chan. “How to vote privately using bitcoin”. In: *International Conference on Information and Communications Security*. Springer, 2015, pp. 82–96.