Summer 8-10-2021

Privacy Preservation & Security Solutions in Blockchain Network

Saide Zhu

Recommended Citation
Zhu, Saide, "Privacy Preservation & Security Solutions in Blockchain Network." Dissertation, Georgia State University, 2021.
doi: https://doi.org/10.57709/23973698

PRIVACY PRESERVATION & SECURITY SOLUTIONS IN BLOCKCHAIN NETWORK

by

Saide Zhu

Under the Direction of Wei Li, PhD and Zhipeng Cai, PhD

ABSTRACT

Blockchain has seen exponential progress over the past few years, and today its usage extends well beyond cryptocurrencies. Its features, including openness, transparency, secure communication, difficult falsification, and multi-consensus, have made it one of the most valuable technology in the world. In most open blockchain platforms, any node can access the data on the blockchain, which leads to a potential risk of personal information leakage. So the issue of blockchain privacy and security is particularly prominent and has become an important research topic in the field of blockchain.

This dissertation mainly summarizes my research on blockchain privacy and security protection issues throughout recent years. We first summarize the security and privacy vulnerabilities in the mining pools of traditional bitcoin networks and some possible protection measures. We then propose a new type of attack: coin hopping attack, in the case of multi-

ple blockchains under an IoT environment. This attack is only feasible in blockchain-based IoT scenarios, and can significantly reduce the operational efficiency of the entire blockchain network in the long run. We demonstrate the feasibility of this attack by theoretical analysis of four different attack models and propose two possible solutions. We also propose an innovative hybrid blockchain crowdsourcing platform solution to settle the performance bottlenecks and various challenges caused by privacy, scalability, and verification efficiency problems of current blockchain-based crowdsourcing systems. We offer flexible task-based permission control and a zero-knowledge proof mechanism in the implementation of smart contracts to flexibly obtain different levels of privacy protection. By performing several tests on Ethereum and Hyperledger Fabric, EoS.io blockchains, the performance of the proposed platform consensus under different transaction volumes is verified.

At last, we also propose further investigation on the topics of the privacy issues when combining AI with blockchain and propose some defense strategies.

INDEX WORDS: Blockchain, Privacy / Security, Consensus Protocol, Crowdsoursing, IoT.

PRIVACY PRESERVATION & SECURITY SOLUTIONS IN BLOCKCHAIN NETWORK

by

Saide Zhu

A Dissertation Submitted in Partial Fulfillment of the Requirements for the Degree of

Doctor of Philosophy

in the College of Arts and Sciences

Georgia State University

2021

PRIVACY PRESERVATION & SECURITY SOLUTIONS IN BLOCKCHAIN NETWORK

by

Saide Zhu

Committee Chair: Wei Li

Committee: Zhipeng Cai

Donghyun Kim

Kai Zhao

Electronic Version Approved:

Office of Graduate Studies

College of Arts and Sciences

Georgia State University

August 2021

DEDICATION

This dissertation is dedicated to my parents Yunhe Zhu and Yu'e Gu, my wife Anqi Lu for their endless support and love during my Ph.D. years. I cannot finish my Ph.D. without their love and encouragement.

ACKNOWLEDGEMENTS

Firstly, I would like to express my sincere gratitude to my advisors Prof. Wei Li and Prof. Zhipeng Cai for the continuous support of my Ph.D study and related research, for their patience, motivation, and immense knowledge. Their guidance helped me in all the time of research and writing of this thesis. I could not have imagined having a better advisor and mentor for my Ph.D study.

Besides my advisors, I would like to thank the rest of my thesis committee: Prof. Donghyun Kim, and Dr. Kai Zhao, for their insightful comments and encouragement, but also for the hard question which incented me to widen my research from various perspectives.

My sincere thanks also goes to professors and colleagues in my group and department. I would like to show my great gratitude to Prof. Yingshu Li, Prof. Dongjing Miao, Prof. Zhuojun Duan, Prof. Zaobo He, Prof. Meng Han, who offer their great help in my Ph.D. study. I thank Ms. Tammie Dudley, Ms. Adrienne Martin, Mr. Paul Bryan, Ms. Venette Rice, and Ms. Celena Pittman for their patient help, which makes my study at our department much easier and more convenient.

Special thanks for my group colleagues Akshita Maradapu Vera Venkata Sai, Zuobin Xiong, Jishen Yang, Kainan Zhang, Chenyu Wang, Bingyi Xie and Honghui Xu. Without their precious support it would not be possible to conduct this research.

Last but not least, I would like to thank everybody who made the dissertation possible, as well as express my apologies that I could not mention personally one by one.

FUNDING ACKNOWLEDGEMENTS

This dissertation was supported in part by the NSF Grant NOs, 1912753, 1704287, 1741277, 1829674, 2011845.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

- PPS - Pay-Per-Share
- PPLNS - Pay-Per-Last-N-Shares
- IoT - Internet of Things
- VSS - Verifiable Secret Sharing
- PVSS - Publicly Verifiable Secret Sharing
- DPOS - Delegated Proof of Stake
- PBFT - Practical Byzantine Fault Tolerance
- FL - Federated Learning
- TPS - Transaction Per Second
- POW - Proof of Work
- POS - Proof of Stake
- SPV - Simplified Payment Verification
- ZK-SNARK - Zero-Knowledge Succinct Non-Interactive Argument of Knowledge
- DHT - Distributed Hash Table

CHAPTER 1
INTRODUCTION

1.1 Background and Motivation

Privacy is a hot and heterogeneous concept, usually referring to the protection of information that an entity does not want to be known to outsiders. In fact, everyone has something to hide. With the ever-increasing amount of information, people started to focus on information security and protection of personal data privacy during big data exchange. A number of privacy-preserving mechanisms have been proposed. Here I listed some works that benefit this dissertation [2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18].

Starting with privacy in everyday life, Figure.1.1 shows some pop-up windows on our mobile phones that most of us here must have encountered. Are you willing to accept some programs to access personal information in your phone, such as your location information, photo information or your contact info? Of course, if you don't allow it, you may not be able to enjoy some of the services that these apps bring to you. And if you agree, you may worry about whether your personal privacy will be leaked or abused.

Are these companies deserve to be trusted? If you search with the keyword privacy, you may see a large number of such privacy leaks. For example, users could not access Wikipedia data from the server due to 8 service outrages in 2018 [19]. In such a centralized system where a server controls all the transactions, the issue of the controller's silently misbehave likely to occur without effective detection. Also, during the procedure of task assignment, the sensitive information (*e.g.*, location and preference) of users may be revealed by the public.

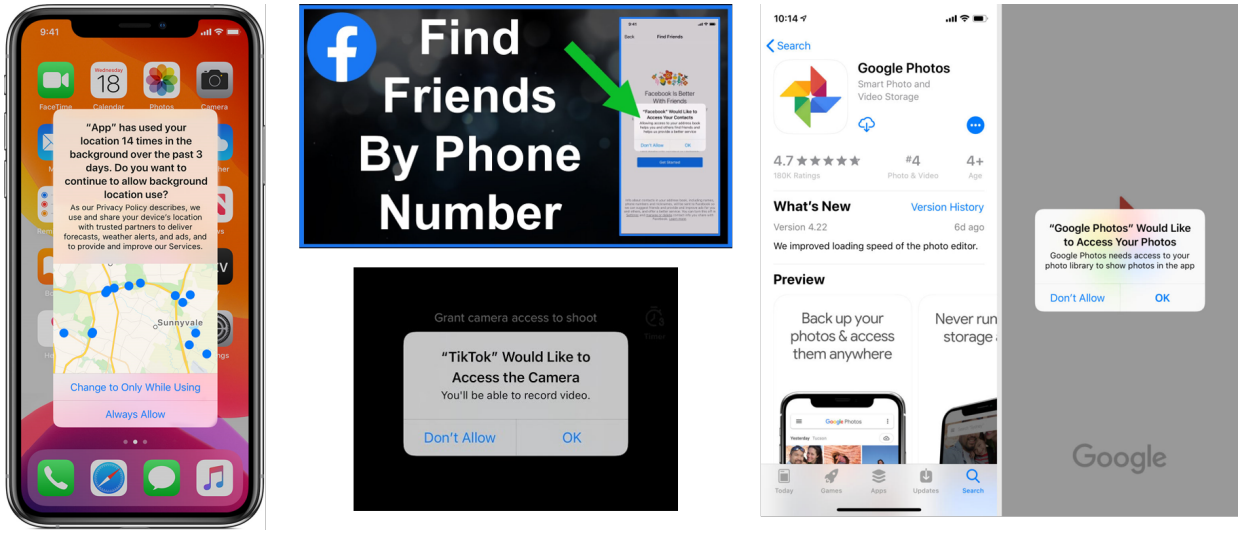


Figure 1.1: Mobile apps want to access your personal information

For example, in Nov. 2017, 57 million Uber driver's information has been hacked [20]. Some companies even sell users' personal information to other companies to make a lot of profits. So, the traditional centralized systems may suffer from the single point of failure.

To solve these issues, a number of countermeasures have been proposed [21, 22, 23, 24, 25, 26, 27, 28, 29].

Zhang *et al.* formulate the distributed bike trip selection modeling problem. They consider the total bicycle trip includes three portions, the way from source to the bike station, the bike trip, and the way from bike station to the destination. This consideration is much more practical than existing works, which just think about the route between bicycle stations. They define the Bike Trip Selection (BTS) issue with the goal that the client can choose the bicycle trip in a deterministic manner rather than a probabilistic one. And they formulate the BTS problem as a game, mapping it to a symmetric network congestion game and proposing

the algorithm for the game to find a suitable route for clients distributively [30]. However, there still a lack systematic management in the coordination of distributed networks. Duan *et al.* proposes a novel distributed MCS framework which jointly optimizes pricing and task allocation [31]. They propose to solve the problem through double decomposition and design a low-complexity iterative algorithm, which ensures that task initiators and mobile users maximize their payoffs. Thus, this methodology accomplishes social welfare maximization and is accordingly drastically different from traditional methodologies. Due to the reason that the task initiators and mobile users do not need to expose the utility and cost functions during task allocations, the algorithm can achieve privacy preservation. However, it lacks the protection of personal information in sub-area.

Besides the above solutions for distributed solutions, in this article, we mainly focus on the solutions using blockchain technology. In 2008, Satoshi Nakamoto proposed a blockchain-based digital money(Bitcoin) peer-to-peer trading platform [32]. In the Bitcoin network, the users that participate in approving and verifying the correctness of Bitcoin transactions are called miners. All the miners need to participant in transaction validation, and the validated transactions form a distributed public ledger that can prevent users from making a double-spending of their Bitcoins. With the generation of blockchain technology, the distributed network architecture can provide a decentralized solution for the traditional centralized platform.

However, the exceptionally critical bottleneck of the Bitcoin blockchain is currently how to improve its scalability. In view of verifiable information, the Transaction Per Second

(TPS) of BTC is around 4-7. As Visa's TPS is ordinarily around 4000, Bitcoin is right now clearly unfit to meet huge scope exchange situations. Also, to guarantee the security and privacy of the blockchain, the POW-based blockchain is required to have huge computational power. As the transactions get congested, miners need to wait for multiple confirmations to ensure the blockchain is secure from double-spending problems. The most instinctive thought is to expand the size of the block and shorten the interval between blocks. However, the main drawback of expanding block size is that it requires more extra storage on the chain. At the same time, it will cost additional time to spread to the whole network, which will lead to more forks and orphaned blocks in the blockchain network. The recent big progress of blockchain technology enables the seamless integration of smart contracts and novel cryptographic tools into blockchain networks, which provides an innovative way to improve privacy.

So, the goal of our work is to design a blockchain-enabled crowdsourcing platform to provide users with diverse privacy protection while improving efficiency. So the whole structure of the dissertation starts from understanding the current blockchain system and the privacy vulnerability of mining pools, and then further proposes the possible security and privacy issues. We propose an attack method called Coin Hopping in the IoT environment. Finally, we adopts both public chain verification and subchain verification, combined with smart contract and zero-knowledge proof to maximize users' access control and privacy protection for submitting data. Compared with existing hybrid blockchain architectures, this work reduces the reliance on relay bridge and proposes a common election mechanism for the

DPOS consensus of the public chain and PBFT consensus of the sub chain, so that the validator on the subchain can be dynamically generated and released when there is a demand, which reduces the waste of resources for useless channel monitoring. We test and compare the performance of DPOS, PBFT, and Casper consensus protocols in terms of execution, latency, and throughput in three different blockchain networks, Eos.io, Ethereum, and Hyperledger Fabric, respectively. The data shows that DPOS outperforms Ethereum's existing consensus by a factor of around 30 for large transaction volumes of 5,000-10,000, and PBFT outperforms it by a factor of 3 for small transaction volumes of 10-5,000, which verifies the reasonableness and feasibility of using different consensus in the proposed hybrid blockchain architecture. To our best knowledge, it has groundbreaking significance in the research of blockchain-based crowdsourcing. Finally, we focus on the privacy problems when combining blockchain with AI. On the basis of uploading local parameters in Federated Learning, we added pairwise masks and individual masks to further protect the end users' data privacy. And we use the non-interactive PVSS technology to solve the user's drop out situation. Our experiment results show a trade-off between privacy protection level and efficiency.

1.2 Dissertation Organization

The rest of the dissertation is organized as follows: Chapter 2 summarizes the background and related literature; A new attack strategy in IoT-based Blockchain networks is introduced in Chapter 3. We analyse the feasibility of that attack and propose some defense directions. Chapter 4 investigates the problems of blockchain-based crowdsoursing, and propose a hybrid

blockchain architecture specifically designed for crowdsoursing tasks to provide dynamic data access controls and diverse privacy protections. Chapter 5 investigates the privacy solutions when combining blockchain with federated learning. Chapter 6 provides future directions and Chapter 7 concludes our work.

CHAPTER 2
BLOCKCHAIN TECHNOLOGY PRELIMINARIES

Blockchain is a term used in the field of information technology. In essence, it is a shared database in which data or information is stored with characteristics such as "unforgeable", "traceable", "traceable", "open and transparent" and "collectively maintained". "open and transparent" and "collectively maintained". Based on these features, blockchain technology has laid a solid foundation of "trust" and created a reliable "cooperation" mechanism, which has a broad application prospect. In this chapter, we introduce some of the terminology in blockchain and the various basics that will be used in the following research.

2.1 Nacamodo Blockchain

Bitcoin is the first decentralized digital currency in the world. All the nodes in the network are involved in validation of transactions. The validated transactions form a public ledger, which does not allow participants to clarify a double spending of one's Bitcoin. Nakamoto consensus, considered to be one of Bitcoin core innovations, uses a challenging computational puzzle to determine the owner of next state block [32]. Nodes that participate in approving And verifying the correctness of Bitcoin transactions are called miners. The process of verifying transactions is also the process of mining Bitcoins. Specifically, the miner who successfully calculates the result will get a 12.5 Bitcoin (BTC) as the block Reward.

2.2 Pool Mining

Any miner in Bitcoin network who wants to generate a new block in the blockchain must figure out a computational puzzle. The computational puzzle in Bitcoin performs a double SHA-256 computation for each block. As the following inequality shows, every miner will be asked to compute an output value less than a threshold which indicates the difficulty of the network [32].

$$sha256(Merkel + sha256(PreBlock) + Nounce) \leq D \tag{2.1}$$

Technically, in order to get block reward, each miner has to generate a random nonce value to satisfy the requirement of the computational puzzle. The randomness feature of SHA-256 computation in the puzzle makes it nearly impossible for miners to take the inverse operation. Even a slight change of the input can result in a completely different output. Hence, the only approach for miners to find the nonce value is exhaustion, which consumes a lot of computational power. The difficulty of the computational puzzle dynamically changes in Bitcoin network. In particular, for every 2016 block mined, the network adjusts the difficulty, thereby regulating the average time per mining out a block is 10 minutes [33].

Mining pool is a necessary infrastructure for Bitcoin and other virtual cryptocurrencies. The meaning of its existence is to enhance the stability of Bitcoin mining and stabilize miners' revenues. Over the past three years, there has been a dramatic increase of computational power in Bitcoin network [34]. The difficulty of the network grows fast correspondingly, which

causes a great variance of revenues in solo mining. Thus, most miners choose to gather their computation power together to raise their possibility of finding a new block. Once a block is successfully mined in the pool, the pool manager fairly distributes the reward to the participants corresponding to their expended computation effort. Hence, the miners in the pool can get a much stable revenue than performing solo mining.

For example, the Bitcoin network hash rate on Jun 17, 2017, is 5094526985 GH/s [33]. Suppose someone gets a powerful Antminer S9 mining equipment which has a 13.5 TH/s hash rate. It still takes him 7.2 years to mine a block (get the 12.5 BTC reward) in average by solo mining. However, if he uses the Antminer S9 to join in the pool with 15% of the total network computational power, the revenue would be $13.055 per day without considering the pool service fee. That is the reason why most miners choose to mine in a pool.

If a miner in the pool finds a solution of the new block, the POW mechanism ensures he can not reward himself. When someone changes the public key of the coinbase transaction in Merkle tree, it will cause the nonce value does not match with the Block ID he generated. Hence, the block will not pass through the verification process of other miners. The feature ensures all the reward could only be distributed by pool manager.

2.2.1 Share

Share is the minimum workload as defined in the pool. In order to share the reward of BTC, a miner should at least submits one share to the pool manager, and the amount of the reward he can earn is determined by the distribution mechanism the pool adopts. As we discussed above, the computational puzzle asks miners to figure out a value less than the

network difficulty.

2.2.2 Full Solution

Once a share meets the difficulty of Bitcoin puzzle, this share is considered as a full solution. With this full solution submitted, pool manager then can get the 12.5 BTC block reward and distribute it to all the participants. In most cases, the expected reward of submitting a full solution is the same as submitting a share.

Specifically, the difficulty of Bitcoin mining can be expressed by the number of leading zeros of the blockhash. For example, for the 500155 block, there are 73 leading zeros, so the difficulty of this puzzle is 2^{73}. However, when a miner is mining in a pool, he will be given an easier difficulty which generated by pool manager. For instance, he only needs to find out a solution with 50 leading zeros. Each solution that meets the pool difficulty is called a share. Thus, in the above example, finding a share is 2^{23} times easier than finding the original solution. By submitting shares, the pool can verify how much work participants have done.

2.3 Reward Distribution Mechanism

The reward distribution mechanisms determine how pools assign rewards to their participants. Right now, ten biggest pools which have largest hashrates possess over 93% of total computational power [35], and many of these pools adopt different distribution mechanisms, such as Pay-Per-Share(PPS), Slush Method, Pay-Per-Last-N-Shares, etc. When analyzing a reward system, the key factor need to be considered is the $fairness$. It is expected that

Name	Reward Mechanism	PPS fee	Other fee
Antpool	PPLNS & PPS	2.5%	0%
BTC.com	FPPS	0%	4%
Slush Pool	Score		2%
BTCC Pool	PPS	2%	0%
P2Pool	PPLNS		0%
BTCDig	DGM		0%

Table 2.1: Reward Mechanism Statistics [1]

each submitted share deserves a similiar amount of rewards in one game, and the fluctuation of the payoff is not affected by the time factor. In 2016, Schrijvers *et al.* defined that the incentive-compatibility of a reward distribution system is when reporting full solution at once is miners best strategy [36]. The incentive-compatibility is considered to be a good evaluation metrics for analyzing the fairness.

All these Reward Distribution mechanisms are aimed to provide a fair and attractive distribution of rewards among pool participants. From a miners perspective, he wants to join a pool which can make his expected payoff to be stable and also as more as possible. From a pool managers view, he is intended to attract more participant to enlarge his computational power, at the same time lower the risk of getting bankrupt. Table 1 below shows the reward distribution mechanisms for some famous mining pools.

The most common mechanism adopted by open pools are PPS and PPLNS. This two mechanisms can be easily deployed in pools. Also, it is clear for participants in such pools to calculate their expected payout. Due to the stable revenue of PPS mechanism, it is generally known that operators in the PPS pool will draw relatively high fees. Thus, PPS pools give participants a feeling of stable but not superior. In order to change this situation, operators often introduce some unique elements into their allocation to attract more participants. For

example, AntPool uses a variety of distribution mechanisms at the same time for miners to choose; BTC.com uses Full pay per share (FPPS). This mechanism not only share regular dividends (12.5 BTC for now) but also some of the transaction fees, thus increasing miner's expected revenue. In future Bitcoin transactions, and the part of transaction fee will gradually increase. Therefore, FPPS can well adapt to the future development of Bitcoin pool mining.

For Slush pool and Geometric method, these two mechanisms are not widely implemented in Bitcoin system due to their obvious shortcomings. Some other mechanisms may combine several of above methods to let participants themselves decide how they want to get paid.

One big issue in reward distribution system nowadays is security. Existing works have shown a variety of attack strategies in pool mining, especially in those running PPS and PPLNS mechanisms.

2.4 Pool Hopping

Pool hopping is considered to be one of the most vulnerable weaknesses of the proportional block reward distribution method. According to [37], with such a proportional distribution mechanism, the longer a mining round is, the more shares submitted in a mining pool. In other words, every share in a longer round is worth less. Thus, rational miners choose to mine only when the expected reward is high and to leave when it is low. The problem of pool hopping is that if a rational miner who chooses to adopt hopping strategy can constantly earn a profit more than the expected reward, i.e., a rational miner can perform pool hopping

strategy for enhancing the received reward. As a result, in the mining pool, the honest miners who do not hope lose their deserved profits.

2.5 Merkle Tree

Merkle tree, which is a type of data structure (usually a binary tree and possibly a multi-fork tree), stores hash values of data contents. In a blockchain (*e.g.,* Bitcoin), Merkle tree is used to quickly summarize and verify the integrity of block data by hashing transactions. Each hash node is generated from two adjacent data nodes continuously and recursively, leaving only one Merkle root stored in the block headers.

The main advantages of using a Merkle tree in a blockchain system is that it greatly improves the efficiency and scalability of the blockchain, so that the block header only needs to contain the root hash value without having to encapsulate all the underlying data, which makes the verification run efficiently. Also, in the blockchain, the Merkle tree supports the Simplified Payment Verification Protocol (SPV), which allows transaction data to be executed without running a full node. Thus, the payment confirmation problem can be solved under light client conditions.

2.6 Smart Contract

The smart contracts were firstly deployed on Ethereum using *Solidity* language. In the smart contracts, the codes will be automatically executed when a condition is triggered by the contract contents, reducing the cost of communication and supervision. Since the smart contracts are implemented on a blockchain, they possess some critical features in-

cluding accountability and decentralization. Also, the blockchain can guarantee that the contents of the smart contracts cannot be changed. When it comes to crowdsourcing, the blockchain platforms are expected to process data and control access permission for various tasks. Therefore, the smart contracts can be employed as a programmable interface for developers to set up different procedures.

2.7 Zero-Knowledge Proof

Protecting user privacy is an important focus in the blockchain. By using zero-knowledge proof (ZK-proof), private information can be preserved during the verification process [38, 39]. In our work, Zero-Knowledge Succinct Non-Interactive Argument of Knowledge (ZK-SNARK) is utilized to construct optimized ZK-proof to help verify transactions in the communications between the public chain and the subchains. Typically, the ZK-SNARK algorithm has three components: (1) a setup algorithm is run to produce a public parameter to establish the SNARK; (2) an attestation is generated by a prover and sent to the SNARK verifier on the blockchain; and (3) the correctness of the proof is verified via calling the verifier module.

CHAPTER 3
COIN HOPPING ATTACK IN BLOCKCHAIN-BASED IOT

3.1 Introduction

Blockchain (e.g., Bitcoin and Ethereum) has experienced explosive progress in the past years, and its usage today is far beyond cryptocurrency. Thanks to its unique features, including openness, transparency, secure communication, difficulty in falsification, and multi-consensus, blockchain is treated as one of the most promising technologies to promote many emerging applications, such as Internet of Things (IoT), Big Data, and Smart Cities [40]. The virtue of blockchain is to verify and record transactions by users (i.e., block miners) in a distributed and secure manner; especially, transaction verification is accomplished through block mining process. Due to the extremely high mining difficulty, miners are likely to form mining pools to together computational power and share rewards for improving mining success rate as well as enhancing expected revenue [37] – *such mining pool has become an indispensable important part of a blockchain network.*

Meanwhile, in order to pursue more profits, malicious miners utilize the flaws of reward distribution mechanisms to manipulate their mining behaviors, leading to revenue reduction to other honest miners and severe impacts on the involved mining pool(s) – this is so-called pool mining attack. To prevent miners' malicious behaviors, researchers have performed a lot of efforts on the analysis of pool mining attacks, including pool hopping attack [37], selfish mining attack [41, 42], and block withholding attack [35, 43, 44]. *No stopping here, in the combination of blockchain and IoT, a new and more sly type of pool mining attack*

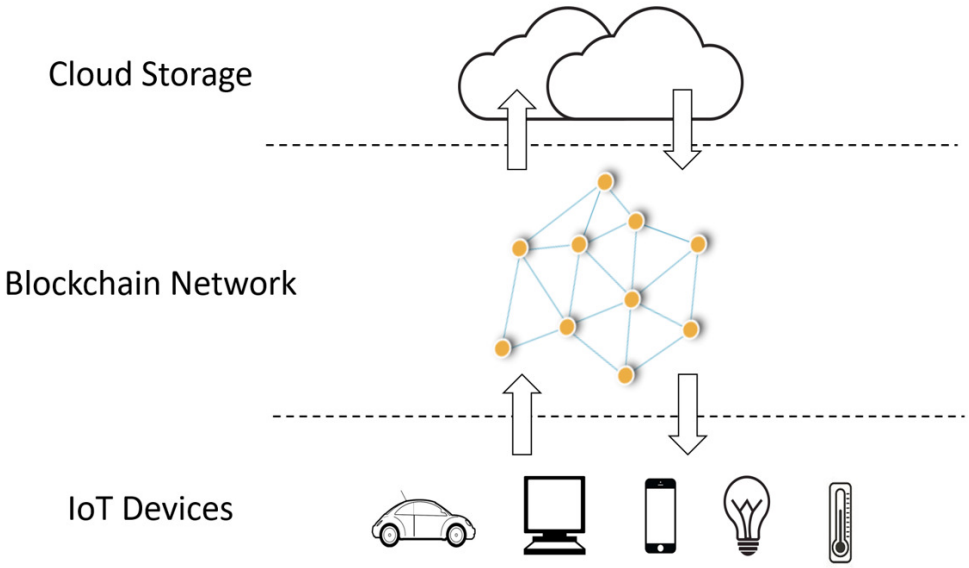


Figure 3.1: Blockchain-based IoT Framework

(termed **coin hopping attack***) emerges, which unfortunately, has never been addressed in literature.*

Before explaining coin hopping attack, an overview on the blockchain-based IoT is introduced. Currently, the existing blockchain-based IoT architectures are designed based on a typical three-layer hierarchical network framework, containing IoT devices, blockchain network, and cloud storage [45, 46, 47, 48, 49, 50], which is shown in Fig. 3.1. Compared with the traditional cryptocurrency blockchain network, the blockchain-based IoT has several unique characteristics. First of all, besides the specific mining equipments (e.g., ASIC devices [51]) of the traditional blockchain network, IoT devices can be also employed as miners for improving device utilization and mining rate. Second, as the blockchain-based IoT is expected to embrace various networks, *mining compatibility should be achieved*, which

requires that (some) IoT miners are able to run different consensus protocols for different blockchain networks. But, on the other hand, mining compatibility makes it *hard or even impossible for these IoT miners to judge which consensus protocols they are running.* These unique characteristics provide a pool manager with more flexibility and freedom to maliciously transfer the mining power of his pool(s) among different blockchain networks for extra profit, and this malicious behavior is named **coin hopping attack**.

Motivated by the above observations, in this chapter, we propose an in-depth study on coin hopping attack. Since no work has been done for coin hopping attack, our study faces some challenging problems.

1. The first one is *how to analyze coin hopping attack.* More specifically, the formal definition of coin hopping attack, the attack scenarios, and the pool manager's behaviors should be studied.

2. To better understand coin hopping attack, we need to solve the problem: *when dose coin hopping attack occur?* In other words, the condition of successfully launching coin hopping attack should be identified.

3. If the blockchain-based IoT suffers coin hopping attack, *what are the impacts on miners and networks?*

In order to answer these problems, our study starts from defining the concept of coin hopping. Then, a rigorous theoretical analysis is carried out to investigate coin hopping attack under different pool mining scenarios, including one pure pool, two pure pools, one

mixed pool, and merged mining pool. What's more, in each scenario, the conditions to launch a successful coin hopping attack are proved. Finally, the impacts and defense directions are analyzed. In this Chapter, our multi-fold contributions are summarized below.

- To the best of our knowledge, this is the first work to study coin hopping attack, thereby filling the blank in literature.
- The feasibility of coin hopping attack is rigorously proved, which indicates the vulnerability of blockchain-based IoT to coin hopping attack.
- The conditions in various pool mining scenarios to perform coin hopping attack are identified, which builds a theoretical foundation for future research of coin hopping.
- The multi-dimensional consequences of coin hopping attack are thoroughly investigated from the viewpoints of miners and networks.
- To resist coin hopping attack, two defense directions are proposed.

The rest organization of this Chapter is as follows. The framework of the blockchain-based IoT is described in Section 3.2. The preliminaries of Bitcoin and Ethereum incentive mechanisms are illustrated in Section 3.3. The theoretical analysis of coin hopping attack in various scenarios are conducted in Sections 3.4 and 3.5. Finally, this chapter is concluded in Section 4.6.

3.2 Analytical Model and Framework

Blockchain can work as a public ledger to record every transaction in a secure manner. By applying blockchain technology into IoT, a centralized entity is no longer necessary as devices can communicate securely in a distributed manner without single point of failure. In such application, there are three key benefits, including *building trust*, *reducing cost*, and *accelerating transaction* [52].

Recently, the combination of blockchain and IoT has attracted more and more attention from both industry and academic. For examples, the Autonomous Decentralized Peer-to-Peer Telemetry (ADEPT) project incorporated blockchain into their distributed IoT architecture to achieve autonomous device coordinating, peer-to-peer messaging, and distributed file sharing [45]; Zyskind *et al.* proposed a blockchain-based IoT architecture consisting of device & service, blockchain network and DHT(distributed hash table), which can guarantee secure interaction among distributed IoT devices as well as ensure identity access control [46]; similarly, Ali Dorri *et al.* designed an architecture with smart home, overlay-blockchain network and cloud storage [47] to increase the security and accessibility level of IoT network. To sum up, these blockchain-based IoT architectures are built based on a common three-layer hierarchical framework as shown in Fig. 3.1, where IoT devices (i.e., IoT miners) can communicate and interact through blockchain networks and the information is stored in clouds. The framework is adopted in the analysis of this Chapter.

Since the superiority of IoT mainly lies in the ubiquitous connectivity of various networks, such as smart homes, smart grids, and smart vehicular networks, the blockchain-based IoT

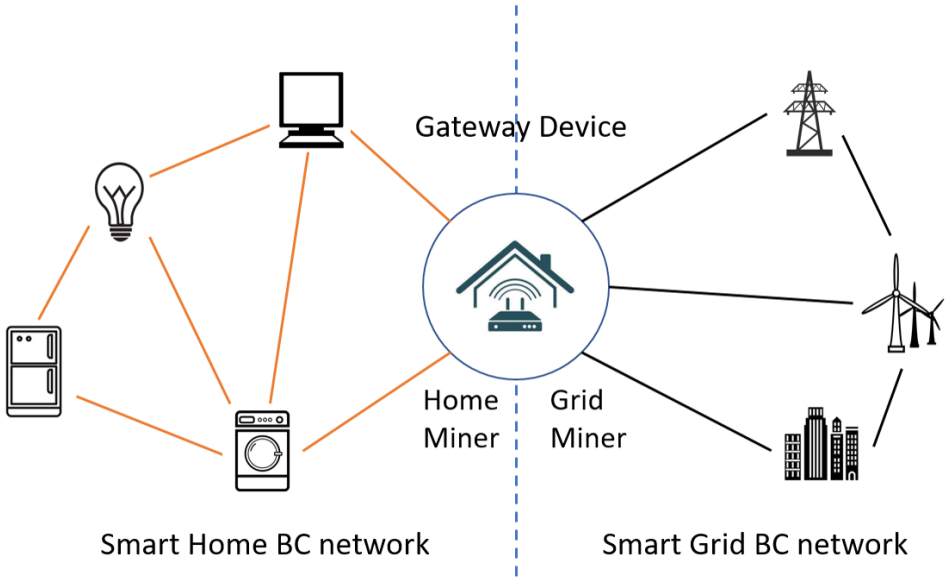


Figure 3.2: IoT devices serve for different blockchain networks

should achieve *mining compatibility* to carry forward this superiority, which can be demonstrated from the following three aspects. On one hand, various consensus protocols as well as cryptocurrencies can co-exist in the blockchain-based IoT, because different blockchain networks may adopt different consensus protocols. On the other hand, to improve resource utilization and reduce mining cost, (some) smart devices in IoT can serve as miners for different blockchain networks, because a smart device may belong to different networks.

An illustrative example is presented in Fig. 3.2 where the smart home miners are able to integrate with the homes Internet gateway or a separate stand-alone devices [47, 51]. Due to the computational power these devices have, they are also suitable to offer verification service for smart grid [53].

3.3 Incentive Mechanisms in Bitcoin and Ethereum

The amount of a pool's revenue is directly proportional to the amount of computational power it has. A pool manager usually draws a certain percentage of the reward as a handling fee, which is the pool manager's income. Different consensus mechanisms determine different reward mechanisms [37]. For example, since the Bitcoin network runs the POW consensus [32] and Ethereum runs the Casper consensus [54], the puzzles in their networks are different. Therefore, different formulas are used to calculate the expected revenue. Besides, the reward distribution mechanism of the mining pool can affect the calculation of the pool manager's income. Currently, there are various blockchain-based cryptocurrencies, including Bitcoin [32], Ethereum [55], EOS [56], Litecoin [57], Bitcoin Cash [58], and so on.

As Bitcoin and Ethereum are two of the most popular crypto-currencies, we use them as the examples in our analysis to illustrate the security vulnerability of the blockchain-based IoT. In our analysis, the common PPS/PPLNS allocation mechanism is adopted, in which the expected probability of successfully mining a block by the pool is equal to the ratio of the mining power of the pool to the entire network power [37]. Indeed, such analysis also has essentially explanatory power in a more general scenario where other different blockchain-based cryptocurrencies co-exist.

3.3.1 Bitcoin Incentive Mechanism

Assume that P represents the ratio of the mining power owned by a pool to the total Bitcoin mining power in the network. Suppose B_{BTC} is the Bitcoin block reward (e.g., 12.5 BTC

per new block mined in the current network [59]). Accordingly, for each new block mined in the network, the expected reward of the pool is $P \cdot B_{BTC}$. Let f denote the pool fee, S_{BTC} be the frequency of mining a new Bitcoin block, and U_{BTC} represent the expected revenue of the pool in the Bitcoin network. Thus, U_{BTC} can be computed in Eq. (3.1).

$$U_{BTC} = P \cdot B_{BTC} \cdot S_{BTC} \cdot f. \tag{3.1}$$

3.3.2 Ethereum Incentive Mechanism

The incentive mechanism of Ethereum is different from that of Bitcoin. In Ethereum, a new block may be either a normal block or an uncle block. The uncle block is an alternative block at the same height as the parent of a block and should be considered to be orphaned in the future because it is not on the longest chain [55].

Accordingly, when analyzing the Ethereum block reward, the first consideration is whether the new block is a normal block or an uncle block, because their revenue calculations are different in the Ethereum network. Suppose p is the probability that a new block is a normal block, and then $1 - p$ is the probability that a new block is an uncle block. Formally, let U_{NETH} and U_{UETH} denote the expected revenues of a normal block and an uncle block, respectively. Then the expected revenue for a newly mined Ethereum block U_{ETH} can be calculated as follows:

$$U_{ETH} = p \cdot U_{NETH} + (1 - p) \cdot U_{UETH} \tag{3.2}$$

For a normal Ethereum block, the total block reward contains three parts. (i) The first

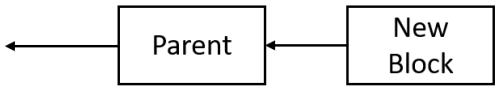
part is *a basic block reward* B_{ETH} with the amount of exact 3 Ethers at present. (ii) The second part is the gas fee, which could be treated as the transaction fee in Bitcoin block reward. Since the amount of gas fee of each block varies in network environment and is relatively a small portion compared with the block reward, the gas fee is not considered in our revenue analysis. (iii) The third part is *an extra block reward* per uncle block generated. If the uncle blocks exist, an extra amount of $\frac{1}{32}$ block reward per uncle is provided. For each normal block, the maximum of 2 uncles are allowed [55].

In the calculation of the total block reward for a normal block, three cases need to be analyzed, which are presented in Fig. 3.3. For each new normal block, the miner can always obtain the basic block reward B_{ETH}. If the new normal block has an uncle, the miner has a probability of $1-p$ to receive an additional $\frac{1}{32}$ block reward. Similarly, when there are two uncles as shown in Fig. 3.3.(c), the miner can get an additional $\frac{1}{32}$ block reward with $(1-p)^2$ probability. Thus, the expected revenue of mining a normal block U_{NETH} can be computed in Eq. (3.3)

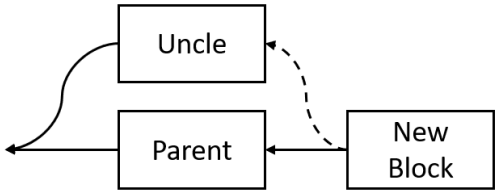
$$U_{NETH} = P \cdot B_{ETH} \cdot [1 + \frac{1-p}{32} + \frac{(1-p)^2}{16}] \cdot S_{ETH} \cdot f, \tag{3.3}$$

where P the mining power of the pool and S_{ETH} is the frequency of mining a new block in the Ethereum network.

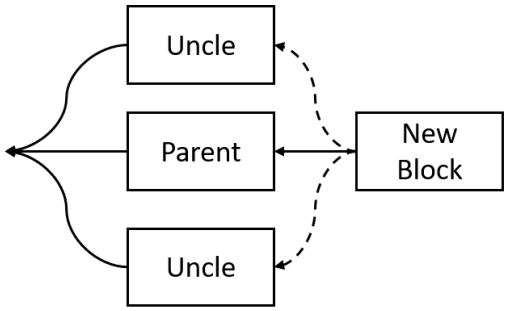
If a new block is treated as an uncle block, the miner could receive the uncle block reward B_{UETH}. Currently, $B_{UETH} = \frac{3}{4}B_{ETH}$ in average [60], and the expected revenue of mining an



(a) Situation 1: basic block reward



(b) Situation 2: with one uncle block



(c) Situation 3: with two uncle blocks

Figure 3.3: Three situations for mining a new normal block

uncle block can be computed via Eq. (3.4).

$$\begin{aligned} U_{UETH} =& P \cdot B_{UETH} \\ =& \frac{3}{4} P \cdot B_{ETH}. \end{aligned} \tag{3.4}$$

Thus, the expected revenue of mining a new Ethereum block is:

$$\begin{aligned} U_{ETH} =& p \cdot U_{NETH} + (1-p) \cdot U_{UETH} \\ =& P \cdot [p \cdot B_{ETH} \cdot (1 + \frac{1-p}{32} + \frac{(1-p)^2}{16}) \\ & + (1-p) \cdot B_{UETH}] \cdot S_{ETH} \cdot f. \end{aligned} \tag{3.5}$$

3.4 Coin Hopping Analysis

Coin Hopping references to the malicious behavior of a pool manager who uses its miners' computational power to verify another higher rewarded blockchain without informing the miners in its mining pool(s).

In the blockchain-based IoT, a pool manager may take charge of one or multiple mining pools in different blockchain-based networks. Suppose a mining pool consisting of multiple gateway devices is currently working for the Bitcoin-based smart home network. If the pool manager finds that the Ethereum-based smart grid network can provide a higher reward for transaction verification. The manager could transfer all the mining power of the pool in the smart home network to work for the smart grid network while using the smart home network standard to reward those gateway devices in his pool. By doing this, the pool manager can probably earn more benefit. This strategy of maliciously transferring mining power is called *Coin Hopping Attack*.

Such coin hopping attack may not occur in the traditional blockchain networks due to the fact that the miners in most mining pools are ASIC devices [61], which usually can deal with only one specific kind of blockchain puzzle. Thus, when the pool manager performs

coin hopping attack, the miners within the pool cannot help verify any blockchain that adopts another different consensus protocol. For example, the ASIC miner in a Proof of Work (PoW)-based blockchain network can only solve the puzzle that includes exact two SHA-256 operations but has nothing to do with a Proof of Stake (PoS)-based blockchain. Thus, when the pool manager try to conduct the coin hopping attack to another blockchain running different consensus protocol, ASIC miners could always know it happens. However, to achieve compatibility throughout the blockchain-based IoT, smart devices, such as smart phones and smart bulbs, are employed as miners to deal with various blockchain puzzles. Meanwhile, some issues are brought in: (i) these smart devices typically lack the ability to sense a pool manager's behavior due to the limitation of computational power; and (ii) although the miners can know whether the pool manager has received other blockchain rewards by checking the public record of blockchain information, they have no way to judge whether they have contributed verification effort. These issues provide the pool manager with great freedom to maliciously transfer the miners' computational power for verifying transactions from other blockchains without the miner's permission. In other words, the blockchain-based IoT is vulnerable to coin hopping attack.

In following analysis, we consider three different scenarios where coin hopping attack happens between Bitcoin blockchain and Ethereum blockchain. First, a single pool that currently working for only one blockchain network is studied in Section 3.4.1. Next, we analyze the case when the attacker has two pools with one working for Bitcoin blockchain and the other working for Ethereum blockchain in Section 3.4.2. Finally, a more realistic

Table 3.1: Notation Table

Notation	Description
m	Mining power of Ethereum network
α	Mining power of pool A with $\alpha \in [0, 1]$
β	Mining power of pool B with $\beta \in [0, 1]$
γ	Pool mining power rate works for the Bitcoin network
B_{BTC}	Reward of mining each Bitcoin block
B_{ETH}	Reward of mining each Ethereum block
B_{UETH}	Reward of mining each Ethereum uncle block
S_{BTC}	Frequency of mining a new Bitcoin block
S_{ETH}	Frequency of mining a new Ethereum block
f	Pool fee percentage
p	Probability of being an Ethereum normal block
U_{BTC}	Expected revenue of P_{att} in Bitcoin network
U_{ETH}	Expected revenue of P_{att} in Ethereum network
k	Conversion rate between Bitcoin and Ethereum block rewards

scenario where a single pool working for both different blockchain networks (i.e., a mixed pool) is investigated in Section 3.4.3. For a better presentation, the main notations are described in Table 3.1.

3.4.1 Single Pure-Pool

In the first scenario, we consider that a pool manager P_{att}, who owns a pool A in the Bitcoin blockchain network, is the attacker that can launch coin hopping attack. At the same time, there is another network using the Ethereum blcokchain for transaction verification in IoT. To earn more profits, P_{att} may choose part of his miners' computational power to verify the Ethereum network while using his Bitcoin deposit to award the miners under the standard of Bitcoin network.

As we consider P_{att} has enough deposit for different kinds of cryptocurrencies, a more extreme situation is that P_{att} privately transfers his total mining power from the Bitcoin network to the Ethereum network as presented in Fig. 3.4. Assume that the mining power

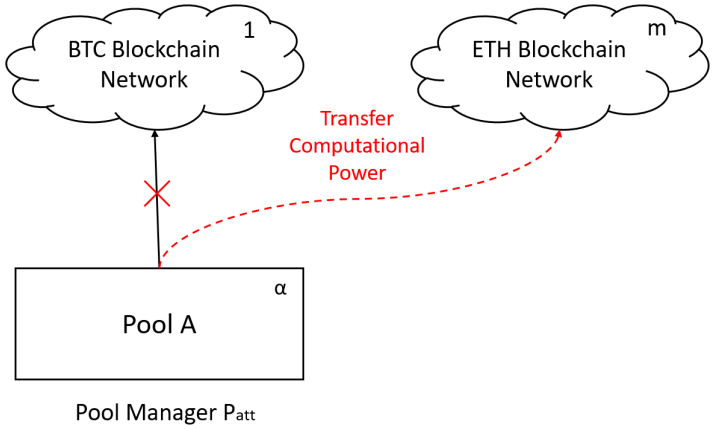


Figure 3.4: Coin hopping in a pure pool

of the entire Bitcoin network is 1. Since the probability that a pool mines a new block is equal to the ratio of the mining power of the pool to that of the entire Bitcoin network, the probability that P_{att}'s pool mines a new block is $\frac{\alpha}{1}$. By now, P_{att} has two available mining strategies: (i) honestly mining in the Bitcoin network and (ii) performing coin hopping attack to mine in the Ethereum network.

If P_{att} mines honestly in the Bitcoin network, the expected revenue of mining a new Bitcoin block can be calculated as follows:

$$U_{BTC} = \frac{\alpha}{1} \cdot B_{BTC} \cdot S_{BTC} \cdot f. \tag{3.6}$$

If P_{att} performs coin hopping attack by transferring all his mining power in pool A to the Ethereum network. He has a probability of $\frac{\alpha}{m+\alpha}$ to mine a new Ethereum block. Thus, we can use Eq. (3.7) to calculate the expected revenue of mining a block in the Ethereum

network.

$$\begin{aligned} U_{ETH} =& \frac{\alpha}{m+\alpha} \cdot [p \cdot B_{ETH} \cdot (1 + \frac{1-p}{32} + \frac{(1-p)^2}{16}) \\ & + (1-p) \cdot B_{UETH}] \cdot S_{ETH} \cdot f. \end{aligned} \tag{3.7}$$

To check whether attacker P_{att} can get a higher revenue through coin hopping, we subtract Eq. (3.7) from Eq. (3.6). As a new block is generated every 10 minutes in the Bitcoin network while a new block is verified every 12 seconds in the Ethereum network [55, 62], we have $S_{BTC} = \frac{S_{ETH}}{50}$. Also, the average uncle block reward in the Ethereum network is currently $\frac{3}{4}$ of a normal block reward, which indicates that $B_{UETH} = \frac{3}{4} \cdot B_{ETH}$ [60]. To identify the general relationship between Bitcoin block reward and Ethereum block reward, we assume $B_{BTC} = k \cdot B_{ETH}$ with $k > 0$ where k is the conversion rate between a Bitcoin block reward and an Ethereum block reward (i.e., the exchange rate between two different cryptocurrencies). Hence, the revenues of honest mining and coin hopping can be further represented as follows:

$$\begin{aligned} U_{BTC} =& \frac{\alpha}{1} \cdot B_{BTC} \cdot S_{BTC} \cdot f \\ =& \frac{\alpha \cdot k}{50} \cdot B_{ETH} \cdot S_{ETH} \cdot f. \end{aligned} \tag{3.8}$$

$$\begin{aligned} U_{ETH} =& \frac{\alpha}{m+\alpha} \cdot [p \cdot B_{ETH} \cdot (1 + \frac{1-p}{32} + \frac{(1-p)^2}{16}) \\ & + (1-p) \cdot B_{UETH}] \cdot S_{ETH} \cdot f \\ =& \frac{\alpha}{m+\alpha} \cdot \frac{2p^3 - 5p^2 + 35p + 24}{32} \cdot B_{ETH} \cdot S_{ETH} \cdot f. \end{aligned} \tag{3.9}$$

Note that p is the probability that a new Ethereum block is a normal block, and then we let $c = \frac{2p^3-5p^2+35p+24}{32}$ for computation simplicity. Thus, the revenue difference of the two mining strategies can be computed by Eq. (3.10).

$$U_{ETH} - U_{BTC} = (\frac{\alpha \cdot c}{m+\alpha} - \frac{\alpha \cdot k}{50}) \cdot B_{ETH} \cdot S_{ETH} \cdot f. \tag{3.10}$$

As $B_{ETH} > 0$, $S_{ETH} > 0$, and $f > 0$, we have $U_{ETH} - U_{BTC} > 0$ if Eq. (3.11) holds, which means the attacker P_{att} is able to earn more revenue through coin hopping attack.

$$(\frac{\alpha \cdot c}{m+\alpha} - \frac{\alpha \cdot k}{50}) \geq 0. \tag{3.11}$$

Furthermore, because $\alpha \in [0, 1]$, we can conclude that Eq. (3.11) holds as long as Eq. (3.12) can be satisfied. This indicates that the pool manager P_{att} can enhance his expected revenue via coin hopping attack if and only if Eq. (3.12) can hold.

$$\alpha \leq \frac{50 \cdot c}{k} - m. \tag{3.12}$$

Therefore, as a pool manager with mining power, he can choose to perform coin hopping attack when his mining power is equal or smaller than a certain threshold suggested in Eq. (3.12). In particular, the threshold is determined by the blockchain network environments, including the mining power of the entire blockchain (e.g., Bitcoin and Ethereum) network, the frequency of generating a new block in the blockchain network, the exchange rate between two different cryptocurrencies, and the probability of being a normal Ethereum

block (it also indicates the uncle block rate in the Ethereum network). According to the real-world statistics of current Bitcoin & Ethereum blockchain information in May 2018, the uncle block rate of the Ethereum network is around 17%. So, we have $p = 0.83$ and $c = 1.59$. The hash rate of the Ethereum network is 276,441.08 GH/s [60], and the hash rate of the Bitcoin network is 31,722,970.71 TH/s [33]. Then, the Ethereum network mining power is $m \approx \frac{1}{114755}$ when the mining power of the Bitcoin network is normalized to 1. The value of a Bitcoin block is about \$8000, and the value of an Ethereum block is about \$700 [63]. Thus, the block reward conversion rate is $k \approx 47.6$. By substituting this statistics, the condition of launching coin hopping attack is $\alpha \leq 1.67$, which indicates that this condition can be met by pool with any size. *Thus, in the blockchain-based IoT, it is feasible for a single-pool manager to transfer mining power from a Bitcoin-based network to an Ethereum-based network for revenue enhancement.*

Notice that in the blockchain-based IoT, coin hopping attack can also happen reversely from an Ethereum-based network to a Bitcoin-based network when $\alpha \geq \frac{50 \cdot c}{k} - m$.

3.4.2 Double Pure-Pool

The second scenario considers the pool manager P_{att} owns two pools A and B which possess α and β computational power, respectively. Suppose that pool A verifies transactions for the Bitcoin blockchain while pool B works for the Ethereum blockchain. Fig. 3.5 shows the attacker P_{att} can privately uses all the mining power of his two pools to verify the Ethereum network and simultaneously awards the miners in pool A under the Bitcoin standard.

Formally, U_A is used to represent the expected revenue of pool A and U_B is used to denote

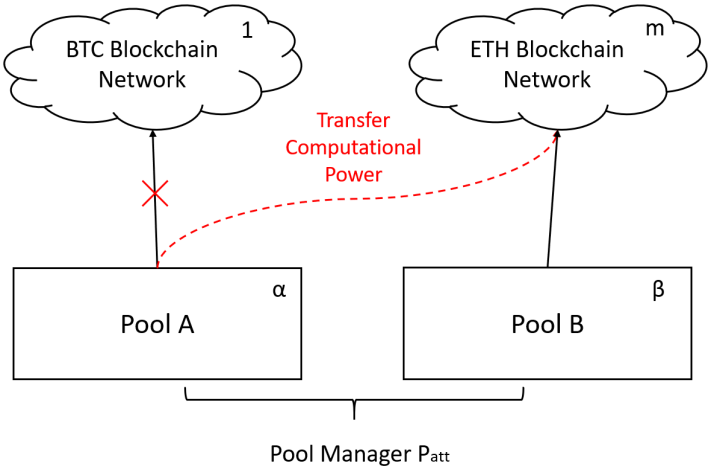


Figure 3.5: Coin hopping in two pure pools

the expected revenue of pool B, thereby the total expected revenue for the pool manager $U_{P_{att}}$ is $U_A + U_B$. If P_{att} mines honestly, the probability for pool A to mine a new block in the Bitcoin network is $\frac{\alpha}{1}$, the probability for pool B to find a new block in the Ethereum network is $\frac{\beta}{m}$, and the pool manager P_{att}'s total expected revenue can be computed by Eq. (3.13).

$$\begin{aligned} U_{P_{att}} =& U_A + U_B \\ =& \frac{\alpha}{1} \cdot B_{BTC} \cdot S_{BTC} \cdot f \\ &+ \frac{\beta}{m} \cdot [p \cdot B_{ETH} \cdot (1 + \frac{1-p}{32} + \frac{(1-p)^2}{16}) \\ &+ (1-p) \cdot B_{UETH}] \cdot S_{ETH} \cdot f, \end{aligned} \tag{3.13}$$

in which the mining power of the entire Bitcoin network is 1 and that of the entire Ethereum is m. For a better comparison, both the computation of U_A and Eq. (3.13) are rewritten as

the explicit expressions of B_{ETH} and S_{ETH}, which are respectively presented as follows.

$$\begin{aligned} U_A =& \frac{\alpha}{1} \cdot B_{BTC} \cdot S_{BTC} \cdot f \\ =& \frac{\alpha k}{50} \cdot B_{ETH} \cdot S_{ETH} \cdot f. \end{aligned} \tag{3.14}$$

$$\begin{aligned} U_{P_{att}} =& U_A + U_B \\ =& \frac{\alpha k}{50} \cdot B_{ETH} \cdot S_{ETH} \cdot f \\ &+ \frac{\beta}{m} \cdot [p \cdot B_{ETH} \cdot (1 + \frac{1-p}{32} + \frac{(1-p)^2}{16}) \\ &+ (1-p) \cdot B_{UETH}] \cdot S_{ETH} \cdot f \\ =& (\frac{\alpha k}{50} + \frac{\beta c}{m}) \cdot B_{ETH} \cdot S_{ETH} \cdot f. \end{aligned} \tag{3.15}$$

When P_{att} launches coin hopping attack towards pool A, all his mining power $\alpha + \beta$ are used for the Ethereum network. Therefore, P_{att} can mine a new block in the Ethereum network with a probability of $\frac{\alpha+\beta}{m+\alpha}$ and obtain the following expected revenue, defined as $U_{P_{att}}^{A \to ETH}$.

$$\begin{aligned} U_{P_{att}}^{A \to ETH} =& \frac{\alpha + \beta}{m + \alpha} \cdot [p \cdot B_{ETH} \cdot (1 + \frac{1-p}{32} + \frac{(1-p)^2}{16}) \\ &+ (1-p) \cdot B_{UETH}] \cdot S_{ETH} \cdot f \\ =& \frac{(\alpha + \beta) \cdot c}{m + \alpha} \cdot B_{ETH} \cdot S_{ETH} \cdot f. \end{aligned} \tag{3.16}$$

Then, if $U_{P_{att}}^{A \to ETH} \geq U_{P_{att}}$, the pool manager has sufficient motivation to perform coin hopping attack in pool A for revenue enhancement; that is, the pool manager P_{att} can increase his expected revenue by transferring mining power to the Ethereum network if Eq. (3.17)

can hold.

$$\begin{aligned} U_{P_{att}}^{A\to ETH} - U_{P_{att}} =&[\frac{(\alpha+\beta)\cdot c}{m+\alpha} - \frac{\alpha k}{50} - \frac{\beta c}{m}] \\ &\cdot B_{ETH}\cdot S_{ETH}\cdot f \\ \geq& 0. \end{aligned} \tag{3.17}$$

Since $B_{ETH} > 0$, $S_{ETH} > 0$, and $f > 0$, Eq. (3.17) can be equivalently simplified to the following equation.

$$[\frac{(\alpha+\beta)\cdot c}{m+\alpha} - \frac{\alpha k}{50} - \frac{\beta c}{m}] \geq 0,$$

which can be further simplified as Eq. (3.18).

$$\begin{aligned} &50m(\alpha+\beta)\cdot c - m(m+\alpha)\alpha k - 50(m+\alpha)\beta c \geq 0 \\ \Rightarrow &- mk\alpha^2 + (50mc - m^2k - 50\beta c)\alpha \geq 0. \end{aligned} \tag{3.18}$$

Because α is in the range $[0, 1]$, we have

$$\alpha \leq \frac{50mc - m^2k - 50\beta c}{mk}. \tag{3.19}$$

Thus, when Eq. (3.19) is satisfied, attacker P_{att} can earn more profits via performing coin hopping attack towards pool A. More specifically, the value of $\frac{50mc-m^2k-50\beta c}{mk}$ is analyzed in the following three cases.

- **Case 1:** if $\frac{50mc-m^2k-50\beta c}{mk} < 0$, attacker P_{att} would never gain a larger expected revenue from launching coin hipping attack in pool A.

- **Case 2:** if $0 \leq \frac{50mc-m^2k-50\beta c}{mk} \leq 1$, attacker P_{att} with $\alpha \in [0, \frac{50mc-m^2k-50\beta c}{mk}]$ can get a

larger expected revenue through coin hopping attack in pool A.

- **Case 3:** if $\frac{50mc-m^2k-50\beta c}{mk} > 1$, attacker P_{att} can always increase his expected revenue by conducting coin hopping attack towards pool A.

By adopting the real blockchain statistics, we obtain $\beta \leq 0.5m$ due to the reason of avoiding 51% attack [62]. Then, it can be figured out when $\beta \leq 0.4m$, the value $\frac{50mc-m^2k-50\beta c}{mk} \geq 1$, which falls into Case 3; when $0.4m \leq \beta \leq 0.5m$, attack P_{att} with hash power $\alpha \leq \frac{50mc-m^2k-50\beta c}{mk}$ could get more benefit . *This indicates that in reality, whether an attacker can earn more revenue through coin hopping attack from a Bitcoin network to an Ethereum network is determined by the network condition in blockchain-based IoT.*

Similarly, P_{att} can also perform coin hopping by using the mining power of pool B to mine Bitcoin blocks and award the miners of B with Ethereum standard. Then, P_{att} can mine a new block in the Bitcoin network with a probability of $\frac{\alpha+\beta}{1+\beta}$ and the corresponding expected revenue, denoted by $U_{P_{att}}^{B\rightarrow BTC}$, is calculated as follows.

$$U_{P_{att}}^{B\rightarrow BTC} = \frac{\alpha+\beta}{1+\beta} \cdot B_{BTC} \cdot S_{BTC} \cdot f. \tag{3.20}$$

Under this attack, whether P_{att} can enhance his expected revenue is determined by the value of $\frac{mk-mk\alpha-50c}{50c}$, for which there are three cases for discussion.

- **Case 1:** if $\frac{mk-mk\alpha-50c}{50c} < 0$, attacker P_{att} never gains more profits from coin hipping attack towards pool B.

- **Case 2:** if $0 \leq \frac{mk-mk\alpha-50c}{50c} \leq 1$, attacker P_{att} with $\beta \in [0, \frac{mk-mk\alpha-50c}{50c}]$ can get an increase in his expected revenue through attacking pool B.

- **Case 3:** if $\frac{mk-mk\alpha-50c}{50c} < 1$, attacker P_{att} is always able to earn a bigger expected revenue by performing coin attack in pool B.

If substituting real Bitcoin & Ethereum information with $\alpha \leq 0.5$, we have $\frac{mk-mk\alpha-50c}{50c} \approx -1 < 0$, which located in case 1. Thus, attacker P_{att}'s expected revenue would be reduced if he launches coin hopping attack toward pool B in the Ethereum network. However, the network and economic condition may vary time to time, and it is still feasible for an attacker to perform coin attack towards the Ethereum network. *Therefore, in the blockchain-based IoT, an Ethereum network is still vulnerable to coin hopping attack.*

3.4.3 Mixed Pool

Note that in real-world, most open pools nowadays can mine more than one cryptocurrency at the same time. In this subsection, a more practical scenario is taken into account, in which P_{att} is the pool manager of pool A that simultaneously works for both the Bitcoin and the Ethereum networks. Suppose that the total computational power of pool A is α and $\gamma \in [0, 1]$ is the percentage of computational power of pool A in the Bitcoin network. From Fig. 3.6, one can see that P_{att} can privately utilizes his computational power in the Bitcoin network to verify the transactions from the Ethereum network.

The expected revenue of honest mining consists of two parts (i.e., U_{BTC} and U_{ETH}) and

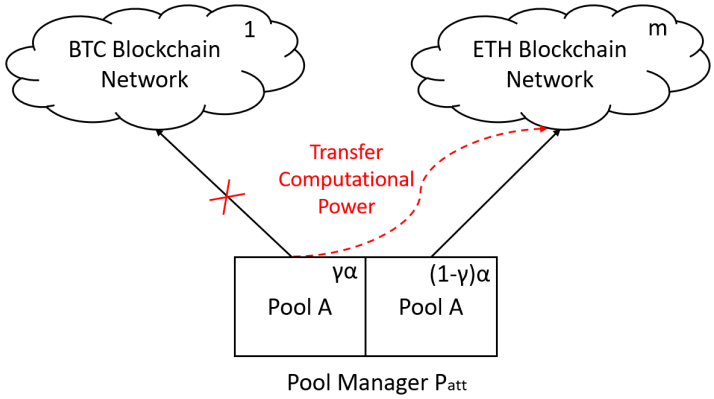


Figure 3.6: Coin hopping in a mixed pool

is computed in Eq. (3.21).

$$
\begin{aligned}
U_{P_{att}} =& U_{BTC} + U_{ETH} \\
=& \frac{\gamma\alpha}{1} \cdot B_{BTC} \cdot S_{BTC} \cdot f \\
&+ \frac{(1-\gamma)\alpha}{m} \cdot (p \cdot B_{ETH} \cdot (1 + \frac{1-p}{32} + \frac{(1-p)^2}{16}) \\
&+ (1-p) \cdot B_{UETH}) \cdot S_{ETH} \cdot f \\
=& \frac{\gamma\alpha k}{50} \cdot B_{ETH} \cdot S_{ETH} \cdot f \\
&+ \frac{(1-\gamma)\alpha c}{m} \cdot B_{ETH} \cdot S_{ETH} \cdot f \\
=& (\frac{\gamma\alpha k}{50} + \frac{(1-\gamma)\alpha c}{m}) \cdot B_{ETH} \cdot S_{ETH} \cdot f.
\end{aligned}
\tag{3.21}
$$

When P_{att} performs coin hopping attack, all the computational power α works for the Ethereum network, and the hash rate of the Ethereum network is increased by $\gamma\alpha$. Formally, $U_{P_{att}}^{BTC \to ETH}$ is used to represent P_{att}'s expected revenue yielded by coin hopping attack and

is calculated as,

$$\begin{aligned} U_{P_{att}}^{BTC\to ETH} =& \frac{\alpha}{m+\gamma\alpha} \cdot [p \cdot B_{ETH} \cdot (1 + \frac{1-p}{32} + \frac{(1-p)^2}{16}) \\ & + (1-p) \cdot B_{UETH}] \cdot S_{ETH} \cdot f \\ =& \frac{\alpha}{m+\gamma\alpha} \cdot B_{ETH} \cdot S_{ETH} \cdot f. \end{aligned} \tag{3.22}$$

Thus, attacker P_{att} is likely to launch attack if the revenue difference between honest mining and attacking is non-negative. The condition for successful coin hopping attack towards the Bitcoin network is formally expressed in Eq. (3.23)

$$\begin{aligned} U_{P_{att}}^{BTC\to ETH} - U_{P_{att}} =& (\frac{\alpha}{m+\gamma\alpha} - \frac{\gamma\alpha k}{50} - \frac{(1-\gamma)\alpha c}{m}) \\ & \cdot B_{ETH} \cdot S_{ETH} \cdot f \\ \geq& 0, \end{aligned} \tag{3.23}$$

which can be equivalently simplified to be Eq. (3.24).

$$\begin{aligned} & \frac{\alpha}{m+\gamma\alpha} - \frac{\gamma\alpha k}{50} - \frac{(1-\gamma)\alpha c}{m} \geq 0 \\ \Rightarrow & 50mc\alpha - (m+\gamma\alpha)\gamma\alpha km - 50(m+\gamma\alpha)(1-\gamma)\alpha c \geq 0. \end{aligned} \tag{3.24}$$

Since α, γ are in the range $[0, 1]$, we have

$$(50\alpha c - mk\alpha)\gamma + (50mc - 50\alpha c - m^2 k) \geq 0, \tag{3.25}$$

which can be satisfied if the following equation holds.

$$\gamma \geq \frac{m^2 k + 50\alpha c - 50mc}{50\alpha c - mk\alpha}. \tag{3.26}$$

Thus, when Eq. (3.26) is met, attacker P_{att} can conduct coin hopping attack towards the Bitcoin network to earn more revenue. The analysis on the value of $\frac{m^2k+50\alpha c-50mc}{50\alpha c-mk\alpha}$ is addressed below.

- **Case 1:** if $\frac{m^2k+50\alpha c-50mc}{50\alpha c-mk\alpha} < 0$, attacker P_{att} can always succeed in increasing his expected revenue via coin hopping attack.
- **Case 2:** if $0 \leq \frac{m^2k+50\alpha c-50mc}{50\alpha c-mk\alpha} \leq 1$, attacker P_{att} with $\gamma \in [\frac{m^2k+50\alpha c-50mc}{50\alpha c-mk\alpha}, 1]$ can get an increased expected revenue through coin hopping attack.
- **Case 3:** if $\frac{m^2k+50\alpha c-50mc}{50\alpha c-mk\alpha} > 1$, attacker P_{att} always fails to receive a higher expected revenue by performing coin hopping attack towards the Bitcoin network.

With the real Bitcoin & Ethereum information, we have $\gamma\alpha < 0.5$, $(1-\gamma)\alpha < 0.5m$, and the value of $\frac{m^2k+50\alpha c-50mc}{50\alpha c-mk\alpha}$ is slightly small than 1, but larger than 0, which is always in Case 2. Thus, P_{att} can get more benefit through performing coin hopping attack when $\gamma \in [\frac{m^2k+50\alpha c-50mc}{50\alpha c-mk\alpha}, 1]$. In other words, *whether it is beneficial for an attacker to transfer mining power from a real Bitcoin network to a real Ethereum network is affected by the network condition in the blockchain-based IoT.*

Besides, P_{att} can perform coin hopping by transferring mining power from the Ethereum network to the Bitcoin network, in which the probability of mining a new Bitcoin block is $\frac{\alpha}{1+(1-\gamma)\alpha}$ and the corresponding expected revenue is denoted as $U_{P_{att}}^{ETH\rightarrow BTC}$ that is estimated in Eq. (3.27).

$$U_{P_{att}}^{ETH\rightarrow BTC} = \frac{\alpha}{1+(1-\gamma)\alpha} \cdot B_{BTC} \cdot S_{BTC} \cdot f. \tag{3.27}$$

To obtain the condition where $U_{P_{att}}^{ETH \to BTC} \geq U_{P_{att}}$, we need to solve Eq. (3.28).

$$\begin{aligned} &(100\gamma c + m\gamma^2 k - 50\gamma^2 c - m\gamma k - 50c)\alpha \\ &+ mk - m\gamma k + 50c - 50\gamma c \geq 0. \end{aligned} \tag{3.28}$$

By equivalently simplifying the above inequality, we have

$$\begin{aligned} &(100\gamma c + m\gamma^2 k - 50\gamma^2 c - m\gamma k - 50c)\alpha \\ &+ mk - m\gamma k + 50c - 50\gamma c \geq 0 \\ \Rightarrow &\alpha \geq \frac{m\gamma k + 50\gamma c - mk - 50c}{100\gamma c + m\gamma^2 k - 50\gamma^2 c - m\gamma k - 50c}. \end{aligned} \tag{3.29}$$

Thus, when Eq. (3.29) holds, attacker P_{att} can earn more revenue by conducting coin hopping attack towards the Ethereum network. The specific cases of $\frac{m\gamma k + 50\gamma c - mk - 50c}{100\gamma c + m\gamma^2 k - 50\gamma^2 c - m\gamma k - 50c}$ is detailed in the following.

- **Case 1:** if $\frac{m\gamma k + 50\gamma c - mk - 50c}{100\gamma c + m\gamma^2 k - 50\gamma^2 c - m\gamma k - 50c} < 0$, attacker P_{att} can always gain more benefit from coin hopping attack.

- **Case 2:** if $0 \leq \frac{m\gamma k + 50\gamma c - mk - 50c}{100\gamma c + m\gamma^2 k - 50\gamma^2 c - m\gamma k - 50c} \leq 1$, attacker P_{att} with $\alpha \in [\frac{m\gamma k + 50\gamma c - mk - 50c}{100\gamma c + m\gamma^2 k - 50\gamma^2 c - m\gamma k - 50c}, 1]$ can enhance his expected revenue through this attack.

- **Case 3:** if $\frac{m\gamma k + 50\gamma c - mk - 50c}{100\gamma c + m\gamma^2 k - 50\gamma^2 c - m\gamma k - 50c} > 1$, attacker P_{att} cannot make a revenue improvement via coin hopping attack.

Therefore, one can see that *although coin hopping attack towards an Ethereum network cannot benefit an attacker in the real Bitcoin & Ethereum network environments currently,*

it still poses a severely potential threat to the blockchain-based IoT as network environments may vary with time.

From our theoretical analysis, we know that (i) the conditions and reasons for successfully launching coin hopping attack in the above three scenarios are essentially the same; and (ii) a pool manager can judge whether it is beneficial to perform coin hopping attack by analyzing the network environments.

3.4.4 Attack Consequence Analysis

In this subsection, the multi-dimensional serious consequences caused by coin hopping attack are studied from the viewpoint of economic benefit. Particularly, the most practical scenario (see Section 3.4.3), in which coin hopping attack is towards the Bitcoin network in a mixed pool, is used as an example for analysis. We start our analysis from the investigation of the miners' expected rewards

3.4.4.1 Inner-Pool Bitcoin Miner

When the pool manager honestly mines, the inner-pool Bitcoin miners have a computational power of $\gamma\alpha$. Thus, they could together share the revenue U_{InBTC}:

$$U_{InBTC} = \frac{\gamma\alpha}{1} \cdot B_{BTC} \cdot S_{BTC} \cdot (1 - f). \tag{3.30}$$

When coin hopping attack happens, all the miners' power is transferred from the Bitcoin network to the Ethereum network. Thus, they should be awarded based on the Ethereum standard and together get a total expected revenue, denoted by $U'^{deserve}_{InBTC}$, as computed in

Eq. (3.31).

$$U_{InBTC}^{'deserve} = \frac{\gamma\alpha c}{m + \gamma\alpha} \cdot B_{ETH} \cdot S_{ETH} \cdot (1 - f). \tag{3.31}$$

However, to obtain an increased revenue, the malicious pool manager in fact uses the Bitcoin standard to award them, and thereby they actually receive a revenue $U_{InBTC}^{'}$ that is calculated as,

$$U_{InBTC}^{'} = \frac{\gamma\alpha}{1 - \gamma\alpha} \cdot B_{BTC} \cdot S_{BTC} \cdot (1 - f). \tag{3.32}$$

Based on our prior analysis, it is obvious to conclude that $U_{InBTC}^{'deserve} > U_{InBTC}^{'} > U_{InBTC}$. Thus, the inner-pool Bitcoin miners could get a total reward slightly higher than that of honest mining, but they actually lose revenue they deserved.

Remark: In the pool mining process, an individual miner's received reward is decided by the amount of shares he submits as well as the total submitted shares in the pool, which is dynamic and unpredictable. That is, an individual miner's received reward is not deterministic. Thus, we analyze the total reward of all the inner-pool Bitcoin miners rather than an individual Bitcoin miner's reward. With the same reason, in the following analysis, we compute and compare the total reward.

3.4.4.2 Inner-Pool Ethereum Miner

The inner-pool Ethereum miners own a computational power of $(1-\gamma)\alpha$ and share the total revenue U_{InETH}.

$$U_{InETH} = \frac{(1-\gamma)\alpha c}{m} \cdot B_{ETH} \cdot S_{ETH} \cdot (1-f). \tag{3.33}$$

When the pool manager performs coin hopping attack, the total mining power used for the Ethereum network is increased by $\gamma\alpha$. Therefore, the probability of finding a new Ethereum block by them drops and their total expected revenue becomes:

$$U'_{InETH} = \frac{(1-\gamma)\alpha c}{m+\gamma\alpha} \cdot B_{ETH} \cdot S_{ETH} \cdot (1-f). \tag{3.34}$$

Since $U'_{InETH} < U_{InETH}$, they suffer from revenue loss caused by coin hopping attack.

3.4.4.3 Outer-Pool Bitcoin Miner

The outer-pool Bitcoin miners have the computational power of $1-\gamma\alpha$, and their mining power remains the same when the attacker performs coin hopping attack. First, we calculate their total reward U_{OutBTC} of honest mining:

$$U_{OutBTC} = \frac{1-\gamma\alpha}{1} \cdot B_{BTC} \cdot S_{BTC} \cdot (1-f). \tag{3.35}$$

With the existence of coin hopping attack, there is no mining power working for the Bitcoin network in the attacker's pool. Thus, the expected revenue for the outer-pool Bitcoin

miners becomes:

$$U'_{OutBTC} = 1 \cdot B_{BTC} \cdot S_{BTC} \cdot (1 - f). \tag{3.36}$$

Therefore, the outer-pool Bitcoin miners could expect a higher reward because $U'_{OutBTC} > U_{OutBTC}$.

3.4.4.4 Outer-Pool Ethereum Miner

Without coin hopping attack, the outer-pool Ethereum miners own a mining power of $m - (1 - \gamma)\alpha$ and share a total reward U_{OutETH}:

$$U_{OutETH} = \frac{m - (1 - \gamma)\alpha}{m} \cdot c \cdot B_{ETH} \cdot S_{ETH} \cdot (1 - f). \tag{3.37}$$

If coin hopping attack occurs, the total mining power of the Ethereum network is increased by $\gamma\alpha$. Therefore, the probability of finding a new Ethereum block by them is decreased and their corresponding expected revenue U'_{OutETH} changes to

$$U'_{OutETH} = \frac{m - (1 - \gamma)\alpha}{m + \gamma\alpha} \cdot c \cdot B_{ETH} \cdot S_{ETH} \cdot (1 - f). \tag{3.38}$$

Since $U'_{OutETH} < U_{OutETH}$, they also lose revenue because of coin hopping attack.

The above analysis on the miners' expected rewards also provides us with more insight into the impact of coin hopping attack at the network level. Note that the total mining power in both the Bitcoin and the Ethereum networks do not change when the pool manager launches coin hopping attack. Thus, as outer-pool Bitcoin miners could earn more when coin

hopping attack is performed, more miners may be attracted to work for Bitcoin network. Due to such change in the network environments, the pool manager has more chances to hop between the Bitcoin and the Ethereum networks for revenue enhancement, e.g., from the Bitcoin network to the Ethereum network or from the Ethereum network to the Bitcoin network. Consequently, *the stability of the two involved networks cannot be continuously maintained in a long-term period*. What is worse, due to coin hopping attack, the pool manager can obtain an increase in his long-term expected revenue, but all the miners' long-term expected revenue would be lower than the amount they deserve; that is, *a part of the miners' deserved rewards is plundered by the malicious pool manager*. As a result, *the blockchain-based IoT with vulnerability to coin hopping attack gradually lose its attractiveness and competitiveness and will eventually stagnate*.

3.5 Merged Mining Analysis

In the previous section, the Bitcoin and the Ethereum blockchains are used as examples to analyze the feasibility of coin hopping attack in the blockchain-based IoT. Here, we investigate a more sly hopping behavior, in which coin hopping attack happens during merged mining process.

Merged mining is the process of simultaneously mining two or more different cryptocurrencies based on the same consensus algorithm [64]. Two of the most classic examples of merged mining are Litecoin and Dogecoin, as well as Namecoin and Bitcoin [65]. Technically speaking, merged mining reuses the shares from a parent cryptocurrency as valid POW for

one or more auxiliary chains, which indicates the difficulty level of the child blockchains is lower than that of the parent chain [64]. *To implement merged mining, a pool manager only needs to make minor modification to the parent blockchain script, which makes coin hopping attack easier to be performed but more difficult to be detected.* In this situation, the pool manager can get the equivalent profits from the auxiliary chains as additional bonus instead of just taking the pool fee.

In the blockchain-based IoT, there exist some cases where a mining pool simultaneously serves for two or more networks that can run merged mining. For example, we can imagine two blockchain-based IoT networks, which are smart vehicle and intelligent transportation system (ITS). As we known, vehicle is a part of the whole transportation system, thereby the blockchain for smart vehicle can be treated as an auxiliary chain for ITS. Thus, the difficulty of puzzle in ITS chain is harder than smart vehicle, which means when miners verifying the transactions in smart vehicle blockchain, they in fact doing partial work for the ITS blockchain.

3.5.1 Attack Behavior

Without loss of generality, we consider the scenario in Section 3.4.3 and use Bitcoin and Namecoin as examples to analyze the feasibility of coin hopping attack in merging mining. Suppose that P_{att} holds computational power of α, where γ is the percentage of computational power of pool A that mines in the Bitcoin network. Hence, $(1-\gamma)\alpha$ is the computational power for the Namecoin network. Let B_{NMC} be the block reward for Namecoin, S_{NMC} be the frequency of mining a new Namecoin block, and n is the hash rate of the entire Namecoin

network. Thus, the pool manager's expected revenue contains two parts, including U_{BTC} and U_{NMC}, and can be computed by Eq. (3.39).

$$\begin{aligned} U_{P_{att}} =& U_{BTC} + U_{NMC} \\ =& \frac{\gamma\alpha}{1} \cdot B_{BTC} \cdot S_{BTC} \cdot f \\ &+ \frac{(1-\gamma)\alpha}{n} \cdot B_{NMC} \cdot S_{NMC} \cdot f. \end{aligned} \tag{3.39}$$

When the pool manager implements coin hopping attack through conducting merged mining on the Bitcoin chain, the total computational power of the Namecoin network is enhanced to $n + \gamma\alpha$. Thus, the pool manager's expected revenue becomes

$$\begin{aligned} U_{P_{att}}^{BTC \overset{merge}{\longrightarrow} NMC} =& \frac{\gamma\alpha}{1} \cdot B_{BTC} \cdot S_{BTC} \cdot f \\ &+ \frac{\gamma\alpha}{n+\gamma\alpha} \cdot B_{NMC} \cdot S_{NMC} \\ &+ \frac{(1-\gamma)\alpha}{n+\gamma\alpha} \cdot B_{NMC} \cdot S_{NMC} \cdot f. \end{aligned} \tag{3.40}$$

Correspondingly, the increase of the expected revenue is

$$\begin{aligned} & U_{P_{att}}^{BTC \overset{merge}{\longrightarrow} NMC} - U_{P_{att}} \\ =& \frac{\gamma\alpha}{n+\gamma\alpha} \cdot B_{NMC} \cdot S_{NMC} + \\ & [\frac{(1-\gamma)\alpha}{n+\gamma\alpha} - \frac{(1-\gamma)\alpha}{n}] \cdot B_{NMC} \cdot S_{NMC} \cdot f. \end{aligned} \tag{3.41}$$

Obviously, from Eq. (3.41), it can be seen that $U_{P_{att}}^{BTC \overset{merge}{\longrightarrow} NMC} - U_{P_{att}} > 0$ always holds, i.e., *coin hopping attack via merged mining can always help the pool manager earn more benefit in the blockchain-based IoT*. This is consistent with the fact that merged mining is

always a welcome choice for pool managers because of the additional block reward.

3.5.2 Attack Consequence Analysis

In previous subsection, we show that the pool manager can earn extra benefit by performing coin hopping attack through merged mining. In this part, we discuss the attack consequence from aspects of the miners and the networks.

3.5.2.1 Inner-Pool Bitcoin Miner

Without any coin hopping attack, the inner-pool Bitcoin miners' computational power is $\gamma\alpha$ and their expected reward is U_{InBTC} that is calculated as

$$U_{InBTC} = \frac{\gamma\alpha}{1} \cdot B_{BTC} \cdot S_{BTC} \cdot (1-f). \tag{3.42}$$

When the attacker performs coin hopping attack during merged mining process, the inner-pool Bitcoin miners should earn a bonus reward from the Namecoin network. Thus, their expected reward $U_{InBTC}^{'deserve}$ is:

$$\begin{aligned} U_{InBTC}^{'deserve} =& \frac{\gamma\alpha}{1} \cdot B_{BTC} \cdot S_{BTC} \cdot (1-f) \\ &+ \frac{\gamma\alpha}{n+\gamma\alpha} \cdot B_{NMC} \cdot S_{NMC} \cdot (1-f). \end{aligned} \tag{3.43}$$

However, since they do not even know merged mining is undergoing in the attacker's pool, the inner-pool miners only expect the reward $U_{InBTC}^{'}$ from the Bitcoin network; especially, $U_{InBTC}^{'} = U_{InBTC}$. That is, the inner-pool miners' received reward is less than the amount they deserve, and their benefit is plundered by the pool manager.

3.5.2.2 Inner-Pool Namecoin Miner

Under the situation where the pool manager mine honestly, the inner-pool Namecoin miners' expected reward U_{InNMC} is:

$$U_{InNMC} = \frac{(1-\gamma)\alpha}{n} \cdot B_{NMC} \cdot S_{NMC} \cdot (1-f). \tag{3.44}$$

If coin hopping happens, the total mining power of the Namecoin network increases because of the merged mining. Thus, the inner-pool Namecoin miners' expected reward changes to U'_{InNMC}, which is obtained in Eq. (3.45).

$$U'_{InNMC} = \frac{(1-\gamma)\alpha}{n+\gamma\alpha} \cdot B_{NMC} \cdot S_{NMC} \cdot (1-f). \tag{3.45}$$

As $U'_{InNMC} < U_{InNMC}$, the inner-pool Namecoin miners lose their benefits in the presence of coin hopping attack.

3.5.2.3 Outer-Pool Bitcoin Miner

In fact, there is no change in the Bitcoin network when the pool manager implement coin hopping attack via merged mining. Thus, the outer-pool Bitcoin miners' expected reward remains unchanged, i.e., $U'_{OutBTC} = U_{OutBTC}$ where U'_{OutBTC} represents the expected reward when coin hopping attack happens.

3.5.2.4 Outer-Pool Namecoin Miner

If there is no coin hopping attack, the outer-pool Namecoin miners' mining power is $n-(1-\gamma)\alpha$ and thus share a reward U_{OutNMC}.

$$U_{OutNMC} = \frac{n-(1-\gamma)\alpha}{n} \cdot B_{NMC} \cdot S_{NMC} \cdot (1-f). \tag{3.46}$$

If there is coin hopping attack, the total mining power of the Namecoin network increases and the outer-pool Namecoin miners could receive a reward $U^{'}_{OutNMC}$

$$U^{'}_{OutNMC} = \frac{n-(1-\gamma)\alpha}{n+\gamma\alpha} \cdot B_{NMC} \cdot S_{NMC} \cdot (1-f). \tag{3.47}$$

Because $U^{'}_{OutNMC} < U_{OutNMC}$, the outer-pool Namecoin miners receive less when coin hopping attack happens.

To sum up, when coin hopping attack is launched through merged mining, the miners' received rewards are either less than the amount they deserve during merged mining process or reduced due to merged mining. That is, all the miners definitely suffer from benefit loss in the presence of coin hopping attack in merged mining scenario. Moreover, compared with the attack scenarios in Section 3.4, launching coin hopping attack during merge mining process has the following features.

- It is easier for a pool manager to perform, for which the reasons lie in two aspects: (i) only some minor modifications are needed on the parent chain script; and (ii) no extra condition is needed for its success.

- It is harder to be sensed by the miners, because there is no computational power transformation.

As a result, *coin hopping attack in merge mining has more serious impact to hinder the development of the blockchain-based IoT.*

3.6 Chapter Summary

As the development of blockchain-based IoT has been greatly promoted in recent years, the security issue of blockchain-based IoT is being paid more attention. In this paper, we focus on the analysis of coin hopping attack, which is easy to launch but hard to be detected. The novelty of our work lies in the following aspects: (i) the feasibility of coin hopping attack is proved; (ii) the conditions of implementing coin hopping attack are identified; (iii) the severe impacts on the blockchain-based IoT are analyzed.

CHAPTER 4

ZKCROWD: A HYBRID BLOCKCHAIN-BASED CROWDSOURCING PLATFORM

4.1 Introduction

With the rapid development and wide application of crowdsourcing, the shortcomings of the traditional crowdsourcing systems are gradually exposed. (1) The traditional centralized crowdsourcing systems are vulnerable to a single point of failure. For example, users could not access Wikipedia data from the server due to 8 service outrage in 2018 [19]. (2) In such a centralized system where a server controls all the transactions, the issue of controller's silently misbehave is likely to occur without effective detection. (3) When a conflict of opinions exists between the task requesters and the workers, the issues of Free-riding (workers receive rewards without making real efforts) and False-reporting (requesters try to repudiate the payment) could happen in the system [66]. (4) During the procedure of task assignment, the sensitive information (*e.g.*, location and preference) of crowdsourcing participants may be revealed by the public. For example, in Nov. 2017, 57 million Uber driver's information has been hacked [20]. To solve these issues, a number of countermeasures have been proposed.

To name some: cryptographic techniques were used to preserve user's private information [67, 68, 69]; and reputation systems were adopted to tackle the issues of free-riding and false-reporting [70, 71]. However, most of the existing works fail to simultaneously overcome the aforementioned issues for crowdsourcing.

The recent big progress of blockchain technology enables the seamless integration of

smart contracts and novel cryptographic tools into blockchain networks, which provides an innovative way to improve the performance of crowdsourcing. In [2, 49, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85], a variety of solutions have been presented for privacy issues. In order to offer a security guarantee for the users' participation, identity authentication systems were designed in [86, 87]. In [73], the combination of public and private chains is used to protect user's private information, in which however, the authority of the agent entity needs to be further discussed. Nevertheless, all of these works ignore the scalability and verification efficiency of the blockchain network architecture which are the performance bottleneck of the crowdsourcing systems.

Motivated by the above observations, in this Chapter, we propose to design a blockchain-enabled crowdsourcing platform to provide users with diverse privacy protection while improving the efficiency of crowdsourcing. To achieve our goal, we have to tackle the following challenging problems. (1) *Which consensus protocol should be adopted to enhance the transaction verification speed as well as to provide security guarantee?* For instance, under the traditional Proof-of-Work (POW) consensus protocol, the blockchain networks suffer from issues of low verification rate, scalability bottleneck and huge energy consumption [62]. (2) *How to balance the tradeoff between transparency and privacy in crowdsourcing?* Note that transparency is one of the most important advantages of the blockchain to achieve accountability. But, on the other hand, transparency could result in privacy leakage. (3) *How to effectively verified the protected transaction information?* A crowdsourcing platform is expected to ensure that the correctness of private information is well verified while the exact

transaction information is unknown by the public. (4) *How to integrate the public and private transaction information?* As user's privacy preference varies in person, it is desired to offer flexible privacy protection. In this situation, some transactions may be public, while others are private. The integration of transaction information needs to be well arranged to meet the users' privacy requirements.

Our research endeavor for dealing with the aforementioned challenges is to design a hybrid blockchain-enabled crowdsourcing platform, called **zkCrowd**. The zkCrowd is composed of a public chain and multiple private subchains, in which public and private tasks are managed on the public chain and subchains, respectively. By exploiting such a hybrid blockchain architecture, the answers of public and private tasks in crowdsourcing can be separately verified, so as to provide privacy protection for the answers of private tasks. For the purpose of enhancing verification speed, Delegated Proof of Stake (DPOS) and Practical Byzantine Fault Tolerance (PBFT) consensuses are implemented on the public chain and subchains, respectively. Compared with existing hybrid blockchains, such as Smilo [88], AERGO [89], and Xinfin [90], our zkCrowd has the following major advantages: (1) In zkCrowd, the cross-chain communication only involves the transmission of answer confirmation from the subchains to the public chain without the issue of cross-chain communication atomicity. Therefore, there is no need to build a bridge module for transaction synchronization, improving verification efficiency for crowdsourcing tasks as well as reducing energy consumption. (2) In crowdsourcing systems, due to the dynamic of tasks and mobility of users, the number of tasks is fluctuating. Inspired by this characteristic, on zkCrowd, a subchain is dynamically created

and released according to the task requirement. After a private task has been finished, the subchain is released, and the corresponding subchain validators can turn back to the public chain. That is, the subchain validators do not need to be always in the monitor state, reducing resource consumption. (3) Our zkCrowd has a unified election mechanism for the public chain and subchains. The voting mechanism of the subchain validators in PBFT is combined with the voting witnesses in DPOS. Therefore, the dynamic subchain establishment does not yield any extra overhead to the blockchain system when selecting validators. The effectiveness of our zkCrowd is evaluated through comprehensive performance analysis and experiment comparison. To sum up, the major contributions of this Chapter are addressed in the following.

- An innovative hybrid blockchain platform, zkCrowd, is elaborately designed for distributed crowdsourcing, in which transaction privacy and transparency can be effectively balanced.

- With utilizing DPOS and PBFT consensus protocols, the transaction verification efficiency can be significantly increased, reducing transaction latency and energy consumption in the crowdsourcing system.

- Diverse privacy protection is achieved by accomplishing zero-knowledge proof and flexible task-based permission control in the smart contracts on the proposed hybrid blockchain architecture.

- Both the theoretical analysis and experiments can validate the advantages of our

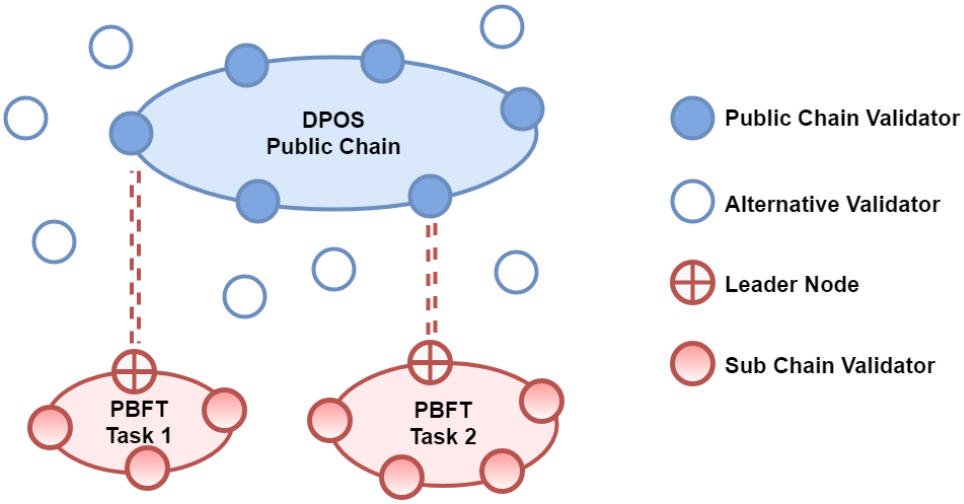


Figure 4.1: The blockchain architecture of zkCrowd

zkCrowd compared with the state-of-the-art.

The remaining organization of this Chapter is as follows. Our hybrid blockchain platform, zkCrowd, and the implementation process are detailed in Section 4.2 and Section 4.3, respectively. The theoretical analysis is shown in Section 4.4, and the experiments are analyzed in Section 4.3.7. Finally, this Chapter is concluded in Section 4.6.

4.2 Framework of zkCrowd

Our zkCrowd platform is established on a hybrid blockchain network being composed of a public chain and multiple subchains. As shown in Fig. 4.1, on zkCrowd, the public chain and the subchains are used to publish and record the information of public and private crowdsourcing tasks, respectively.

The entities on zkCrowd include:

(1) Requester. Requesters post tasks, collect answers from workers, and pay to workers

and validators for their efforts. Before publishing tasks, they need to deposit an amount of money into the blockchain through the smart contracts, in which the deposit is used to reward the workers who make effort in completing tasks as well as the validators who help verify transactions. Every requester has the right to vote the validators on the public chain and the subchains. A requester may simultaneously play the role as a worker for a different task but is not allowed to work as a validator.

(2) Workers. Workers carry out requesters' tasks and receive payments from the requesters via signing the smart contracts. If a worker participates in a private task, his answers should be submitted to subchain validators. Each worker has the right to vote for public-chain validators and subchain validators. Any worker can not register as a validator, but can potentially become a requester.

(3) Validators. There exist three kinds of validators. (i) *Public chain validators* verify transactions of the public chain, take turns to perform the accounting right, share a public chain ledger, and need to be elected in each time cycle. (ii) *Alternative validators* are the nodes that are not elected as the public chain validators. They should also share the public chain ledger but do not have the right to generate blocks. (iii) *Subchain validators* are elected among the existing alternative validators to verify transactions on the subchains and record both the public chain and subchain ledgers. Any validator cannot work as a public chain validator and a subchain validator at the same time.

All the entities must register as the legitimate participants through a Trustworthy Certificate Authority (TCA) when joining zkCrowd. Only the entities that have obtained the

legal certificates can carry out the activities on zkCrowd.

When the public chain has been set up, the requesters can post public and/or private tasks on zkCrowd, and the workers select their preferred tasks and need the permission from the corresponding requesters to process the tasks. The type of tasks is checked via the smart contracts. For the public tasks, the answers are submitted to the public chain for verification, and the transaction information is kept in the public chain ledger; while for the private tasks, a task-based subchain is built, the answers are uploaded to the corresponding subchain for verification, and the transaction information is written into the subchain ledger. On each subchain, after the subchain ledger gets updated, the leader constructs a new Merkle tree of which the leaves are the block Merkle roots extracted from all block headers and post a transaction confirmation that is computed using ZK-SNARK on the public chain.

This confirmation can be used to trace the leader's behavior because on each subchain, the subchain ledger is shared by all subchain validators. On the other hand, the information of private tasks is allowed to be accessed via the permission stated in the smart contracts and thus can be preserved on the subchains without being revealed by the public. Finally, the payments are assigned to the workers and the validators according to the distribution method (that can be determined by the requesters based on their task requests), and the remaining deposits will be returned to the requesters.

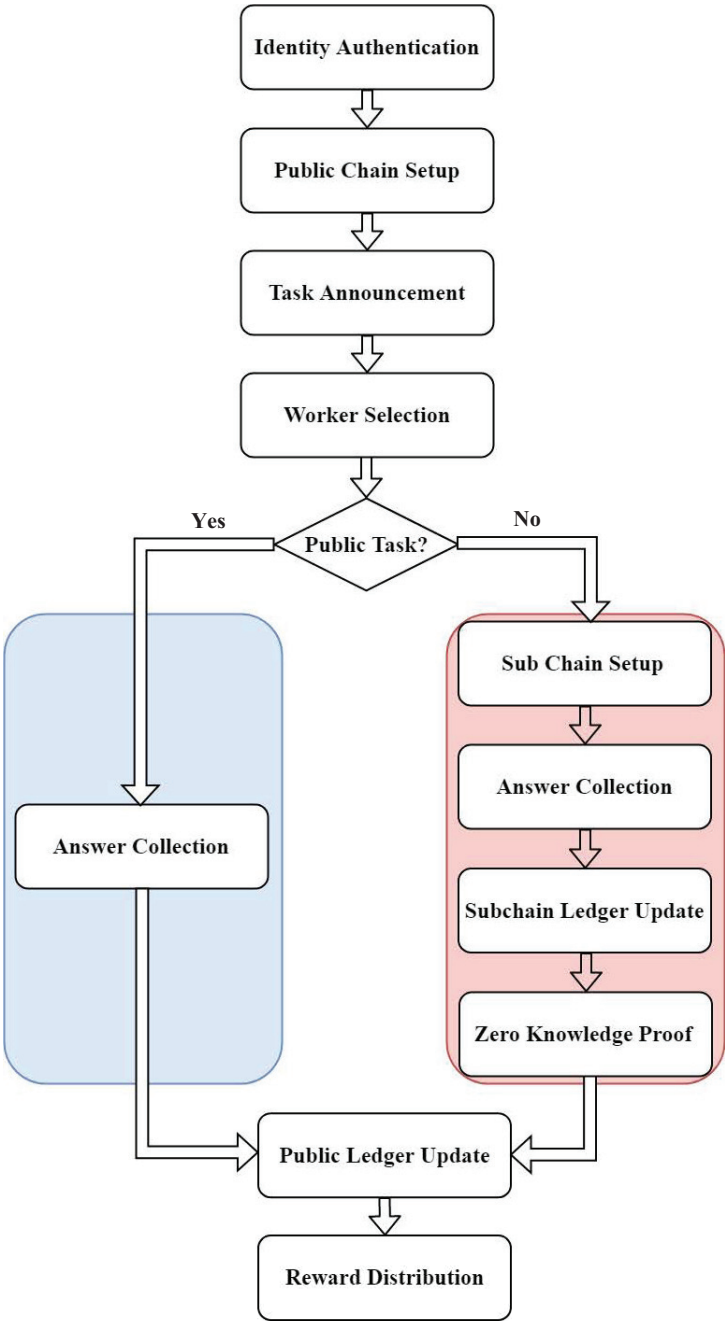


Figure 4.2: Work flow of zkCrowd implementation process

Table 4.1: Notation & Definition

Notation	Definition
pk_*	Public key of entity *
sk_*	Private key of entity *
ut	User type in {r, w, v}
ID_*	Real identity information of entity *
ID_*^c	Certified identity of entity *
ID_t	Task ID
T	Task type in {public, private}
t_1	Deadline for workers to reply to tasks
t_2	Deadline for workers to submit answers
t_r	Worker's response time
t_a	Answer submission time
t_c	Latest answer submission time
R	Total rewards of a task implementation
Num	Maximum number of answers for a task
p	Task implementation preference
cap_w	Worker's capacity
$permit$	Permission to process task
Ans	Task answer
$A_{ID_w^c}$	Answer content submitted by worker
π_a	Attestation of ZK-proof
PP	Public parameter of ZK-proof
$mt.root$	Merkle tree root

4.3 Implementation of zkCrowd

This section details the major phases in the implementation of zkCrowd. For a better presentation, the work flow of the platform implementation is presented in Fig. 5.1, and the notations are summarized in Table 4.1.

4.3.1 Identity Authentication

In crowdsourcing, the authentication of user identities is an important issue [87], but the blockchain itself cannot verify this. To solve the issue, a Trustworthy Certificate Authority (TCA) is brought to take charge of the identity authentication and certificate issuance for the entities.

Initially, an entity prepares key (pk_{ut}, sk_{ut}) using RSA scheme, in which pk_{ut} and sk_{ut}

are respectively the entity's public and private keys, and $ut \in \{r, w, v\}$ is the entity type (r, w, and v stand for requester, worker, and validator, respectively). The entity registers by sending $E(pk_c, pk_{ut}, ID_{ut}, Sig(sk_{ut}, ID_{ut}))$ to TCA, where pk_c is TCA's public key, ID_{ut} is the entity's true identity, and $Sig(\cdot)$ is the secure digital signature algorithm. Then, TCA checks if ID_{ut} and pk_{ut} are in his historic information or not because everyone on zkCrowd cannot use different public keys to prevent Sybil attack. Once ID_{ut} and pk_{ut} are verified to be valid, a unique certified identity ID^c_{ut} and a certification $Cert_{ut}\{ID^c_{ut}, ut, pk_{ut}\}$ are issued and sent back to the entity in the form of $E(pk_{ut}, Cert_{ut}, Sig(sk_c, Cert_{ut}))$ with sk_c being TCA's private key. After a successful registration, the entity can participate in the activities on zkCrowd.

4.3.2 Public Chain Establishment

On zkCrowd, by running the DPOS consensus, the public chain is established by all validators who are issued certificates by TCA [91]. After that, the public chain validators are elected by the registered requesters and workers periodically using the election mechanism of DPOS. In particular, a validator re-election process should be launched if the malicious behaviors of any public chain validator are found. In every election process, the 21 validators who receive the most votes serve as the public chain validators to perform the accounting right in turns as well as the transaction verification in the current period. The rest of the validators are the alternative validators that share the public chain ledger and are available for the election of subchain validators.

4.3.3 Task Announcement & Worker Selection

Each task is published on zkCrowd with the information $Task\{T, ID_t, t_1, t_2, R, Num, p, pk_r, Cert_r, Sig(sk_c, Cert_r)\}$, where $T \in \{public, private\}$ represents the task type, ID_t is the task ID, t_1 and t_2 are respectively the deadlines for the workers to reply to the task information and submit the task answers, R is the total amount of payment for the task implementation, Num is the maximum number of expected answers for the task, p is the task requirements (such as location, duration, and data quality) needed by the task requester to choose workers, pk_r is the task requester's public key, $Cert_r$ is the task requester's certification. Meanwhile, each task requester must make a deposit to the smart contracts. On zkCrowd, the smart contracts are also executed to check if or not the task is announced by a registered requester and the deposit is enough to pay to the workers and validators. The certificates of all requesters and workers can be checked using TCA's digital signatures (*i.e.*, $Sig(sk_c, Cert_r)$ and $Sig(sk_c, Cert_w)$). Once a task successfully passes such a checking process, the task information is delivered to the online workers of zkCrowd.

A worker should return a reply before the response deadline if he prefers to process a task. The reply message is as follows: $Res\{ID_t, ID_w^c, t_r, E(cap_w), pk_w, Cert_w, Sig(sk_c, Cert_w)\}$, where ID_w^c is the worker's registered ID, t_r is the response timestamp, cap_w indicates the worker's capacity (such as location, available time, quality of offered data, and others), and pk_w and $Cert_w$ are respectively the worker's public key and certificate. Since cap_w may contain the worker's private information, it should not be publicly available when submitted to the blockchain. To prevent privacy leakage, the worker encrypts cap_w with the requester's

public key pk_r, *i.e.*, $E(cap_w) = E(pk_r, cap_w, Sig(sk_w, cap_w))$. When the reply is delivered to the requester, he examines the response time to make sure $t_r \leq t_1$, uses sk_r to recover cap_w, and verifies the worker's certificate as well. By considering the task requirements p and the worker's capacity cap_w, the requester can select his preferred workers, issue a permission $permit(ID_t, ID_w^c, Cert_w)$ to each selected worker, and send out the notification message $E(pk_w, permit, Sig(sk_r, permit))$. Each selected worker should upload the answer to zkCrowd before t_2 in the form of $Ans(T, ID_t, A_{ID_w^c}, ID_w^c, t_a, permit)$ with $A_{ID_w^c}$ recording the answer content and t_a being the submission timestamp.

4.3.4 Subchain Establishment

Whenever a private task needs to be processed on zkCrowd, a subchain is built to collect answers and verify transactions. In other words, the establishment of subchains is dynamically determined by the demand of private tasks.

On each subchain, the PBFT consensus [92, 93] is adopted, and the validators are elected among the current highest voted alternative validators. To defend Byzantine failure attack under the PBFT consensus, the total number of nodes must be larger than 3 times of the number of the Byzantine failure nodes in the blockchain network. Formally, assume the number of subchain validators is n_v and the number of Byzantine nodes is n_b. To achieve the Byzantine Fault Tolerance on zkCrowd, we have $n_v = 3n_b + 1$ where $n_b \geq 1$; that is, there must exist at least 4 subchain validators on each subchain. The determination of n_v should consider the following facts: (1) If the maximum number of answers Num for a private task is small, a few validators (*e.g.*, 4 validators) are enough to deal with the transaction volume of

the subchain; while if Num is large, more validators are necessary to handle the transaction verification. (2) A larger value of n_v indicates a stronger ability to resist Byzantine failure attack. (3) As more validators enter a subchain, the communication overhead may increase dramatically, and the verification fee is also increased, leading to a higher cost to run the subchain. Note that on the public chain, 21 validators are sufficient to verify all transactions for the whole system. Since the value of Num may not be too large, a small number of validators (*e.g.*, 4 or 7) is enough. Besides, in the PBFT consensus, a leader node is elected periodically in the round robin manner to update the subchain ledger and communicate with the public chain.

After the private task is completed, the subchain is released and its subchain validators become the alternative validators on zkCrowd. In other words, any subchain validator can re-participate in the election of the public chain validators only after his subchain is released.

4.3.5 Answer Collection on Public Chain

During the procedure of answer collection, the smart contracts should ensure that in $Ans(T, ID_t, A_{ID_w^c}, ID_w^c, t_a, permit)$: (1) the answer is submitted by a permitted worker by checking $permit$; (2) the answer $A_{ID_w^c}$ has not yet been recorded in the answer pool $Pool_a$; and (3) the answer is submitted on time, *i.e.*, $t_a \leq t_2$. If the above three conditions can be simultaneously satisfied, the answer is valid and is recorded into the answer pool. If the number of valid answers exceeds the value of Num, only the first Num valid answers are recorded into blocks; otherwise, all the valid answers are recorded.

4.3.6 Answer Collection on Subchain

For each private task, the procedure of answer collection is similar to that on the public chain. The subchain validators only copy the transaction records from the public-chain ledger and record private task information in the subchain ledgers. The leader of each subchain can utilize zero-knowledge proof to generate a commitment on all subchain blocks of a task and sends it to the public chain, and on the public chain, the smart contracts run ZK-SNARK verifier to check the task completion and records it into the public chain ledger which will be described in Section 4.3.7. Therefore, all private tasks can not be revealed but can be verified on the public chain. In zkCrowd, the type of task determines whether the answers will be handled by the public chain or subchain. Particularly, for one task, different workers may require different privacy protection level of their answers. Thus for a better access control feature, an access list model can be added into the subchain smart contracts, which will be studied in our future work.

4.3.7 Zero-Knowledge Proof

The transaction information of the subchains is sent to the public chain using ZK-SNARK [94] attestation to verify that all the answers are submitted correctly by the subchain leaders for the workers. Within each block, every transaction can be tracked using Merkle root. When a task is completed, one or multiple blocks may be generated on the subchain. In order to enhance the verification efficiency, the leader extracts the Merkle roots from all the block headers to construct a new Merkle tree and computes the new Merkle root. Such a new root,

denoted by $MT.Root(mt.root_{ID_t})$, means that all the answers are submitted for the private task with identity ID_t. Notice that every $2^n - 1$ blocks can form a full binary Merkle tree, in which n could be any positive integer. To make the frequency of answer submission on the subchains flexible, each leader can perform the above operation whenever receiving $2^n - 1$ blocks before the answer submission deadline t_2, where the value of n is determined by the platform. Furthermore, the overhead of computing a new Merkle root can be adjusted by setting the value of n.

Then, the leader counts the total number of answers $|A|$, finds the latest time t_c of all the answers, computes key (epk, esk) and a public ZK-SNARK parameter PP, and generates an attestation π_a. All these keys and public parameters can be computed off the blockchain (such as using an intel SGX) to reduce the on-chain overhead. The attestation together with esk are used as ZK-proof's witness uploaded to the public chain. The smart contracts on the public chain directly call *libsnark* to verify the validity of attestation [95]. Since the validators on the same subchain share the subchain ledger, these validators are able to verify the correctness of the attestation through PP, avoiding the leader's misbehavior.

4.3.8 Reward Distribution

Once a task is finished, the ledgers on the public chain and subchains are updated correspondingly, the reward of each answer and the verification fee of each validator/subchain leader are distributed via the execution of smart contracts. In this Chapter, besides the reward of answers, the requesters are supposed to pay the verification fee. In reality, different task requesters may adopt the same or different reward distribution policies, and various

ways could be utilized by the crowdsourcing platforms to charge validation fee. For examples, each worker's received reward may depend on the number of submitted answers or the quality of submitted data; each validator's received verification fee may be determined by the number of verified transactions; and each subchain leader also obtains an amount of verification fee for his effort to help complete transaction verification and communicate with the public chain. After paying the rewards and validation fee, the rest of the deposits will be returned back to the requesters' accounts. But, if a requester is identified as malicious, his deposit will be distributed to the corresponding selected workers. The design of reward distribution mechanism is out of the scope of this Chapter and will be further investigated in our future work, here is an work that may inspire our design of the incentive algorithm [96].

4.4 Performance Analysis

In this section, we first analyze the advantages of zkCrowd from the aspects of the platform architecture, consensus mechanism and cryptographic methodology. Then, we discuss the performance of zkCrowd to resist possible attacks and compare it with the state-of-the-art.

4.4.1 Platform Architecture

Our zkCrowd has a hierarchical architecture of hybrid blockchains, in which diverse tasks (including public and private tasks) can be effectively processed with different levels of privacy protection and permission control. At the same time, zkCrowd well integrates the characteristics of consensuses on the public chain and subchains. The subchain validators

are directly selected from alternative validators with the highest number of votes, avoiding an extra voting process. That is, such a unified election mechanism does not add additional overhead to the blockchain system when selecting validators. Therefore, the subchain setup process can only use the idle computation resources of the public chain. Moreover, when a private task is completed, the subchain validators will be released back to the public chain. The flexibility of subchain establishment in zkCrowd can further reduce the network overhead and enhance the resource utilization.

4.4.2 Consensus Mechanism

4.4.2.1 DPOS Consensus

Compared with the POW blockchain (whose verification speed is 3.3-7 seconds per transaction [97]) and the POS blockchain (whose verification speed is $\frac{1}{12}$ seconds per transaction [98]), the verification speed of DPOS can reach tens of thousands of transactions per second, which can be applied to enterprise-class crowdsourcing systems [91]. Thus, by adopting the DPOS consensus, the verification speed on the public chain can satisfy the needs of crowdsourcing.

The centralization problem of blockchain networks should be addressed. For examples, in the POW-based Bitcoin network, the total mining power of the top 6 mining pools has exceeded 80% of the entire mining power [99]; in POS systems, a voter with more stakes has a higher influence and rich people will become richer [100]. Although the DPOS consensus is controlled by 21 public validators, their accounting rights rotate every 3 seconds, and for

each time cycle, the public validators must be re-voted. This could avoid the DPOS-based systems becoming centralized.

4.4.2.2 PBFT Consensus

In the PBFT consensus on the subchains, the transactions are verified in a distributed manner. Compared with the existing hybrid blockchain [73] where the private chain is hosted by a single agent on public chain, PBFT consensus in zkCrowd elects the leaders in a round robin manner, which can prevent the silent misbehave and single point of failure problems caused by the private chain leaders. Also, our experiments in Section 4.5 show that the PBFT consensus achieves a good performance in execution time, latency and throughput especially for small transaction volume compared with Casper consensus in Ethereum. The results indicate that the PBFT consensus is very suitable for the subchain of zkCrowd.

4.4.3 Cryptographic Methodology

4.4.3.1 Certificate Authority

Before users join zkCrowd, TCA is employed to check the user's true personal identity and his public key and issue a digital certification. This can guarantee that the identity of each participant in the TCA's record must be unique. Thus, the adoption of TCA can prevent Sybil attack while ensuring anonymity.

4.4.3.2 Zero-Knowledge Proof

For the answers of private tasks on the subchain, a zero-knowledge proof is utilized to ensure the answers are unrevealed on the public chain. Meanwhile, the public chain validators can verify the submission process of the task answers by verifying the commitment of ZK-SNARK. On any subchain, since all subchain validator shares the same ledger, they can verify whether the commitment is created correctly by their leaders using the PP of ZK-SNARK.

The on-blockchain computation of ZK-SNARK is actually very light because the verification process is executed by only calling existing library *libsnark*, while the heavy computation of zero-knowledge Prover is done off-blockchain. Therefore, ZK-SNARK can protect the private answers without introducing much computation overhead.

4.4.4 Attack Resistance

4.4.4.1 51% Attack

In the traditional POW or POS-based blockchain networks, if a malicious attacker controls over half of the network computing power/stake, he can use it as a feature of competitive conditions to cancel the payment transactions that have already occurred. In zkCrowd, it adopts the DPOS consensus, the accounting right is divided into 21 public chain validators, thus preventing the 51% attack.

4.4.4.2 Sybil Attack

An attacker may create multiple identities in order to manipulate the reputation system and gain more benefit in the network. To solve this issue, the TCA in zkCrowd performs identity

authentication and detects the misbehavior. That is, if an attacker owns multiple identities, it cannot get a valid certificate to join zkCrowd.

4.4.4.3 Privacy Leakage

Workers' private information may leak to the public when submitting their answers. In zkCrowd, the subchains with zero-knowledge proof ensure the secure submission of answers without revealing private information on the public chain.

4.4.4.4 Free-riding & False-reporting

Free-riding means that workers receive rewards without making real efforts and false-reporting refers to that dishonest requesters try to repudiate the payments. These types of attacks cannot happen in zkCrowd due to the reason that a detailed smart contract with reward policy can ensure the correct process for a task.

4.4.4.5 Byzantine Failure

Some members in the system may make mistakes or send wrong information, which may lead to information corruption and different decisions made by the consensus, thereby destroying system consistency. In zkCrowd, both the public chain (DPOS) and subchain (PBFT) utilize the Byzantine fault tolerance consensus mechanisms, thus guaranteeing different levels of Byzantine Fault Tolerance.

A comparison between zkCrowd and some existing crowdsourcing platforms is summarized in Table 4.2.

Table 4.2: Comparison of Attack Resistance

Attack	**zkCrowd**	**MTurk** [101]	**Dynamo** [102]	**CrowdBC** [67]	**ZebraLancer** [86]
51% Attack	✓	✓	✓	×	×
Sybil Attack	✓	×	✓	✓	✓
Privacy Leakage	✓	×	×	×	✓
Free-riding	✓	✓	✓	✓	✓
False-reporting	✓	×	×	✓	✓
Byzantine Failure	✓	×	×	×	×

✓: realized attack resistance; and ×: vulnerability to attack.

4.5 Performance Evaluation

In this section, extensive experiments are set up to assess the efficiency of DPOS and PBFT by comparing with Casper consensuses on Ethereum (based on POW and POS [103]).

4.5.1 Experiment Environment

Three private blockchains are built with the following experiment configurations. (i) **DPOS-based EOS.io.** The first private blockchain is on the EOS.io platform running DPOS consensus. The desktop with Intel i7-7700k, 32GB RAM, 512 SSD hard drive is used. The private blockchain is built on EOS DAWN-v3.0.0, tested with *txn_test_gen* plugin on binaryen. One producer and one generator are connected to each other. (i) **PBFT-based Hyperledger Fabric.** The second one is on the Hyperledger Fabric platform using PBFT consensus. The desktop with the Ubuntu 16.04 operating system in VMWare is used, in which Hyperledger Fabric V1.4 is installed and Hyperledger composer is used to interact with the built-in private blockchain. (ii) **Casper-based Ethereum.** The third one is on

the Ethereum platform using Casper consensus. The desktop with Intel i7-7700k, 32GB RAM, 512 SSD hard drive is used. The Eth Geth 1.8.22-stable application is installed to build connections to Ethereum main network, and the solidity compiler is deployed locally to compile the smart contracts and create a private chain.

For each blockchain, we create two accounts to send and receive transactions, respectively. On the three chains, the Go language is exploited to deploy the smart contracts and a chaincode to accomplish multiple answer submission transactions in a row between different accounts. The performance metrics include: (i) **Execution time**, which is the total time interval when all transactions are completed on the platform. (ii) **Throughput**, which refers to the number of successful transactions per second, indicating the verification capability of a blockchain network. (iii) **Latency**, which is an average difference between a transaction completion time and its deployment time.

4.5.2 Result and Analysis

For the same transaction volume, each experiment is run 20 times, and the averaged results are presented in the following figures.

First, the comparison results in Fig. 4.3 and 4.4 reveal the performance of DPOS and Casper, where the transaction volume increases from 5000 to 10000. Since the transaction volume on the public chain could be relatively large, we mainly focus on performance under the scenario with high transaction volume.

From Fig. 4.3, we can easily conclude that the execution time of both EOS.io and Ethereum is getting longer as the transaction volume increases. Particularly, when the

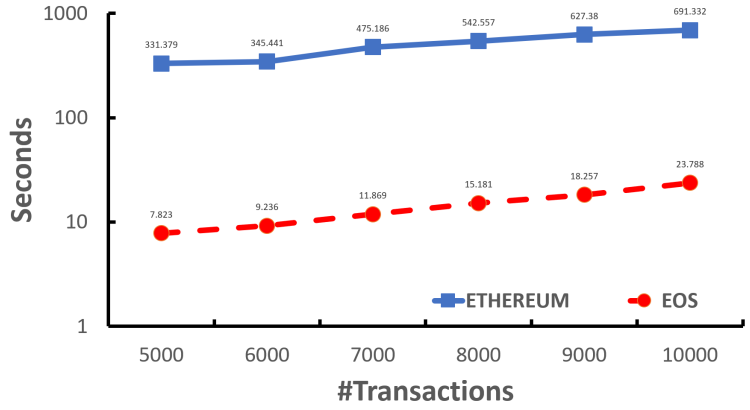


Figure 4.3: Semi-log Plot on Execution Time with 5000-10000 Transactions

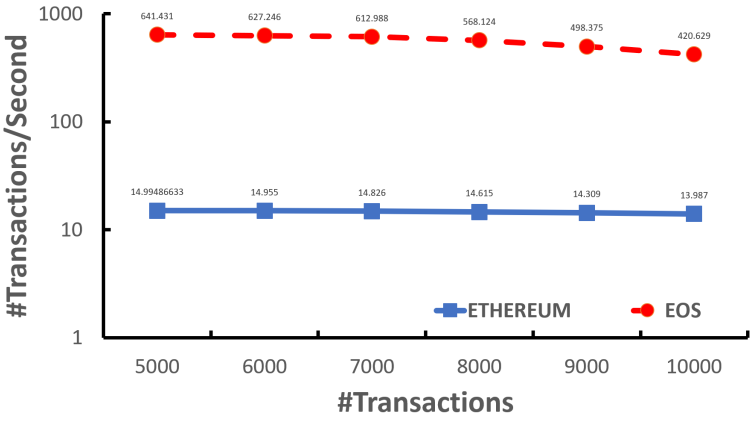


Figure 4.4: Semi-log Plot on Throughput with 5000-10000 Transactions

transaction volume reaches 10000, the execution time of Ethereum is about 30 times of the execution time of EOS.io. The throughput is reported in Fig. 4.4. When the transaction volume reaches 10000, EOS.io can still verify more than 400 transactions per second, while Ethereum only has over 10 transactions per second.

Therefore, the above results validate the advanteges of the DPOS consensus on the public chain.

Then, the performance comparison between PBFT and Casper are presented in Fig. 4.5 to Fig. 4.10 . On the subchains, the transaction volume is usually small, so in the experiments, the number of transactions is changed from 10 to 5000.

Fig. 4.5 shows the execution time of answer submission transactions when transaction volume varies. From the results, we can observe that the execution time of both Hyperledger and Ethereum platforms is getting longer as the transaction volume increases. With the same transaction volume, the execution time of the Hyperledger is much shorter than that of Ethereum.

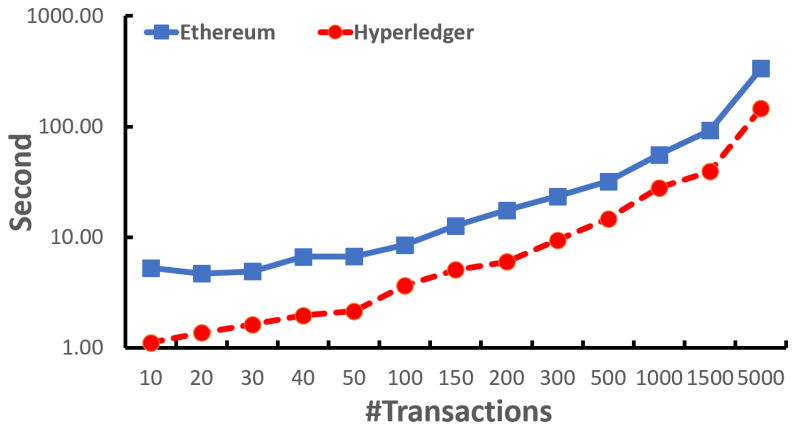


Figure 4.5: Semi-log Plot on Execution Time with Maximum 5000 Transactions

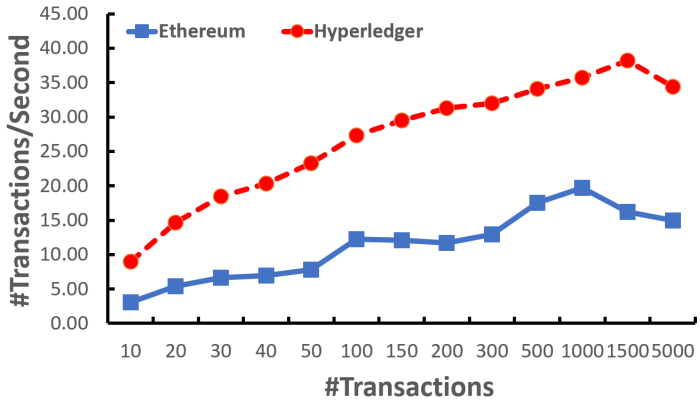


Figure 4.6: Semi-log Plot on Throughput with Maximum 5000 Transactions

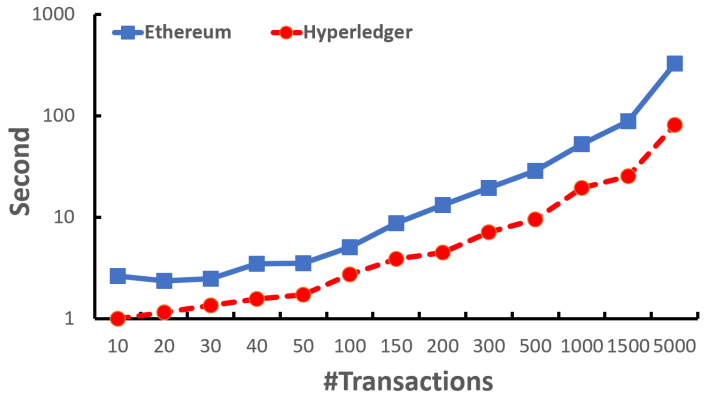


Figure 4.7: Semi-log Plot on Latency with Maximum 5000 Transactions

Since the subchain on zkCrowd is dynamically established based on task demand, usually the transaction volume is not very large on the subchains. Thus, the performance with small transaction volume should be paid more attention. As shown in Fig. 4.8, the results with the transaction volume less than 50 are reported. With the increase of the transaction volume, the execution time of the private chain on Hyperledger grows slowly, while the execution time of the private chain on Ethereum has slight fluctuations. The main reason lies in the

randomness of the block producing rate in POW consensus mechanism. More specifically, from the results of Fig. 4.8, one can find that with small transaction volume, the execution time of Hyperledger is 3-5 seconds shorter than that of Ethereum.

From Fig. 4.6, we can conclude that Hyperledger outperforms Ethereum in terms of throughput. The throughput of both Hyperledger and Ethereum platforms is enhanced first when the transaction volume grows up and then is reduced when the transaction volume becomes larger than a certain value. Such a certain value to obtain the maximum throughput indicates the platform capacity, and performance bottleneck will appear when the transaction volume is larger than this certain value. In Fig. 4.6, Hyperledger achieves its maximum throughput when the transaction volume is around 1500, and Ethereum reaches its maximum throughput when the transaction volume is 1000, which implies that Hyperledger has a larger capacity.

Next, the throughput with the transaction volume smaller than 50 is analyzed. In Fig. 4.9, the results show that (i) the throughput of both Hyperledger and Ethereum increases steadily with the increase of the transaction volume; and (ii) the throughput of Hyperledger is about three times of the throughput of Ethereum.

Finally, Fig. 4.7 presents the latency of Hyperledger and Ethereum by verifying different numbers of transactions. As can be seen from the figure, when the transaction volume is increased, the latency of both platforms grows, and the latency of the Hyperledger is shorter than that of the Ethereum. To obtain more insights, we also compare 5 sets of transactions in Fig. 4.10 and find that the latency of the Hyperledger is about half of the Ethereum.

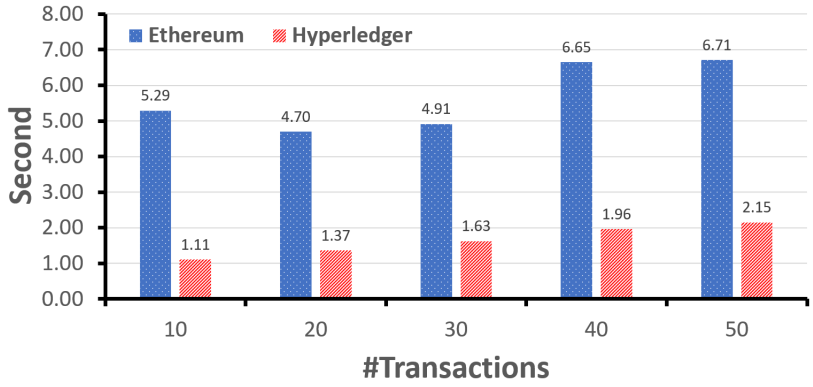


Figure 4.8: Execution Time with Maximum 50 Transactions

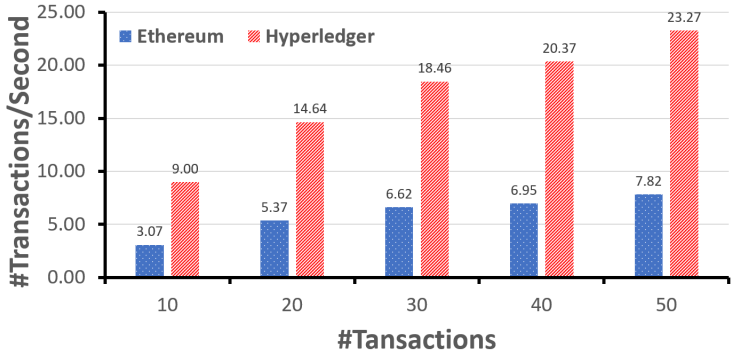


Figure 4.9: Throughput with Maximum 50 Transactions

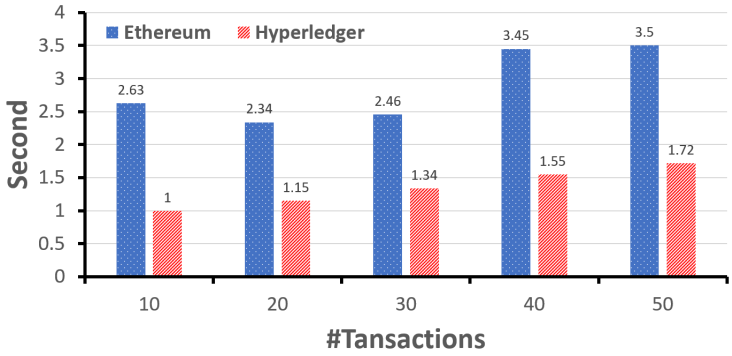


Figure 4.10: Latency with Maximum 50 Transactions

4.6 Chapter Summary

In this Chapter, a hybrid blockchain platform, named zkCrowd, is proposed for crowdsourcing. By integrating a public chain running the DPOS consensus and subchains running the PBFT consensus, our zkCrow can achieve nice features: (i) higher transaction throughput and less execution time compared with traditional POW/POS-based blockchain; (ii) diverse privacy protection and access control for different crowdsourcing tasks with employing smart contracts and zero-knowledge proof; and (iii) resistance to severe attacks outperforming the state-of-the-art. Finally, intensive experiments are performed to validate the effectiveness of zkCrowd via showing the superiority of DPOS and PBFT over Casper.

As our pilot work, the experiments in this Chapter mainly focuses on the efficiency of consensuses used on zkCrowd. In our future work, the fundamental functions of zkCrowd and cross-chain communications will be realized, and the performance of the whole platform will be further improved.

CHAPTER 5

SECURE VERIFIABLE AGGREGATION FOR BLOCKCHAIN-BASED FEDERATED AVERAGING

5.1 Introduction

With the massive growth of network data volume in the big data area, more and more people started to pay attention to information security, particularly personal data privacy [4, 6, 21, 104, 105, 106, 107, 108, 109]. The European Union introduced the "General Data Protection Regulation (GDPR) with strong policies to protect users' privacy and data security in 2016 [110]. The free flow of data under the premise of security and compliance has become the general trend. In 2016, Google first proposed a type of distributed machine learning framework named *Federated Learning (FL)*[111]. The goal of FL is to ac,. hieve cooperative modeling among nodes on the based on data privacy and security, thereby improving the effectiveness of AI models [112]. Subsequently, increasing systematic concepts and optimization schemes on FL were proposed in recent years[113, 114, 115].

There are two challenging problems in federal learning. The first problem is the removal of the centralization of the aggregator. Typically, in order to calculate the global update in federal learning, there is a node or server that plays the role of an aggregator to collect the local gradient updates of all users and calculate the global gradient to share with all users [116]. The centralized aggregator brings the uplink and downlink delay and congestion problems, as well as the single point of failure problem. Blockchain technology has similar distributed characteristics with Federal Learning and becomes a potential tool to achieve

decentralization in FL [116, 117, 118, 119]. Besides decentralization, blockchain also makes the historical records of global updates transparent and immutable. Also, the blockchain's incentive mechanism can track the contribution of each data provider towards the globally optimized model, so that participants can be treated fairly, thereby attracting more data sharers. Although blockchain has these advantages and potentially can improve the transparency and trust in the machine learning process, there are still some issues and challenges that need to be addressed when combining blockchain and federated learning, such as scalability, smart contract vulnerabilities, etc. If we use the blockchain network to replace the aggregator, it is not easy to guarantee confidentiality in the gradient sharing process due to the openness feature of the ledger. Furthermore, the security and privacy protection of the Blockchain often focuses on the consensus protocol level, not the privacy of the data itself [120, 121]. In this chapter, we focus on the secure aggregation of FL parameter uploads and how to better integrate FL with blockchain systems [122].

The second challenge in applying FL is the protection of data privacy [123]. In federated learning, users do not need to upload data to the server. Instead, they upload local gradient updates that are trained locally. However, the existing research has shown that gradients and model parameters can still leak users' private data [23, 124, 125, 126]. In addition, Hitaj *et al.*, Melis *et al.*, Fredrikson *et al.* pointed that even if only the local gradients are sent, training data still can be reversely inferred from the model. Therefore, different techniques were introduced to solve the privacy issues in FL [127, 128, 129]. The work of [130] introduces Secure Multi-party Computation (MPC) to guarantee complete zero-knowledge under a well-

simulated framework. Differential Privacy & k-anonymity were also used to add noise to the data, thus masking some sensitive attributes [127]. Moreover, Homomorphic Encryption was adopted in [131] to protect users' data privacy during parameters and gradients sharing in FL. However, all of these above solutions bring some new problems. The existing research shows that when the number of servers increases, MPC will yield dramatically high communication costs [132]; differential privacy has a disparate impact on model accuracy, especially under FL scenario [133]; and Homomorphic Encryption is not practically applicable to cloud computing because the server cannot decide the carries by the encrypted data [134]. Thus, among those different techniques, the secret-sharing mechanism proposed in [123, 135] is a practical and most commonly used mechanism at the moment if we well control the number of the participants.

In this chapter, we aim to solve the data privacy problem via using Verifiable Secret Sharing in the framework of federal averaging combined with blockchain architecture, for which there are three major challenges:

- One of the characteristics of FL is that the connections between the end-user devices and the server are usually wireless, so the communication cost is relatively high, and thus the number of communications between the blockchain and end-users needs to be reduced.

- Since all blockchain nodes synchronize the same ledger information, the secret shares of the parameter data cannot be stored directly on the chain. Thus the challenge lies in the management of the interaction among the on-chain information, end-user devices,

and the external storage.

- The devices involved in FL are mostly mobile devices that may frequently be switched on and off. Therefore, our solution needs to be robust in dealing with the drop-out problem.

We address the above challenges in this paper and proposed the following novel mechanisms. The proposed architecture uses a combination of blockchain and DHT structure to completely replace the processing and storage functions of the centralized server. We use smart contracts to control user access, add masks and utilize cryptographic tools to users' uploaded data to ensure that users' personal information cannot be easily inferred. Our contribution is three-folded:

- We adopt a federated averaging algorithm to reduce the communication time between users and blockchain.

- We use a double-masking-then-encrypt approach to ensure that the local update is not directly stored on the blockchain, while the smart contract can still calculate the correct weighted average.

- We propose a PVSS algorithm to solve the update drop-out problem and secure the communication between blockchain and DHT.

The organization of this chapter is as follows: The framework of the proposed architecture is described in Section. 5.2. Section. 5.3 and Section. 5.4. We summarize our designs and

give theoretical security analysis in Section. 5.5 and conduct the experiments and evaluations in Section. 5.6. Finally, the conclusion and future work are given in Section. 5.7.

5.2 System Architecture

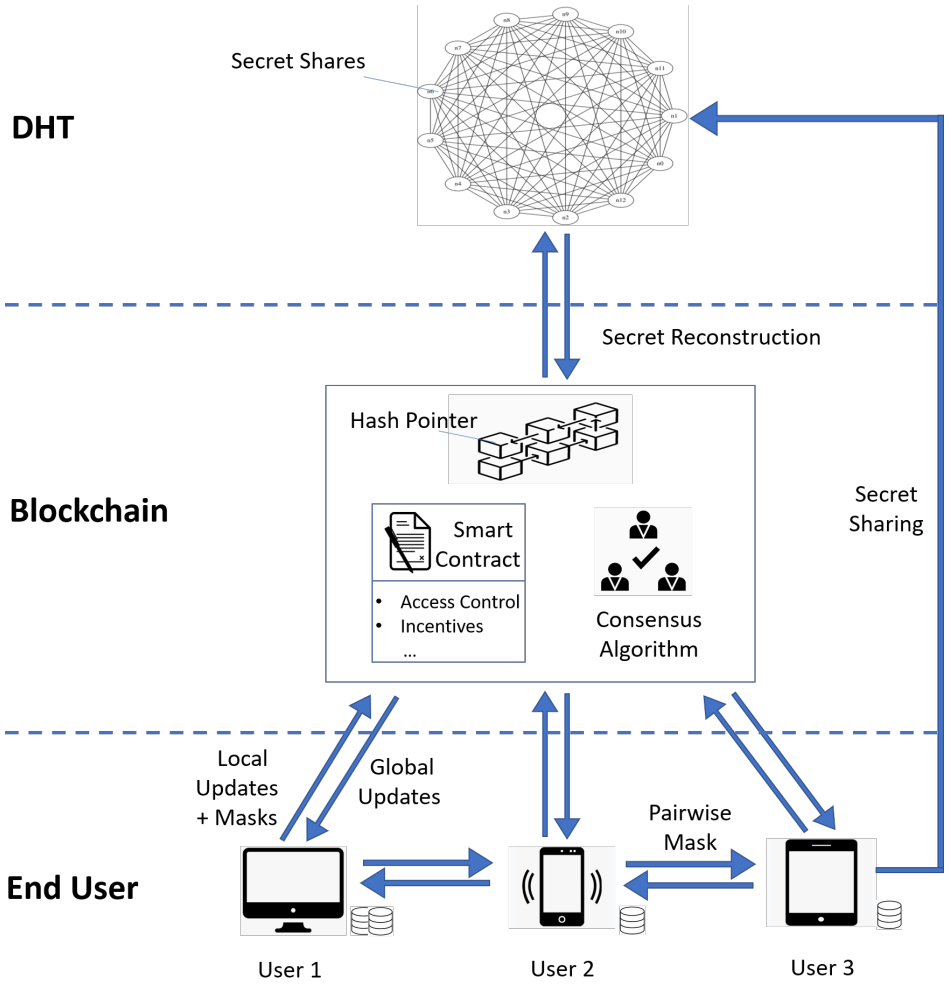


Figure 5.1: System Architecture

Our system is mainly divided into three levels. The bottom layer is the end-user layer. We assume that all end users have certain computing capabilities and can collect data and

perform local gradient update calculations, and they share a pairwise mask between each other and calculate an individual mask to secure the local update parameters. The middle layer is the blockchain network, which mainly serves as the role of an aggregator. Through smart contract, the network can automatically coordinate all local updates and periodically update the global update to end-users. For some blockchain networks (such as Hyperledger Fabric), different federated learning tasks can be run in parallel on various channels. By adding an access control mechanism and incentive mechanism to the blockchain, users can be rewarded by sharing data. The top layer is DHT, which is mainly responsible for the storage of secret data for the pairwise mask. Through interaction with the blockchain network, the purpose of protecting data privacy is achieved.

Next, we will go through how the three layers work cooperatively from the following two aspects: masking approach on local updates and public verifiable secret sharing for privacy protection.

5.3 Mask-then-Encrypt

According to federated averaging, the multiple iterations of $w_i(t)$ calculation on the end-user side can greatly reduce the number of communications between blockchain and users compared with traditional stochastic gradient descent federated learning.

In the following section, we aim to solve the privacy problem when the blockchain is combined with federated learning. Though original data is not uploaded, a large number of existing studies have shown that the uploaded gradient or local parameters transformations

of the original data and can leak the privacy of original data. Therefore, we adopted a mask-then-encrypt security aggregation idea. In this part, we mainly introduce the idea of masking. That is, in the process of uploading the local parameter $w_i(t)$, we let any two clients share a random number in pairs to mask the real value. Therefore, the blockchain nodes cannot know every real $w_i(t)$ but can still perform federated learning.

Let us take a look at a simple example. Assuming that there are only two users who want to upload local updates respectively, and we will modify the data as follows:

$$w_1'(t) = w_1(t) + r, \tag{5.1}$$

$$w_2'(t) = w_2(t) - r. \tag{5.2}$$

In this way, we have the aggregation performed on the blockchain side as:

$$w_1'(t) + w_2'(t) = w_1(t) + r + w_2(t) - r = w_1(t) + w_2(t). \tag{5.3}$$

In a more general case, each pairwise mask is represented as $S_{i,j}, S_{j,i}$, then:

$$w_i'(t) = w_i(t) + \sum_{i<j} S_{i,j} - \sum_{i>j} S_{j,i}. \tag{5.4}$$

In this way, the correctness of aggregation on the server side can also be verified as

follows:

$$\begin{aligned} w(t) &= \sum w_i'(t) \\ &= \sum \{w_i(t) + \sum_{i<j} S_{i,j} - \sum_{i>j} S_{j,i}\} \\ &= \sum w_i(t). \end{aligned} \tag{5.5}$$

However, the above model only works with the following two assumptions:

- All paired users should have the same upload frequency.
- All users must remain online at all times to ensure no drop-out situation occurs.

Therefore, we propose the following two improvements to this protocol.

5.3.1 Double Masking

To solve the uploading frequency issue, the most straight forward idea is to require users unwilling to upload local updates to upload only a copy of the mask information. However, this approach will cause the blockchain to directly obtain mask information, which means that $w_i(t)$ can be directly inferred based on mask information.

Therefore, a double masking idea is put forward here. On the basis of the above pairwise masking, each user also needs to use a random seed b_i to generate an individual mask, denoted as $M_i = PRG(b_i)$. The individual mask does not cancel anything, but we ensure that the blockchain can not get the individual mask and the pairwise mask at the same time. In the following encryption process, the blockchain will request the user's individual

key when the user is online to eliminate the individual mask; and when the user is not online, it will request the Diffie-Hellman key and compute the pairwise mask, noted that an honest node will not reveal information of two masks at the same time.

In this way, the information uploaded by the user is equivalent to:

$$w_i'(t) = w_i(t) + PRG(b_i) + \sum_{i<j} S_{i,j} - \sum_{i>j} S_{j,i}. \tag{5.6}$$

5.3.2 Secret Sharing

Next, we discuss how to solve the drop-out problem. It is a critical issue because once the user is offline, the pairwise mask paired with the user cannot be eliminated when aggregation is performed. Therefore, we need to certain mechanism to recover the mask information when the user offline or when there is no reply from the user.

In the next section, we mainly introduce the PVSS technology when dealing with the drop-out problem. We disperse the secret of b_i and $S_{i,j}$, then store the information into DHT participants in the form of secret shares. The secret-sharing and reconstruction phases are moved into the DHT storage to reduce the cost of direct communication between users and blockchain.

5.4 The Publicly Verifiable Secret Sharing(PVSS)

Our scheme is distributed into two basic sub-protocols. The first is the distribution protocol. The mobile user authorizes the smart contract in blockchain to divide the secret S (the secret

is b_i or $S_{i,j}$) into n shares of $s_1, s_2, ..., s_n$, and distributes them to each DHT participant. The second is the reconstruction protocol, which needs at least t authorized DHT participants in the subset to cooperate to recover the secret S.

The original secret sharing scheme is defined as only resisting passive attacks, which means that its security if all participants honestly follow the protocol. If there are dishonest entities, for example, when the smart contract distributes incorrect shares to one or more DHT participants, or when DHT participants provide incorrect shares, the original scheme cannot work.

We can use the verifiable secret sharing scheme (VSS) to resist those active attacks. For example, the smart contract can encrypt the share of each participant and attach a commitment, then transmit it on the public channel. In this way, the shares can be publicly verified without any leakage. This type of VSS solution has both the attributes of *hiding* and *binding*, where *hiding* means that for a polynomial of degree $t-1$, any t shares are sufficient to reconstruct the secret via interpolation, but less than t shares reveal no information about it, and *binding* means that every participant receives, in addition to its private share, a global commitment $PROOF$ to the polynomial is added as a verifiable valid share.

It is worth noting that although it is possible to publicly verify the correctness of the shares distributed to each participant, it is not guaranteed that each participant submits the correct share. At the same time, when the blockchain interacts with the DHT, one cannot guarantee that there is a secure private channel between the blockchain and each participant. Therefore, all information should be encrypted and transmitted on the open channel. This

is why we need a publicly verifiable secret sharing scheme(PVSS) to protect secrets. So, the difference between PVSS and VSS is that any party, not just the participants of the protocol, can verify the validity of the shares distributed by the dealer.

5.4.1 Algorithm Analysis

5.4.1.1 General Idea

In the system initialization phase, the owner of the secret D and all DHT participants P_i need to register with the prover in the smart contract and select a pair of public and private keys (x_i, y_i).

In the distribution phase, D first generates a share s_i for each participant P_i, and publicly encrypts $E_i(s_i)$ with the public key of the DHT participant. In order to ensure that $E_i(s_i)$ encrypts the correct share s_i, an corresponding proof $PROOF_D$ is added. In addition, $PROOF_D$ also serves as a commitment of D to the secret S, ensuring that the same secret is obtained in the reconstruction protocol, and uploaded to the blockchain. Users who know the encryption algorithm E_i and its corresponding public key can verify the correctness of the encrypted share. The user runs a non-interactive verification algorithm of $PROOF_D$ through the smart contract to verify the correctness of s_i in $E_i(s_i)$. When one or more shared authentication fails, the execution of the protocol is suspended. Thus, anyone can verify the correctness of sharing based on public information.

During the reconstruction of the secret, the DHT participants in the authorized subset decrypt the shared s_i from $E_i(s_i)$ and save it in DHT. These participants can add another

proof $PROOF_{P_i}$ to prove the correctness of s_i. Finally, the smart contract in the blockchain calculates and reconstructs the secret S based on the shares of all authorized participants.

5.4.1.2 Implementation

Next, we introduce the specific protocol of the PVSS:

The basic tool for constructing our publicly verifiable encryption scheme is the high-power discrete logarithm proof protocol. In [136], they call the sub-protocol as DLEQ() protocol.

In this chapter, we use G_q to denote a group of prime order q; g_1,g_2 denote the generators of G_q using appropriate public procedures. Therefore, we assume that finding the discrete logarithm of any two elements in the group is difficult. DLEQ() protocol is based on the Diffie-Hellman algorithm and zero-knowledge proof to ensure the correct operation of secret shares in the protocol process. Suppose the prover knows that $h_1 = (g_1)^l$ and $h_2 = (g_2)^l$, and wants to prove to the participants that he knows such l that makes the relationship $log_{g_1} h_1$ and $log_{g_2} h_2$ stands up.

1. The secret owner allows smart contact to randomly choose an integer $u \in Z_q^*$, and send $a_1 = (gen_1)^u$ and $a_2 = (gen_2)^u$ to the DHT participant.

2. The DHT participant returns a random number $r \in Z_q^*$ and send back to the smart contract.

3. The smart contract calculates $k = u - rl(mod\ q)$ and sends k to the DHT participant.

4. The DHT participant check the equation that $a_1 = gen_1^k h_1^r$ and $a_2 = gen_2^k h_1^r$. If it is true, he believes that the prover knows l; otherwise, do not believe it.

If the random number r in 2) is replaced by a hash value, we can use the $Sign(gen_1, h_1, gen_2, h_2)$ method constructed by Fiat and Shamir in [137] to achieve a non-interactive zero Knowledge proof verification, thereby reducing the back and forth communication between blockchain and the DHT.

1. the smart contract randomly selects an integer $u_1 \in Z_q^*$, and calculates $a_1 = (gen_1)^{u_1}$, $a_2 = (gen_2)^{u_1}$ and $r = H(m||gen_1||gen_2||h_1||h_2||a_1||a_2)$, $c = rl + k(mod\ q)$, and then sends a_1, a_2 and c to the DHT participant.

2. The DHT participant computes the $r = H(m||gen_1||gen_2||h_1||h_2||a_1||a_2)$ and checks $gen_1^c = h_1^r a_1$ and $gen_2^c = h_2^r a_2$.

Using the aforementioned non-interactive subprotocol to be specific to PVSS. In the initialization phase, Participant P_i generates a private key $x_i \in Z_q^*$, and calculate $y_i = G^{x_i}$ as its corresponding public key, where G denotes a selected generator of G_q.

In the distribution phase, the dealer firstly distributes one secret among selected DHT participants $P_1, P_2, ..., P_n$. The dealer then picks a random polynomial $p(x)$ with the largest degree of $t-1$ with coefficients $\alpha_0, \alpha_1, ..., \alpha_{t-1}$:

$$p(x) = \sum_{j=0}^{t-1} \alpha_j x^j. \tag{5.7}$$

The dealer keeps the above polynomial secret, and publishes a corresponding commitments $C_j = g^{\alpha_j}$. The dealer also uses the participant's public keys to encrypt and publish the related shares $Y_i = y_i^{p(i)}$. The dealer then produces a proof of knowledge of $p(i)$ to show

the consistency of the encrypted shares which satisfies: $X_i = g^{p(i)}$ and $Y_i = y_i^{p(i)}$. Through $Sign(g_1, h_1, g_2, h_2)$ method, the generated proof is composed of hash of the random challenge r and n responses k_i.

In the verification phase, The verifier in DHT calculates:

$$X_i = \prod_{j=0}^{t-1} C_j^{i^j}, \tag{5.8}$$

and then obtains:

$$a_{1i} = g^{k_i} X_i^r,\ a_{2i} = y_1^{k_i} Y_i^r. \tag{5.9}$$

During the Reconstruction phase, each participant uses his/her own private key x_j to calculate its share $s_i = Y_i^{\frac{1}{x_i}}$, and then publish a zero-knowledge proof that S_i is the correct decryption of Y_i to the smart contract and other participants. Assuming that all participants in a certain authorized subset can get the correct shared $s_j (j = i_0, i_1, ..., i_{t-1})$. In other words, when the smart contract collects at least t shares, the secret S can be calculated by the Lagrange interpolation formula:

$$S = \sum_{i=0}^{t-1} s_i \prod_{j=0}^{t-1} \frac{s_j}{s_j - s_i}. \tag{5.10}$$

Therefore, in the above algorithm, all participants do not know l and the polynomial $p(x)$, but can verify correctness of the scheme. The smart contract only needs to obtain t related shares to complete the secret reconstruction of a single transaction. Since the private

key of participants will not be disclosed during the process, it can be reused in PVSS many times in the future. The secret corresponding to each PVSS scheme is the local updates uploaded by a user/dealer. In this way, we can use the federated averaging mechanism in Section.3.3 for modeling.

5.5 Design Idea and Security Analysis

5.5.1 Public Trusted Communication Channel

Our solution well integrates secret sharing into federated learning and blockchain deployment. At the initiate stage, the smart contract and all DHT participants need to create a pair of keys to register through the smart contract for further interaction. This step creates a public and trusted secure communication channel for all filtered users . In this way, PVSS can be publicly verified without the need to separately establish private communication channels.

5.5.2 Encryption Security and Public verifiability

All secret shares are encrypted with the public keys of participants during the communication process, so only the corresponding DHT participant can unlock the share. Also, the zero-knowledge proof commitment in PVSS ensures that the smart contract and verifier can publicly verify the correctness of the transmitted content without revealing sensitive information.

Specifically, assuming $G = g^{\alpha}$, $X_i = g^{\beta}$, then cracking the encrption of the shares means calculating $g^{\alpha\beta}$. According to The Diffie-Hellman assumption that it is impossible to crack the encryption of the share in a linear time. In summary, this guarantees that all participants

in the authorized subset can cooperate to restore the correct secret; and participants in any unauthorized subset cannot obtain any (partial) secret information through cooperation.

5.5.3 On chain Storage

Taking into account the storage space problem on the chain, we combine DHT to store encrypted shares, so only the pointer and hash value corresponding to the DHT storage are stored on the chain. When there are multiple federated learning tasks, we can add a serial number to each task in the smart contract to ensure the orderliness of storage, and to handle the access control and priority of different tasks more conveniently.

5.5.4 No dropout issue in share reconstruction

We know that the communication between blockchains and users is mostly transmitted wirelessly, thereby stability cannot be guaranteed, while the connection between DHT and blockchain is much stable. So, our solution can greatly reduce the number of communications between the blockchain and the end user device as long as we have sufficient trust in the DHT participants. When the end user has a dropout problem, the blockchain will request the recovery process of secret sharing with DHT. Therefore, in the process of share recovery, we do not have to worry about the dropout problem of DHT participants, which greatly improves the efficiency of secret recovery.

5.6 Evaluation

To evaluate the performance of the PVSS scheme. We implement and configure the PTO-TECT project from IMB research [138] and build a local platform for threshold secure cryptography. The experiments were conducted on a laptop VMware with Ubuntu 16.04 OS with Intel i7-7660U CPU@2.50GHz with 16 GB RAM.

We use Pseudorandom Functions as a KDF and PRNGs from IBM crypto util library. The distributed RSA prime generation was written in Java using java.security library. And for the Elliptic Curve Diffie Hellman Key Agreement (ECDH), we used the NIST P-256 curve from OpenSSL.

Unlike traditional secret sharing, we construct a public sharing of a secret. This sharing can be validated by anyone who holds the public keys of the shareholding participants. Specifically, the **verifyShare()** verifies that a particular encrypted share is valid using the zero-knowledge and **getSecretCommitment()** returns the Pedersen commitment to the secret that is shared.

The degree of the secret-sharing polynomial is chosen to be the same as the max secret reconstruction threshold of participant t out of n, where our t is set to be $\lceil \frac{n}{3} \rceil$.

Our server listens for the client connections on port (8080 + INDEX-OF-SERVER). As we knew that DHT participants need to be trusted in our setting, so we have 10 defined permissions to each secret share at the client's side for access control, including store, info, delete, etc.

On the premise that the security level is 112 bits and the RSA public key length is 3072

bits, we measured the time for computing secret shared polynomials for all clients and also the execution time on the clients end.

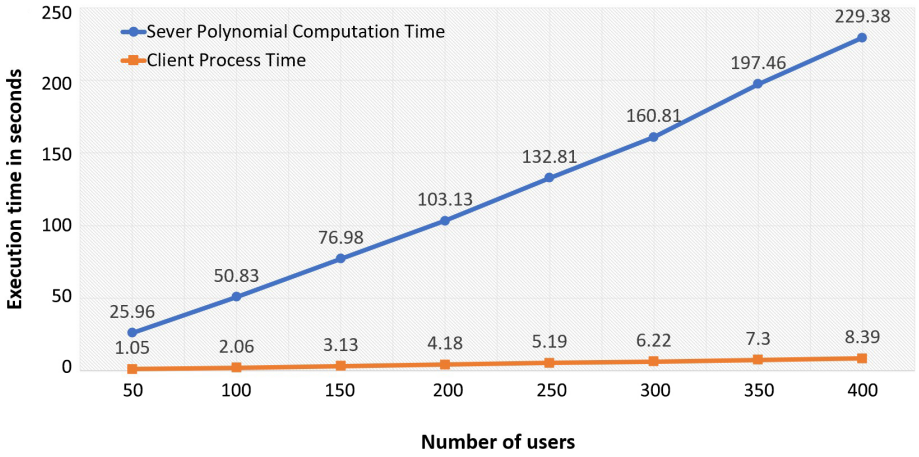


Figure 5.2: Polynomial computation time and client process time

From Fig.5.2, we can see that as the number of users increases, the amount of polynomial calculations on the server-side for shares also increases; when the number of participants is less than 400, the increase in time cost basically appears a linear trend. Therefore, the scheme of sharing shares with all participating users is currently only applicable to small-scale user groups. An improved solution could be to set a fixed number of users n participating in secret sharing, and randomly select a subset of users from DHT nodes to participate in each time, so as to ensure that the running time is within a controllable range.

At the same time, we can see that the running time of the PVSS clients is relatively stable, which also includes the process of importing HTTPS CA certificates. So, that means there is no need to provide a lot of computing power for DHT participants.

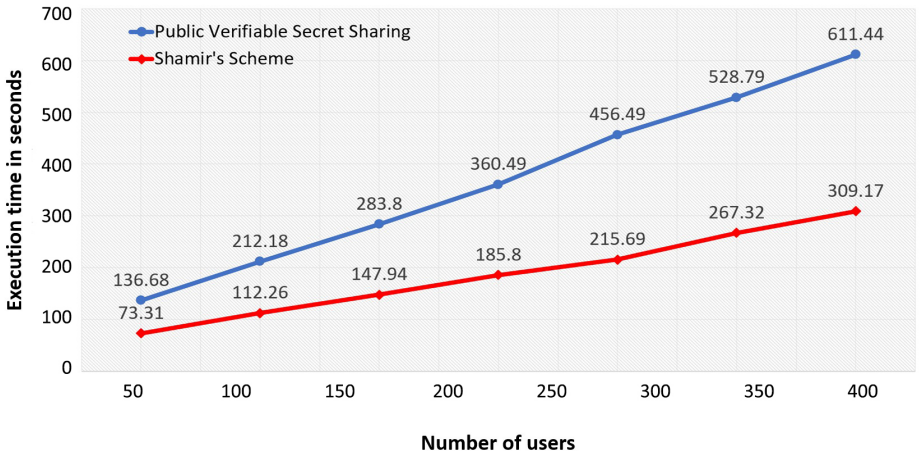


Figure 5.3: Total Executing Time

We also implemented traditional Shamir's threshold secret sharing with the same parameters, and compared the overall execution time (including communication time and secret reconstruction time) of the PVSS and Shamir's schemes on the server side. As shown in Fig.5.3, when the number of users is less than 400, the execution time of PVSS is approximately twice that of Shamir's, which demonstrates the trade-off between privacy protection and communication/computation costs.

5.7 Chapter Summary

In this chapter, we design a blockchain with DHT structure to replace the centralized aggregator in FL, and propose a double-masking-then-encrypt approach to ensure that the local updates cannot be inferred from a classifier model. Finally, a PVSS algorithm is introduced to solve the drop-out problem and secure the communications. Our experiment results show a trade-off between privacy protection level and efficiency. We plan to adopt

the non-interactive PVSS algorithm to further reduce the overall running time.

From the perspective of blockchain smart contract design, apart from some access control, the smart contract in the blockchain will complete at least two main calculations. The first is the reconstruction protocol for the secret shares, and the second is the calculation of federated averaging. Considering the on-chain communication is costly, the following two research directions are proposed:

1. Put on-chain communication into a trusted and safe operating environment, such as intel SGX;

2. Use VSSR technology proposed in [139] to solve the efficiency problems caused by Lagrange interpolation.

CHAPTER 6
FUTURE WORK

The most instinctive thought of achieving high scalability of blockchain is to expand the size of the block and shorten the interval between blocks. However, the main drawback of expanding block size is that it requires more extra storage on the chain. At the same time, it will cost additional time to spread to the whole network, which will lead to more forks and orphaned blocks in the blockchain network.

So, my future research is to enhance the scalability of current blockchain network and achieve the privacy protection feature on chain. I would like to conduct this research from the perspective of the verification algorithms and combining blockchain with federated learning. Here are the objectives and design of approaches to achieves those goals.

Objective 1: Improved TPS through distributed parallel computing

Here are four aspects to improve TPS I would research on: (1) Transaction Verification Approach: The sharding tech. It splits the transactions on the network into different fragments, so that each node only needs to process a small part of the transactions. Then through parallel processing with other nodes on the network, it can increase the efficiency in verification work. (2) Transaction Format: We could adopt the idea of Segregated Witness to rearrange the transaction format. Since the digital signature in the transaction data packaged into the block is only needed during the verification phase, the witness can be separated from other transaction data and the layout of each transaction content can be rearranged. Take the script signature out of the structure of the transaction content and put it at the

bottom. This can improve the verification efficiency during parallel verification. (3) Block Format: Referring to the practice of Bitcoin-NG, blocks can be classified according to the function in the blockchain. For example, the two tasks of leader election and transaction recording are allocated to two types of blocks to realize the parallel execution of mining and transaction recording. This approach can be directly applied to existing blockchains without causing a hard fork and increasing the throughput of the blockchain.

Objective 2: Development of efficient ZK-SNARK proof for transaction verification

We will work on implementing an effective loop recursive snark based on elliptic curve encryption in Ethereum for smart contract calls.

CHAPTER 7
CONCLUSION

This dissertation focuses on solving the problem of privacy protection in blockchain systems.

First we investigate the existing problems of traditional blockchain network and propose a novel coin hopping attack in multiple blockchain systems under IoT environment. We demonstrate the feasibility and prove the harm to the miners of this attack.

Second, we combine various cryptographic techniques and access control mechanisms to design a task-based blockchain platform for crowdsoursing, which solves the centralization problem and various privacy issues of traditional crowdsourcing systems. Our system outperforms existing similar designs in terms of security, and experimentally verifies that the consensus protocol used in the platform performs well in different blockchain transactions volume.

Thirdly, we solve the privacy problem in combining blockchain platform with federated learning by adopting double-masking and pvss technology, but there is still room for further improvement in efficiency.

In the future, we will focus more on solving the scalability problem of blockchain, and we believe this dissertation will also inspire subsequent research towards the privacy protection in blockchain area.

REFERENCES

[1] "Comparison of mining pools," https://en.bitcoin.it/wiki/Comparisonofminingpools.

[2] J. Li, S. Cheng, Z. Cai, J. Yu, C. Wang, and Y. Li, "Approximate holistic aggregation in wireless sensor networks," *ACM Trans. Sens. Networks*, vol. 13, no. 2, pp. 11:1–11:24, 2017.

[3] Z. Cai, Z. He, X. Guan, and Y. Li, "Collective data-sanitization for preventing sensitive information inference attacks in social networks," *IEEE Transactions on Dependable and Secure Computing*, vol. 15, no. 4, pp. 577–590, 2018.

[4] Z. Cai and X. Zheng, "A private and efficient mechanism for data uploading in smart cyber-physical systems," *IEEE Transactions on Network Science and Engineering*, vol. 7, no. 2, pp. 766–775, 2020.

[5] J. Wang, Z. Cai, and J. Yu, "Achieving personalized k-anonymity-based content privacy for autonomous vehicles in CPS," *IEEE Trans. Ind. Informatics*, vol. 16, no. 6, pp. 4242–4251, 2020.

[6] Z. Cai and Z. He, "Trading private range counting over big iot data," in *39th IEEE International Conference on Distributed Computing Systems, ICDCS 2019, Dallas, TX, USA, July 7-10, 2019*. IEEE, 2019, pp. 144–153.

[7] Z. Duan, W. Li, X. Zheng, and Z. Cai, "Mutual-preference driven truthful auction mechanism in mobile crowdsensing," in *39th IEEE International Conference on Distributed Computing Systems, ICDCS 2019, Dallas, TX, USA, July 7-10, 2019*. IEEE,

2019, pp. 1233–1242.

[8] T. Zhu, T. Shi, J. Li, Z. Cai, and X. Zhou, "Task scheduling in deadline-aware mobile edge computing systems," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4854–4866, 2019.

[9] Z. He, Z. Cai, and X. Wang, "Modeling propagation dynamics and developing optimized countermeasures for rumor spreading in online social networks," in *35th IEEE International Conference on Distributed Computing Systems, ICDCS 2015, Columbus, OH, USA, June 29 - July 2, 2015*. IEEE Computer Society, 2015, pp. 205–214.

[10] X. Zheng, L. Tian, G. Luo, and Z. Cai, "A collaborative mechanism for private data publication in smart cities," *IEEE Internet Things J.*, vol. 7, no. 9, pp. 7883–7891, 2020.

[11] M. Siddula, Y. Li, X. Cheng, Z. Tian, and Z. Cai, "Privacy-enhancing preferential LBS query for mobile social network users," *Wirel. Commun. Mob. Comput.*, vol. 2020, pp. 8 892 321:1–8 892 321:13, 2020.

[12] Z. Cai and Q. Chen, "Latency-and-coverage aware data aggregation scheduling for multihop battery-free wireless networks," *IEEE Transactions on Wireless Communication*, 2021.

[13] X. Zheng, G. Luo, and Z. Cai, "A fair mechanism for private data publication in online social networks," *IEEE Trans. Netw. Sci. Eng.*, vol. 7, no. 2, pp. 880–891, 2020.

[14] Z. Xiong, Z. Cai, Q. Han, A. Alrawais, and W. Li, "Adgan: Protect your location privacy in camera data of auto-driving vehicles," *IEEE Transactions on Industrial Informatics*, 2020.

[15] K. Li, G. Luo, Y. Ye, W. Li, S. Ji, and Z. Cai, "Adversarial privacy preserving graph embedding against inference attack," *IEEE Internet of Things Journal*, 2020.

[16] K. Li, G. Lu, G. Luo, and Z. Cai, "Seed-free graph de-anonymiztiation with adversarial learning," in *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, 2020, pp. 745–754.

[17] M. Xu, F. Zhao, Y. Zou, C. Liu, X. Cheng, and F. Dressler. Blown: A blockchain protocol for single-hop wireless networks under adversarial sinr. [Online]. Available: https://arxiv.org/abs/2103.08361

[18] C. Liu, H. Guo, M. Xu, S. Wang, D. Yu, J. Yu, and X. Cheng. Extending on-chain trust to off-chain – a trustworthy vaccine shipping example. [Online]. Available: https://arxiv.org/abs/2106.15934

[19] Wikipedia down? current status and problems. Accessed 2019-01-18. [Online]. Available: https://downdetector.com/status/wikipedia

[20] M. Isaac, K. Benner, and S. Frenkel. (2017) Uber hid 2016 breach, paying hackers to delete stolen data. [Online]. Available: https://www.nytimes.com/2017/11/21/technology/uber-hack.html

[21] X. Zheng and Z. Cai, "Privacy-preserved data sharing towards multiple parties in industrial iots," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 5, pp. 968–979, 2020.

[22] Z. He, Z. Cai, Q. Han, W. Tong, L. Sun, and Y. Li, "An energy efficient privacy-preserving content sharing scheme in mobile social networks," *Pers. Ubiquitous Comput.*, vol. 20, no. 5, pp. 833–846, 2016.

[23] Y. Liang, Z. Cai, J. Yu, Q. Han, and Y. Li, "Deep learning based inference of private information using embedded sensors in smart devices," *IEEE Netw.*, vol. 32, no. 4, pp. 8–14, 2018.

[24] X. Zheng, Z. Cai, and Y. Li, "Data linkage in smart internet of things systems: A consideration from a privacy perspective," *IEEE Commun. Mag.*, vol. 56, no. 9, pp. 55–61, 2018.

[25] X. Zheng, Z. Cai, J. Li, and H. Gao, "Location-privacy-aware review publication mechanism for local business service systems," in *2017 IEEE Conference on Computer Communications, INFOCOM 2017, Atlanta, GA, USA, May 1-4, 2017*. IEEE, 2017, pp. 1–9.

[26] X. Zheng, Z. Cai, J. Yu, C. Wang, and Y. Li, "Follow but no track: Privacy preserved profile publishing in cyber-physical social systems," *IEEE Internet Things J.*, vol. 4, no. 6, pp. 1868–1878, 2017.

[27] Z. Xiong, W. Li, Q. Han, and Z. Cai, "Privacy-preserving auto-driving: A gan-based approach to protect vehicular camera data," in *2019 IEEE International Conference on Data Mining, ICDM 2019, Beijing, China, November 8-11, 2019*, J. Wang, K. Shim, and X. Wu, Eds. IEEE, 2019, pp. 668–677.

[28] J. Wang, Z. Cai, Y. Li, D. Yang, J. Li, and H. Gao, "Protecting query privacy with differentially private k-anonymity in location-based services," *Pers. Ubiquitous Comput.*, vol. 22, no. 3, pp. 453–469, 2018.

[29] Z. He, Z. Cai, and J. Yu, "Latent-data privacy preserving with customized data utility

for social network data," *IEEE Trans. Veh. Technol.*, vol. 67, no. 1, pp. 665–673, 2018.

[30] J. Zhang, P. Lu, Z. Li, and J. Gan, "Distributed trip selection game for public bike system with crowdsourcing," in *2018 IEEE Conference on Computer Communications, INFOCOM 2018, Honolulu, HI, USA, April 16-19, 2018*, 2018, pp. 2717–2725.

[31] X. Duan, C. Zhao, S. He, P. Cheng, and J. Zhang, "Distributed algorithms to compute walrasian equilibrium in mobile crowdsensing," *IEEE Trans. Industrial Electronics*, vol. 64, no. 5, pp. 4048–4057, 2017.

[32] S. Nakamoto. (2009) Bitcoin: A peer-to-peer electronic cash system. [Online]. Available: http://www.bitcoin.org/bitcoin.pdf

[33] "Blockchain info: detailed information and charts on all bitcoin transac- tions and blocks." https://blockchain.info/.

[34] C. Mooney and S. Mufson, "Why the bitcoin craze is using up so much energy," *The Washington Post*, 2018.

[35] Y. Velner, J. Teutsch, and L. Luu, "Smart contracts make bitcoin mining pools vulnerable," in *International Conference on Financial Cryptography and Data Security*. Springer, 2017, pp. 298–316.

[36] O. Schrijvers, J. Bonneau, D. Boneh, and T. Roughgarden, "Incentive compatibility of bitcoin mining pool reward functions," in *International Conference on Financial Cryptography and Data Security*. Springer, 2016, pp. 477–498.

[37] M. Rosenfeld, "Analysis of bitcoin pooled mining reward systems," *CoRR*, vol. abs/1112.4980, 2011. [Online]. Available: http://arxiv.org/abs/1112.4980

[38] M. Fredrikson and B. Livshits, "Zø: An optimizing distributing zero-knowledge compiler," in *Proceedings of the 23rd USENIX Security Symposium, San Diego, CA, USA, August 20-22, 2014.*, 2014, pp. 909–924.

[39] B. Parno, J. Howell, C. Gentry, and M. Raykova, "Pinocchio: nearly practical verifiable computation," *Commun. ACM*, vol. 59, no. 2, pp. 103–112, 2016.

[40] M. Conoscenti, A. Vetro, and J. C. De Martin, "Blockchain for the internet of things: A systematic literature review," in *Computer Systems and Applications (AICCSA), 2016 IEEE/ACS 13th International Conference of.* IEEE, 2016, pp. 1–6.

[41] I. Eyal and E. G. Sirer, "Majority is not enough: Bitcoin mining is vulnerable," in *International conference on financial cryptography and data security.* Springer, 2014, pp. 436–454.

[42] A. Sapirshtein, Y. Sompolinsky, and A. Zohar, "Optimal selfish mining strategies in bitcoin," in *International Conference on Financial Cryptography and Data Security.* Springer, 2016, pp. 515–532.

[43] I. Eyal, "The miner's dilemma," in *Security and Privacy (SP), 2015 IEEE Symposium on.* IEEE, 2015, pp. 89–103.

[44] L. Luu, R. Saha, I. Parameshwaran, P. Saxena, and A. Hobor, "On power splitting games in distributed computation: The case of bitcoin pooled mining," in *Computer Security Foundations Symposium (CSF), 2015 IEEE 28th.* IEEE, 2015, pp. 397–411.

[45] P. Veena, S. Panikkar, S. Nair, and P. Brody, "Empowering the edge-practical insights on a decentralized internet of things," *Empowering the Edge-Practical Insights on a*

Decentralized Internet of Things. IBM Institute for Business Value, vol. 17, 2015.

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

[47] A. Dorri, S. S. Kanhere, and R. Jurdak, "Towards an optimized blockchain for iot," in *Proceedings of the Second International Conference on Internet-of-Things Design and Implementation*. ACM, 2017, pp. 173–178.

[48] L. Guo, Y. Li, and Z. Cai, "Minimum-latency aggregation scheduling in wireless sensor network," *J. Comb. Optim.*, vol. 31, no. 1, pp. 279–310, 2016.

[49] L. Yu and Z. Cai, "Dynamic scaling of virtual clusters with bandwidth guarantee in cloud datacenters," in *35th Annual IEEE International Conference on Computer Communications, INFOCOM 2016, San Francisco, CA, USA, April 10-14, 2016*. IEEE, 2016, pp. 1–9.

[50] C. Liu, M. Xu, H. Guo, X. Cheng, Y. Xiao, D. Yu, B. Gong, A. Yerukhimovich, and S. andWeifeng Lv. Tokoin: A coin-based accountable access control scheme for internet of things. [Online]. Available: https://arxiv.org/abs/2011.04919

[51] A. Dorri, S. S. Kanhere, and R. Jurdak, "Blockchain in internet of things: challenges and solutions," *CoRR*, vol. abs/1608.05187, 2016. [Online]. Available: http://arxiv.org/abs/1608.05187

[52] What blockchain means for you, and the internet of things. [Online]. Available: https://www.ibm.com/blogs/internet-of-things/watson-iot-blockchain/

[53] M. Mihaylov, S. Jurado, N. Avellana, K. Van Moffaert, I. M. de Abril, and A. Nowé, "Nrgcoin: Virtual currency for trading of renewable energy in smart grids," in *European Energy Market (EEM), 2014 11th International Conference on the*. IEEE, 2014, pp. 1–6.

[54] V. Buterin and V. Griffith, "Casper the friendly finality gadget," *CoRR*, vol. abs/1710.09437, 2017. [Online]. Available: http://arxiv.org/abs/1710.09437

[55] V. Buterin *et al.*, "A next-generation smart contract and decentralized application platform," *white paper*, 2014.

[56] Eos.io: The most powerful infrastructure for decentralized applications. [Online]. Available: https://eos.io/

[57] Litecoin: The cryptocurrency for payments. [Online]. Available: https://litecoin.org/

[58] Bitcoin cash: Peer-to-peer electronic cash. [Online]. Available: https://www.bitcoincash.org/

[59] Bitcoin block reward halving countdown. [Online]. Available: https://www.bitcoinblockhalf.com/

[60] The ethereum block explore. [Online]. Available: https://etherscan.io/blocks

[61] K. J. O'Dwyer and D. Malone, "Bitcoin mining and its energy footprint," in *25th IET Irish Signals Systems Conference 2014 and 2014 China-Ireland International Conference on Information and Communications Technologies (ISSC 2014/CIICT 2014)*, June 2014, pp. 280–285.

[62] J. Bonneau, A. Miller, J. Clark, A. Narayanan, J. A. Kroll, and E. W. Felten, "Sok:

Research perspectives and challenges for bitcoin and cryptocurrencies," in *Security and Privacy (SP), 2015 IEEE Symposium on*. IEEE, 2015, pp. 104–121.

[63] Ethereum and bitcoin price. [Online]. Available: https://www.tradingview.com/symbols/ETHBTC/

[64] A. Judmayer, A. Zamyatin, N. Stifter, A. G. Voyiatzis, and E. Weippl, "Merged mining: Curse or cure?" in *Data Privacy Management, Cryptocurrencies and Blockchain Technology - ESORICS 2017 International Workshops, DPM 2017 and CBT 2017, Oslo, Norway, September 14-15, 2017, Proceedings*. Springer, 2017, pp. 316–333. [Online]. Available: https://doi.org/10.1007/978-3-319-67816-0_18

[65] A. Madeira. (2017) What is merged mining bitcoin and namecoin litecoin and dogecoin? [Online]. Available: http://https://www.cryptocompare.com/mining/guides/what-is-merged-mining-bitcoin-namecoin-litecoin-dogecoin/

[66] X. Zhang, G. Xue, R. Yu, D. Yang, and J. Tang, "Keep your promise: Mechanism design against free-riding and false-reporting in crowdsourcing," *IEEE Internet of Things Journal*, vol. 2, no. 6, pp. 562–572, 2015.

[67] M. Li, J. Weng, A. Yang, and W. Lu, "Crowdbc: A blockchain-based decentralized framework for crowdsourcing," *IACR Cryptology ePrint Archive*, vol. 2017, p. 444, 2017.

[68] Y. Wang, Z. Cai, Z. Chi, X. Tong, and L. Li, "A differentially k-anonymity-based location privacy-preserving for mobile crowdsourcing systems," in *2017 International Conference on Identification, Information and Knowledge in the Internet of Things,*

IIKI 2017, Shandong, China, October 19-21, 2017, 2017, pp. 28–34.

[69] T. Zhu, G. Li, W. Zhou, and P. S. Yu, "Differentially private data publishing and analysis: A survey," *IEEE Trans. Knowl. Data Eng.*, vol. 29, no. 8, pp. 1619–1638, 2017.

[70] Y. Zhang and M. van der Schaar, "Reputation-based incentive protocols in crowdsourcing applications," in *Proceedings of the IEEE INFOCOM 2012, Orlando, FL, USA, March 25-30, 2012*, 2012, pp. 2140–2148.

[71] C. Tanas, S. Delgado-Segura, and J. Herrera-Joancomartí, "An integrated reward and reputation mechanism for MCS preserving users' privacy," in *Data Privacy Management, and Security Assurance - 10th International Workshop, DPM 2015, and 4th International Workshop, QASA 2015, Vienna, Austria, September 21-22, 2015. Revised Selected Papers*, 2015, pp. 83–99.

[72] Z. Chi, Y. Wang, Y. Huang, and X. Tong, "The novel location privacy-preserving CKD for mobile crowdsourcing systems," *IEEE Access*, vol. 6, pp. 5678–5687, 2018.

[73] B. Jia, T. Zhou, W. Li, Z. Liu, and J. Zhang, "A blockchain-based location privacy protection incentive mechanism in crowd sensing networks," *Sensors*, vol. 18, no. 11, p. 3894, 2018.

[74] Z. He, Z. Cai, J. Yu, X. Wang, Y. Sun, and Y. Li, "Cost-efficient strategies for restraining rumor spreading in mobile social networks," *IEEE Trans. Veh. Technol.*, vol. 66, no. 3, pp. 2789–2800, 2017.

[75] Y. Liang, Z. Cai, Q. Han, and Y. Li, "Location privacy leakage through sensory data,"

Secur. Commun. Networks, vol. 2017, pp. 7 576 307:1–7 576 307:12, 2017.

[76] L. Zhang, Z. Cai, and X. Wang, "Fakemask: A novel privacy preserving approach for smartphones," *IEEE Trans. Netw. Serv. Manag.*, vol. 13, no. 2, pp. 335–348, 2016.

[77] M. Siddula, Y. Li, X. Cheng, Z. Tian, and Z. Cai, "Anonymization in online social networks based on enhanced equi-cardinal clustering," *IEEE Trans. Comput. Soc. Syst.*, vol. 6, no. 4, pp. 809–820, 2019.

[78] S. Zhu, Z. Cai, H. Hu, Y. Li, and W. Li, "zkcrowd: A hybrid blockchain-based crowdsourcing platform," *IEEE Trans. Ind. Informatics*, vol. 16, no. 6, pp. 4196–4205, 2020.

[79] S. Zhu, W. Li, H. Li, L. Tian, G. Luo, and Z. Cai, "Coin hopping attack in blockchain-based iot," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4614–4626, 2019.

[80] S. Cheng, Z. Cai, J. Li, and H. Gao, "Extracting kernel dataset from big sensory data in wireless sensor networks," *IEEE Trans. Knowl. Data Eng.*, vol. 29, no. 4, pp. 813–827, 2017.

[81] S. Cheng, Z. Cai, and J. Li, "Curve query processing in wireless sensor networks," *IEEE Trans. Veh. Technol.*, vol. 64, no. 11, pp. 5198–5209, 2015.

[82] Z. Cai, R. Goebel, and G. Lin, "Size-constrained tree partitioning: Approximating the multicast k-tree routing problem," *Theor. Comput. Sci.*, vol. 412, no. 3, pp. 240–245, 2011.

[83] Z. Cai, G. Lin, and G. Xue, "Improved approximation algorithms for the capacitated multicast routing problem," in *Computing and Combinatorics, 11th Annual International Conference, COCOON 2005, Kunming, China, August 16-29, 2005, Proceedings*,

ser. Lecture Notes in Computer Science, L. Wang, Ed., vol. 3595. Springer, 2005, pp. 136–145.

[84] Z. Duan, W. Li, and Z. Cai, "Distributed auctions for task assignment and scheduling in mobile crowdsensing systems," in *37th IEEE International Conference on Distributed Computing Systems, ICDCS 2017, Atlanta, GA, USA, June 5-8, 2017*, K. Lee and L. Liu, Eds. IEEE Computer Society, 2017, pp. 635–644.

[85] M. Xu, S. Liu, D. Yu, X. Cheng, S. Guo, and J. Yu. Cloudchain: A cloud blockchain using shared memeory consensus and rdma. [Online]. Available: https://arxiv.org/abs/2106.04122

[86] Y. Lu, Q. Tang, and G. Wang, "Zebralancer: Private and anonymous crowdsourcing system atop open blockchain," in *38th IEEE International Conference on Distributed Computing Systems, ICDCS 2018, Vienna, Austria, July 2-6, 2018*, 2018, pp. 853–865.

[87] S. Matsumoto and R. M. Reischuk, "IKP: turning a PKI around with decentralized automated incentives," in *2017 IEEE Symposium on Security and Privacy, SP 2017, San Jose, CA, USA, May 22-26, 2017*, 2017, pp. 410–426.

[88] Smilo: The hybrid blockchain platform with a conscience. Accessed 2019-07-28. [Online]. Available: https://smilo.io/

[89] Aergo: It's not a blockchain. it's the blockchain for business. Accessed 2019-07-28. [Online]. Available: https://www.aergo.io/

[90] Xinfin: Enterprise ready hybrid blockchain for global trade and finance. Accessed 2019-07-28. [Online]. Available: https://www.aergo.io/

[91] K. Samani. (2018) Delegated proof of stake: Features and trade-offs. [Online]. Available: https://multicoin.capital/wp-content/uploads/2018/03/DPoS-Features-and-Tradeoffs.pdf

[92] M. Castro and B. Liskov, "Practical byzantine fault tolerance," in *Proceedings of the Third USENIX Symposium on Operating Systems Design and Implementation (OSDI), New Orleans, Louisiana, USA, February 22-25, 1999*, 1999, pp. 173–186.

[93] H. Sukhwani, J. M. Martínez, X. Chang, K. S. Trivedi, and A. Rindos, "Performance modeling of PBFT consensus process for permissioned blockchain network (hyperledger fabric)," in *36th IEEE Symposium on Reliable Distributed Systems, SRDS 2017, Hong Kong, Hong Kong, September 26-29, 2017*, 2017, pp. 253–255.

[94] A. E. Kosba, A. Miller, E. Shi, Z. Wen, and C. Papamanthou, "Hawk: The blockchain model of cryptography and privacy-preserving smart contracts," in *IEEE Symposium on Security and Privacy, SP 2016, San Jose, CA, USA, May 22-26, 2016*, 2016, pp. 839–858.

[95] E. Ben-Sasson, A. Chiesa, D. Genkin, E. Tromer, and M. Virza, "Snarks for C: verifying program executions succinctly and in zero knowledge," in *Advances in Cryptology - CRYPTO 2013 - 33rd Annual Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2013. Proceedings, Part II*, 2013, pp. 90–108.

[96] Y. Lin, Z. Cai, X. Wang, F. Hao, L. Wang, and A. M. V. V. Sai, "Multi-round incentive mechanism for cold start-enabled mobile crowdsensing," *IEEE Trans. Veh. Technol.*, vol. 70, no. 1, pp. 993–1007, 2021.

[97] K. Croman, C. Decker, I. Eyal, A. E. Gencer, A. Juels, A. E. Kosba, A. Miller, P. Saxena, E. Shi, E. G. Sirer, D. Song, and R. Wattenhofer, "On scaling decentralized blockchains - (A position paper)," in *Financial Cryptography and Data Security - FC 2016 International Workshops, BITCOIN, VOTING, and WAHC, Christ Church, Barbados, February 26, 2016, Revised Selected Papers*, 2016, pp. 106–125.

[98] A. Hertig. How will ethereum scale? Accessed 2019-02-04. [Online]. Available: https://www.coindesk.com/information/will-ethereum-scale

[99] A. Miller, A. E. Kosba, J. Katz, and E. Shi, "Nonoutsourceable scratch-off puzzles to discourage bitcoin mining coalitions," in *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, Denver, CO, USA, October 12-16, 2015*, 2015, pp. 680–691.

[100] G. C. Fanti, L. Kogan, S. Oh, K. Ruan, P. Viswanath, and G. Wang, "Compounding of wealth in proof-of-stake cryptocurrencies," *CoRR*, vol. abs/1809.07468, 2018.

[101] Amazon mechanical turk: Access a global, on-demand, 24x7 workforce. Accessed 2019-08-04. [Online]. Available: https://www.mturk.com/

[102] N. Salehi, L. Irani, M. S. Bernstein, A. Alkhatib, E. Ogbe, K. Milland, and Clickhappier, "We are dynamo: Overcoming stalling and friction in collective action for crowd workers," in *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems, CHI 2015, Seoul, Republic of Korea, April 18-23, 2015*, 2015, pp. 1621–1630.

[103] V. Buterin and V. Griffith, "Casper the friendly finality gadget," *CoRR*, vol.

abs/1710.09437, 2017.

[104] L. Yu, L. Chen, Z. Cai, H. Shen, Y. Liang, and Y. Pan, "Stochastic load balancing for virtual resource management in datacenters," *IEEE Trans. Cloud Comput.*, vol. 8, no. 2, pp. 459–472, 2020.

[105] X. Zheng, L. Tian, G. Luo, and Z. Cai, "A collaborative mechanism for private data publication in smart cities," *IEEE Internet of Things Journal*, vol. 7, no. 9, pp. 7883–7891, 2020.

[106] S. Zhu, Z. Cai, H. Hu, Y. Li, and W. Li, "zkcrowd: a hybrid blockchain-based crowdsourcing platform," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 6, pp. 4196–4205, 2019.

[107] Z. Cai, Z. Duan, and W. Li, "Exploiting multi-dimensional task diversity in distributed auctions for mobile crowdsensing," *IEEE Transactions on Mobile Computing*, 2020.

[108] Z. Cai, Z. Xiong, H. Xu, P. Wang, W. Li, and Y. Pan, "Generative adversarial networks: A survey towards private and secure applications," *CoRR*, vol. abs/2106.03785, 2021. [Online]. Available: https://arxiv.org/abs/2106.03785

[109] Z. Xiong, Z. Cai, D. Takabi, and W. Li, "Privacy threat and defense for federated learning with non-i.i.d. data in aiot," *IEEE Transactions on Industrial Informatics*, pp. 1–1, 2021.

[110] J. Scherer and G. Kiparski, "Buchbesprechungen. feiler, lukas / forgó, nikolaus / weigl, michaela: The eu general data protection regulation (gdpr): A commentary," *Comput. und Recht*, vol. 34, no. 6, pp. 69–70, 2018.

[111] J. Konecný, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon, "Federated learning: Strategies for improving communication efficiency," *CoRR*, vol. abs/1610.05492, 2016.

[112] E. Bagdasaryan, A. Veit, Y. Hua, D. Estrin, and V. Shmatikov, "How to backdoor federated learning," in *The 23rd International Conference on Artificial Intelligence and Statistics, AISTATS 2020, 26-28 August 2020, Online [Palermo, Sicily, Italy]*, 2020, pp. 2938–2948.

[113] T. Yang, G. Andrew, H. Eichner, H. Sun, W. Li, N. Kong, D. Ramage, and F. Beaufays, "Applied federated learning: Improving google keyboard query suggestions," *CoRR*, vol. abs/1812.02903, 2018.

[114] H. Wang, Z. Kaplan, D. Niu, and B. Li, "Optimizing federated learning on non-iid data with reinforcement learning," in *39th IEEE Conference on Computer Communications, INFOCOM 2020, Toronto, ON, Canada, July 6-9, 2020*, 2020, pp. 1698–1707.

[115] J. Kang, Z. Xiong, D. Niyato, S. Xie, and J. Zhang, "Incentive mechanism for reliable federated learning: A joint optimization approach to combining reputation and contract theory," *IEEE Internet Things J.*, vol. 6, no. 6, pp. 10 700–10 714, 2019.

[116] P. Ramanan, K. Nakayama, and R. Sharma, "BAFFLE : Blockchain based aggregator free federated learning," *CoRR*, vol. abs/1909.07452, 2019.

[117] S. Awan, F. Li, B. Luo, and M. Liu, "Poster: A reliable and accountable privacy-preserving federated learning framework using the blockchain," in *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security, CCS*

2019, London, UK, November 11-15, 2019, 2019, pp. 2561–2563.

[118] H. Kim, J. Park, M. Bennis, and S. Kim, "Blockchained on-device federated learning," *IEEE Communications Letters*, vol. 24, no. 6, pp. 1279–1283, 2020.

[119] M. Xu, C. Liu, Y. Zou, F. Zhao, J. Yu, and X. Cheng, "wchain: A fast fault-tolerant blockchain protocol for multihopwireless networks," *IEEE Transactions on Wireless Communication*, 2021.

[120] A. E. Kosba, A. Miller, E. Shi, Z. Wen, and C. Papamanthou, "Hawk: The blockchain model of cryptography and privacy-preserving smart contracts," in *IEEE Symposium on Security and Privacy, SP 2016, San Jose, CA, USA, May 22-26, 2016*, 2016, pp. 839–858.

[121] S. Zhu, W. Li, H. Li, L. Tian, G. Luo, and Z. Cai, "Coin hopping attack in blockchain-based iot," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4614–4626, 2019.

[122] V. Drungilas, E. Vaičiukynas, M. Jurgelaitis, R. Butkienė, and L. Čeponienė, "Towards blockchain-based federated machine learning: Smart contract for model inference," *Applied Sciences*, vol. 11, no. 3, p. 1010, 2021.

[123] K. Bonawitz, V. Ivanov, B. Kreuter, A. Marcedone, H. B. McMahan, S. Patel, D. Ramage, A. Segal, and K. Seth, "Practical secure aggregation for privacy-preserving machine learning," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS 2017, Dallas, TX, USA, October 30 - November 03, 2017*, 2017, pp. 1175–1191.

[124] B. Hitaj, G. Ateniese, and F. Pérez-Cruz, "Deep models under the GAN: information

leakage from collaborative deep learning," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS 2017, Dallas, TX, USA, October 30 - November 03, 2017*, 2017, pp. 603–618.

[125] L. Melis, C. Song, E. D. Cristofaro, and V. Shmatikov, "Exploiting unintended feature leakage in collaborative learning," in *2019 IEEE Symposium on Security and Privacy, SP 2019, San Francisco, CA, USA, May 19-23, 2019*, 2019, pp. 691–706.

[126] M. Fredrikson, S. Jha, and T. Ristenpart, "Model inversion attacks that exploit confidence information and basic countermeasures," in *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, Denver, CO, USA, October 12-16, 2015*, 2015, pp. 1322–1333.

[127] R. C. Geyer, T. Klein, and M. Nabi, "Differentially private federated learning: A client level perspective," *CoRR*, vol. abs/1712.07557, 2017.

[128] Z. Wang, M. Song, Z. Zhang, Y. Song, Q. Wang, and H. Qi, "Beyond inferring class representatives: User-level privacy leakage from federated learning," in *2019 IEEE Conference on Computer Communications, INFOCOM 2019, Paris, France, April 29 - May 2, 2019*, 2019, pp. 2512–2520.

[129] S. Truex, N. Baracaldo, A. Anwar, T. Steinke, H. Ludwig, R. Zhang, and Y. Zhou, "A hybrid approach to privacy-preserving federated learning," in *Proceedings of the 12th ACM Workshop on Artificial Intelligence and Security, AISec@CCS 2019, London, UK, November 15, 2019*, 2019, pp. 1–11.

[130] H. Zhu, "On the relationship between (secure) multi-party computation and (secure)

federated learning," *CoRR*, vol. abs/2008.02609, 2020.

[131] S. Hardy, W. Henecka, H. Ivey-Law, R. Nock, G. Patrini, G. Smith, and B. Thorne, "Private federated learning on vertically partitioned data via entity resolution and additively homomorphic encryption," *CoRR*, vol. abs/1711.10677, 2017.

[132] P. Kairouz, H. B. McMahan, B. Avent, A. Bellet, M. Bennis, A. N. Bhagoji, K. Bonawitz, Z. Charles, G. Cormode, R. Cummings, R. G. L. D'Oliveira, S. E. Rouayheb, D. Evans, J. Gardner, Z. Garrett, A. Gascón, B. Ghazi, P. B. Gibbons, M. Gruteser, Z. Harchaoui, C. He, L. He, Z. Huo, B. Hutchinson, J. Hsu, M. Jaggi, T. Javidi, G. Joshi, M. Khodak, J. Konecný, A. Korolova, F. Koushanfar, S. Koyejo, T. Lepoint, Y. Liu, P. Mittal, M. Mohri, R. Nock, A. Özgür, R. Pagh, M. Raykova, H. Qi, D. Ramage, R. Raskar, D. Song, W. Song, S. U. Stich, Z. Sun, A. T. Suresh, F. Tramèr, P. Vepakomma, J. Wang, L. Xiong, Z. Xu, Q. Yang, F. X. Yu, H. Yu, and S. Zhao, "Advances and open problems in federated learning," *CoRR*, vol. abs/1912.04977, 2019.

[133] E. Bagdasaryan, O. Poursaeed, and V. Shmatikov, "Differential privacy has disparate impact on model accuracy," in *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, H. M. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. B. Fox, and R. Garnett, Eds., 2019, pp. 15 453–15 462.

[134] Z. Cao and L. Liu, "On the weakness of fully homomorphic encryption," *CoRR*, vol.

abs/1511.05341, 2015. [Online]. Available: http://arxiv.org/abs/1511.05341

[135] C. Hu, W. Li, X. Cheng, and J. Yu, "A secure and verifiable secret sharing scheme for big data storage," *IEEE Transaction on Big Data*, vol. 4, no. 3, pp. 341–355, 2018.

[136] B. Schoenmakers, "A simple publicly verifiable secret sharing scheme and its application to electronic voting," in *Annual International Cryptology Conference*. Springer, 1999, pp. 148–164.

[137] A. Fiat and A. Shamir, "How to prove yourself: Practical solutions to identification and signature problems," in *Advances in Cryptology - CRYPTO '86, Santa Barbara, California, USA, 1986, Proceedings*, ser. Lecture Notes in Computer Science, A. M. Odlyzko, Ed., vol. 263. Springer, 1986, pp. 186–194.

[138] J. Resch, C. Cachin, H. Krawczyk, T. Rabin, and C. Stathakopoulou, "protect: A platform for robust threshold cryptography," 2019. [Online]. Available: https://github.com/jasonkresch/protect

[139] S. Basu, A. Tomescu, I. Abraham, D. Malkhi, M. K. Reiter, and E. G. Sirer, "Efficient verifiable secret sharing with share recovery in BFT protocols," in *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security, CCS 2019, London, UK, November 11-15, 2019*, L. Cavallaro, J. Kinder, X. Wang, and J. Katz, Eds. ACM, 2019, pp. 2387–2402.