Towards a Blockchain-based Trustful Mechanism for IoT-Enabled Data Trading Systems

by

Seyednima (Nima) Khezr

A thesis submitted to the Lakehead University in partial
fulfillment of the requirement for the degree of
Doctor of Philosophy (Ph.D.)
in
Electrical and Computer Engineering

Lakehead University, Thunder Bay, Ontario, Canada

Examining Committee Membership

The following served on the Examining Committee for this thesis. The decision of the Examining Committee is by majority vote.

External Examiner:
Amr Youssef
Professor, Dept. of Engineering and Computer Science,
Concordia University, Canada

Internal Member:
Thangarajah Akilan
Assistant Professor, Dept. of Software Engineering,
Lakehead University, Canada

Yimin Yang
Assistant Professor, Dept. of Computer Science,
Lakehead University, Canada

Supervisor:
Rachid Benlamri
Professor, Dept. of Software Engineering
Lakehead University, Canada

Co-supervisor:
Abdulsalam Yassine
Associate Professor, Dept. of Software Engineering
Lakehead University, Canada

Author's Declaration

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

Internet of Things (IoT) devices generate and collect massive amounts of IoT data. Monetizing the flood of data generated by the IoT devices has enabled the creation of IoT data trading systems where individuals and businesses may trade data. In the current IoT data trading systems, a third-party broker collects and manages IoT data for buyers who would like to promote their services and make more profit. However, there are three main challenges that may hinder the development of secure IoT data trading systems. First, there is a lack of data transparency and ownership. While the economic value of IoT data is increasing, it is not very well known how this data can be conceptualized, measured, and monetized in a trusted and transparent way. Second, the literature lacks studies about performance models to demonstrate IoT data trading system usability in real-world systems. Third, the reputation of the trading parties is an important attribute that affects their profitability and trading prosperity. However, current reputation systems are prone to malicious manipulation and single point of failure.

This thesis identifies and addresses the three above challenges for IoT data trading systems. First, this thesis introduces a trustful IoT data trading system based on the blockchain as a means of providing anonymity, security, transparency, and mutual trust for participants. Using a game-theoretic approach, this study develops a strategic negotiation model that maximizes data buyers' utility. To ensure that data owners' IoT data are accessible by trustful buyers, a novel mechanism design is used to impede untruthful buyers from accessing the IoT data. Second, this thesis evaluates the performance of the blockchain-based IoT data trading system using the Hyperledger blockchain. Unlike existing research, this study measures and analyzes transaction throughput, latency, elapsed time, and resource consumption (memory consumption, CPU utilization, and disc read/write operations). Third, this thesis proposes a blockchain-based reputation system capable of avoiding failures by enhancing the Raft consensus mechanism. This thesis also proposes an adaptive learning mechanism that allows the data providers and consumers to enhance their reputation and review credibility scores. Lastly, this thesis carries out extensive theoretical analysis with respect to economic and security properties.

Acknowledgements

I want to express my sincere gratitude to my advisors, Dr. Rachid Benlamri and Dr. Abdulsalam Yassine, for their profound supervision, guidance, support, encouragement, and long hours of discussions. Both inspire me to work hard, strive for perfection, and embrace and deal with challenges. I had the great fortune of being supervised by them. I would also like to thank my committee members for their valuable comments.

I would like to express my sincere gratitude to my love, Shima, for all the encouragement, sacrifices, patience, motivation and, most importantly, her understanding that sometimes "I am tired & busy".

I would like to thank my parents from the bottom of my heart. Although they are physically far away from me, I feel they are with me all the time. Their unconditional support and love have made all my accomplishments possible. Finally, I could not have completed this thesis without the support of my friends, Mahdieh, Navid, and Mohannad, who provided stimulating discussions as well as happy distractions to rest my mind outside of my research.

Finally, I would like to thank Lakehead University, the Department of Engineering, and NSERC for providing financial support during my Ph.D. studies.

Dedication

This thesis is dedicated to the honor of my sister Niloufar. She passed away in hospital following eight years of a long battle against cancer. She was an incredibly strong girl, smart, kind, caring, thoughtful, who took things in her stride and stayed positive throughout her life. I kept my promise to her by completing my Ph.D. Rest in peace, my lovely sister.

Table of Contents

List of Figures x

List of Tables xii

1 Introduction 1

2 Literature Review 6

2.1 Background 6

2.2 Blockchain and Smart Contract 8

2.3 Data Trading Systems 9

2.4 Blockchain-based Reputation Systems 10

2.5 Summary 12

3 Towards a Trustful Game-Theoretic Mechanism for Data Trading in the Blockchain-IoT Ecosystem 14

3.1 Data Market Structure 14

3.1.1 Assumptions 15

3.1.2 Process details of the proposed system 16

3.1.3 Data Value 18

3.1.4 Data Aggregator Utility 18

3.2 A Repeated Game Theoretic Approach for Data Trading Between Buyers 19

3.2.1 Data Buyers Utility 20

3.3 Nash Equilibrium Solutions . . . 22
3.4 Mechanism Design . . . 26
3.4.1 Winner allocation stage . . . 27
3.4.2 Payment Stage . . . 29
3.5 Evaluation of Results . . . 30
3.5.1 Computational Efficiency Analysis . . . 31
3.5.2 Budget Balance Analysis . . . 32
3.5.3 Individual Rationality Analysis . . . 32
3.5.4 Bidding Learning Analysis . . . 33
3.5.5 Truthfulness Analysis . . . 35
3.6 Summary . . . 35

4 Performance Analysis of Blockchain-based IoT Data Trading Systems 37
4.1 Introduction . . . 37
4.2 Data trading model based on blockchain . . . 39
4.3 Implementation Setup Using Hyperledger Fabric . . . 41
4.4 Performance Analysis . . . 44
4.5 Summary . . . 53

5 A Blockchain-based Reputation System for IIoT Data Ecosystem 54
5.1 Introduction . . . 54
5.2 System Model . . . 56
5.2.1 Design Goals . . . 57
5.2.2 System Setup . . . 58
5.2.3 System Registration . . . 60
5.2.4 Payments . . . 60
5.2.5 Reputation Computation and Verification . . . 61
5.3 Security Analysis . . . 62

5.4 Implementation and Experimental Results 64
5.4.1 Raft Consensus Mechanism 65
5.4.2 Experimental Results 66
5.4.3 Performance Analysis 68
5.5 Summary 72

6 Blockchain-based Reputation System For IoT Data Ecosystem: A Utility Maximization Approach 74
6.1 System Model 74
6.1.1 Blockchain Registration 75
6.1.2 Data Providers Utility 78
6.1.3 Data Consumers Utility 78
6.1.4 Adaptive Learning Mechanism 79
6.2 Implementation and Experimental Results 80
6.2.1 Evaluation of Results 81
6.2.2 Performance Analysis 83
6.3 Summary 85

7 Conclusions and Future Works 86
7.1 Conclusions 86
7.2 Future Works 88

References 90

Author's Publications 99

List of Figures

2.1 An example of data trading system. 7

3.1 High-level architecture of the proposed system. 15
3.2 Sequence diagram describes the interaction between DA and buyers. . 17
3.3 Valuation function. This example shows that the valuation v_i of buyers B decreases on the basis of the satisfaction rate φ. Note that this example does not take into account the average of all satisfaction rates. 21
3.4 Computationally efficient property. 31
3.5 Budget balance property. 32
3.6 Individual rationality property. 33
3.7 Learning process. 34
3.8 Boxplot presenting the bid learning processing using fictitious play. . 34
3.9 Truthfulness property. 36

4.1 An example of blockchain-based data trading system. 39
4.2 The blockchain transaction flow on Hyperledger Fabric V2.2 consisting of 2 organizations, each operating one running peer and a database inside a Docker container and orderer running a single channel. . . . 43
4.3 The data generation phase. 45
4.4 Average latency varying workloads and sending rates. 47
4.5 Transaction throughput under open, query, and transfer workloads with varied sending rates. 48
4.6 Running times of test queries. 50

5.1 High-level architecture of the proposed system. 57
5.2 The transaction workflow on Hyperledger Fabric V2.2 using Raft consensus mechanism. 65
5.3 The result of running the RepGossip protocol during the election process with non-negligible link failures. A link failure of 0.1 indicates that every message sent over a particular link has a likelihood of 10% lost. 67
5.4 Comparison between different timeout intervals with respect to the crashed nodes. 68
5.5 Computational costs. 69
5.6 Transaction throughput under *Open*, *Query*, and $Transfer$ workloads with varied sending rates. 70
5.7 Transaction latency under *Open*, *Query*, and $Transfer$ workloads with varied sending rates. 71
5.8 Memory consumption for *Open* workload with 500 transactions. . . . 72
5.9 CPU consumption for *Open* workload with 500 transactions. 72

6.1 High-level architecture of the proposed system. 75
6.2 Data provider's utility in repeated interactions overtime. 82
6.3 Data consumer's utility in repeated interactions overtime. 83
6.4 Performance analysis. 84

List of Tables

2.1 Common types of IoT driven data that may be used for monetization 7
2.2 Comparison between our work and existing studies 10
2.3 Comparison between our work and existing studies 12

3.1 Notations. 19

4.1 Summary of Performance with 500 and 1000 Transactions 46
4.2 Creating the assets in CouchDB 48
4.3 Retrieval the assets from CouchDB 49
4.4 Resource Consumption for Open Workload with 500 Transactions . . 51
4.5 Resource Consumption for Query Workload with 500 Transactions . 51
4.6 Resource Consumption for Transfer Workload with 500 Transactions 51
4.7 Resource Consumption for Open Workload with 1000 transactions . 52
4.8 Resource Consumption for Query Workload with 1000 transactions . 52
4.9 Resource Consumption for Transfer Workload with 1000 transactions 52

5.1 Notations. 59

6.1 Notations. 76
6.2 Experimental setup. 81

Chapter 1

Introduction

The IoT ecosystem is expanding daily, connecting the physical and digital worlds to transform the way we live and do business. With an increasing number of connected devices, a huge amount of data is instantly collected, aggregated, and exploited in new applications in areas such as smart homes, smart cities, and health [1]. According to the estimates conducted by the International Data Corporation (IDC) report [2], 41.6 billion IoT devices will be connected to the Internet by 2025 and generating 79.4 zettabytes of data. As the IoT devices become more instrumented and interconnected, data will grow exponentially [3]. Data from IoT devices has spawned a new data economy in which people and companies can sell and exchange data [4]. As data continues to pile up, the data economy will continue to emerge and enable new IoT data marketplaces. Several companies, such as Terbine and Dfintech, have developed real-world applications to manage and monetize IoT generated data. These applications allow IoT device owners to sell their data to various stakeholders.

Most existing applications that assist the IoT device owners to sell their data in exchange for money fail to clearly explain how, where, or with whom the users' data are being shared. These applications also tend to package the owners' data for sale to other companies repeatedly [5, 6]. Such information changes hands or ownership and the monetary benefit that companies are receiving as a result of selling the data packages is not passed back to IoT device owners [7]. Therefore, one significant aspect that needs to be taken into consideration is ensuring data owners have control of their data and have the autonomy to decide what information is collected, how it is used, and most importantly, how much it is worth. Consequently, designing a trustworthy data market, capable of selling and buying data which incentivizes the participants to maximize their profits under a fair trading mechanism is very critical.

To develop an efficient and trustful data market, a number of challenges need to be addressed. These are summarized below.

1. How to design a trustful and transparent mechanism to control the dissemination of data?
2. Since data records can be sold repeatedly to multiple buyers. A key question to be addressed is how to devise a strategic negotiation model that maximizes the benefit of data owners and buyers?
3. Trust becomes a challenge if data buyers are not trustful and they may misuse the data. A key question is how to impede and impose penalties on untruthful buyers?
4. How to measure the performance of data trading systems in terms of the number of verified transactions, and resource consumption?
5. How can the reputation system take advantage of blockchain technology's tamper-proof characteristics and distributed consensus mechanism?
6. How to design a blockchain-based reputation system that preserves individual identities and review confidentialities?
7. How can sellers and buyers interact in a blockchain-based reputation system to improve their reputation and credibility score?

To tackle the first challenge, we integrated the blockchain as a trustworthy and transparent mechanism that preserves the data owner's control over their data. Blockchain technology is difficult to tamper with and transactions are secure as well as transparent to all parties, including the users who generated the data [6, 8]. As such, blockchain presents a solution for developing a transparent and trustful network for data trading and give the data owners full control of their information and guard their privacy [9].

To tackle the second challenge, we formulate a non-cooperative game in infinite setup between the buyers in which each buyer strategically chooses the bidding price for that specific data to maximize their utilities. In particular, in each one-shot game (stage-game), limited amount of records is traded. The game is played repeatedly and buyers learn from the outcome of the previous stage and update their bids over the next periods to increase their utilities until the demand is met. We show that

the proposed formulation of the non-cooperative game among buyers achieves Nash Equilibrium using pure strategy in a one-shot, discounted finite and infinite repeated horizon, where no buyers in the market can improve their utility by deviating their bids.

To tackle the third challenge, we propose a novel mechanism design based on the trust score. The proposed mechanism design impedes untruthful buyers to obtain the data based on a scoring rule function. Also, if the winner is not fully trusted (i.e., trust score less than 1), we consider a penalty on his/her payment for the current stage of the game.

To tackle the fourth challenge, we measure and analyze transaction throughput, latency, elapsed time, and resource consumption (memory consumption, CPU utilization, and disc read/write operations) using Hyperledger Caliper. Transactions are an important part of the blockchain. To find out how well the system is adding the number of confirmed transactions to the blockchain network, we need to measure and analyze the throughput metric. By default, the underlying data structure of a blockchain does not support an effective method of querying the stored data. To overcome this limitation, we modeled the data in JSON format through CouchDB. The latter supports deploying indexes within the smart contract to make queries more efficient in massive datasets. Indexes enable a database to be queried faster and more efficiently compared to regular queries without indexes. Applying blockchain in data trading systems is not a straightforward task due to high resource consumption. Hence, measuring resource consumption is also important for the efficient management of systems, such as CPU, memory, etc., as well as the successful execution of transactions in the blockchain.

To tackle the fifth, and sixth challenges, we propose a blockchain-based reputation system for the IIoT data ecosystem. We design an anonymous reputation system for the IIoT data ecosystem by leveraging a blind Elliptic Curve Digital Signature (ECDSA) and a non-interactive zero-knowledge proof (ZKP) technique. Furthermore, we build a blockchain network based on the Raft consensus algorithm. With Raft, the proposed system is a crash fault-tolerant, which allows the operation to proceed as planned rather than failing. We further improve the Raft consensus algorithm to avoid single point of failure (SPOF), and link failures. We built a new policy called RepGossip based on the gossip protocol, which withstands the link failures.

To tackle the seventh challenge, we proposed an adaptive learning mechanism in which sellers (data providers) and buyers (data consumers) can learn from their strategies and increase their reputation and credibility scores, respectively.

The main contributions of this research are as follows:

- We formulate a non-cooperative infinitely repeated game in which rational buyers are strategically deciding on their bids and learn from the outcome of each one-shot (stage) game and try to adjust their bids to maximize their utility. The non-cooperative nature of the game in the data market is properly modeled in a one-shot game by carefully defining utility functions. Using this one-shot game as a building block, we then proceed to define finite and infinitely repeated games with a discount factor that captures the repeated interactions among rational buyers.

- We show the existence and uniqueness of the Nash Equilibrium under discontinuous utility function setup. We establish the best response action for buyers and show that such best response action is a standard function, which guarantees the uniqueness of the Nash Equilibrium.

- To ensure data owners' assets are protected and are not being misused within the data trading system, blockchain is used as a means of data transparency and security. Furthermore, we filter out the untruthful buyers based on the scoring function, which is calculated through the trust score. In the payment stage, we consider a penalty for the winner if he/she is not fully trusted. Even with considering a penalty, we ensure that the individual rationality property is set up, which implies the winner buyer receives a non-negative utility.

- We designed a blockchain-based reputation system for the Industrial IoT (IIoT) data ecosystem. We propose an anonymous reputation system for the IoT data ecosystem by leveraging a blind ECDSA (Elliptic Curve Digital Signature Algorithm) and a non-interactive ZKP (Zero-knowledge proof) technique.

- We propose a blockchain-based reputation system where data providers and data consumers can maximize their utility while engaged in the IoT data ecosystem. We offer an adaptive learning mechanism that allows the data providers and consumers to enhance their reputation and review credibility scores.

- We demonstrate and provide a comprehensive theoretical and experimental analysis of the proposed system which satisfies the economic properties including, computationally efficient, truthful, and individually rational.

- Through experimental results, we evaluate the performance of the blockchain-based data trading system using different metrics, such as transaction throughput, latency, and resource consumption under varied scenarios and parameters using Hyperledger Caliper. We show that the proposed system can be easily deployed on IoT devices at a low cost.

The organization of this research is as follows. Chapter 2 presents a background and comprehensive survey of related literature on the data trading systems and the blockchain technology. Chapter 3 presents a trustful blockchain-based repeated game mechanism for IoT data trading ecosystem. Chapter 4 describes the performance analysis of blockchain-based IoT data trading systems. Chapter 5 introduces a blockchain-based reputation system for IIoT data ecosystem. Chapter 6 introduces a utility maximization approach for the blockchain-based reputation system in the IoT data ecosystem. Chapter 7 summarizes the thesis with future research directions.

Chapter 2

Literature Review

In this chapter, in section 2.1, we discuss how the data trading system works and follow by the fundamental concept of blockchain technology in section 2.2. Afterward, we review the existing studies data trading sytems and blockchain-based reputation systems in section 2.3, and 2.4, respectively.

2.1 Background

IoT devices and data-driven applications are generating huge data. With massive number of IoT devices and applications, the total amount of data created by IoT devices will reach 847 ZB per year by 2021 [10]. Indeed, data are becoming the most valuable asset in use as well as in trade [11]. Such valuable data received considerable attention from organizations to tailor their services to potential consumers and make a profit. Table 2.1 summarizes common IoT data types and how IoT driven data can be monetized.

As a result, various IoT applications and devices such as smartphones, wearable devices have brought great convenience to people, while producing huge amounts of data. Data have become a valuable asset, which creates a new business called data trading. According to a report from MarketsandMarkets [14], the data market will grow to 229.4 billion dollars by 2025. Figure 2.1 illustrates an example of a data trading system, in which the market includes data providers, data brokers, and data consumers. The three stakeholders continuously interact with each other while trading data.

Table 2.1: Common types of IoT driven data that may be used for monetization

IoT data types	Examples
Data collected from Apps	Hikers' data which is produced through IoT smart devices are useful for a variety of applications: mapping terrain, trail info, maps, detailed reviews, measuring body performance, and so forth [12]. Applications that assist the hiker in achieving these goals also tend to package the hiker's data for sale to other companies [12].
Sensor data	Pharmaceutical companies looking to improve sales can purchase anonymized health data that is generated by IoT sensors in order to find new customers and more effectively target their product marketing [4].
Smart meter data	Advertisement companies want to know what type of equipment the house is using, so they can target their advertisement to maximize consumers' attention to their products [13].

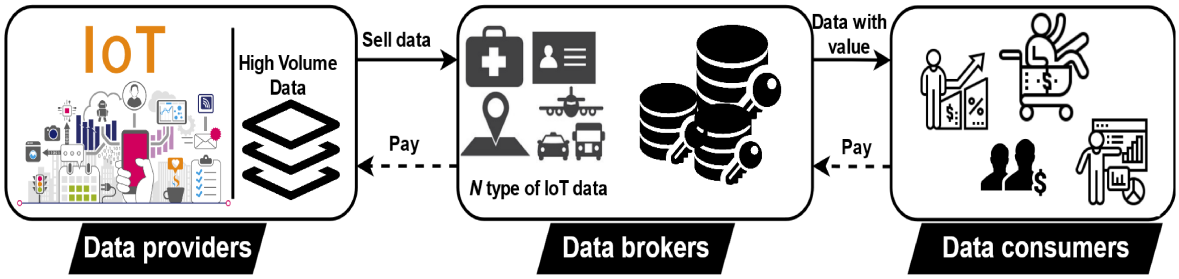


Figure 2.1: An example of data trading system.

In the data market, the data providers (data owners) generate data from various sources, such as smart devices and social networks, and are willing to sell their data in exchange for money. The data market handles N IoT data types (e.g., energy consumption, location, purchase history, etc.). The data broker obtains opt-in based agreement for data collection from each data provider, and then the data broker continuously gathers personal data based on the agreement. Once agreement is completed, the data broker sells the data to data consumers (e.g., data buyers) and pass back the money to providers.

2.2 Blockchain and Smart Contract

A blockchain technology is identified as a distributed ledger technology for peer-to-peer (P2P) network digital data transactions that may be publicly or privately distributed to all users, allowing any type of data to be stored in a reliable and verifiable way [15, 16]. Blockchain is the underlying platform of cryptocurrencies (e.g. Bitcoin, Ethereum) that facilitates a P2P transaction system to eliminate third-party [17, 18]. Every block of data is cryptographically connected with previous blocks by secure hash functions [8]. The methodology of certifying a block whether it is valid or not, in such a way, is called Proof-of-Work (PoW) consensus algorithm (protocols) [19]. The block will be added into the chain after performing the consensus algorithm, every node in the network admits this block and incessantly spreads the chain [20]. Several consensus algorithms such as Proof-of-Stake (PoS), Practical Byzantine Fault Tolerance (PBFT), Delegated Proof of Stake (DPoS), Proof of Authority (PoAu), etc perform a similar job [21]. Blockchains can be classified as public, and depending on their application. Public blockchains have no single owner and are visible by anyone (e.g., Bitcoin). Private (permissioned) blockchains work based on access controls which use privileges (e.g., Hyperledger Fabric). Blockchain network have the following characterizes [22, 9]:

- Decentralized: There is no centralized authority in the blockchain network.
- Highly secure: Blockchains rely heavily on advanced cryptographic techniques.
- Anonymity: A participant of the blockchain network only provides a pseudonym and a unique public/secret key pair.
- Transparency: Any transaction in the blockchain is publicly verifiable for the participants in the network.
- Immutability: Any transaction that is confirmed in the blockchain network cannot be altered.
- Double-spending prevention: In Bitcoin, no malicious participant can spend the same digital currency at different transactions.

Etherem [17] is another distributed blockchain network that supports its own digital currencies, i.e. Ether. It shares many common features with the Bitcoin, e.g. anonymity, openness, transparency, etc. However, Ethereum implements an

account-based model [17] instead of the UTXO model in the Bitcoin. Ethereum first introduced the smart contract [23] to the blockchain network. It is a computer program which is embedded in the blockchain to digitally facilitate, verify or enforce the negotiation of a contract [23]. The smart contract specifies the rules and conditions of involved participants in the network and can take actions (e.g., sharing data, transferring cryptocurrencies) when conditions are met [24]. All smart contracts have a unique address and are stored in the blockchain [17].

2.3 Data Trading Systems

Extensive research has been conducted in order to monetize and trade data [25, 26, 27, 28, 29]. Oh et al. [25] proposed a non-cooperative game for data trading with privacy valuation for data consumers in the IoT environment. The paper introduced a method to unify the unit price of data for data brokers as well as an optimization model to maximize data providers' profits. Similarly, in other work, Oh et al. [26] proposed a data trading model between data owners and consumers as two natural logarithmic functions and a data broker who processes data and provides service to the consumers. This model guaranteed that a data broker will find a global maximum point to reach the best probability deal to sell the data. Tian et al. [27] proposed an optimal contract-based model for data trading between data sellers and consumers. This model maximizes the data seller's payoff while satisfying individual rationality and incentive compatibility properties for data consumers. The work in [28] introduced an iterative auction mechanism for data trading to coordinate the selfish agents in an optimal way to prevent direct access to private information. Khokhar et al. [29] proposed an entropy-based trust computation model to verify the correctness of data from untrusted data providers in the data market. This model utilized the Vickrey–Clarke–Groves auction mechanism for the valuation of data providers' attributes for determining truthful pricing strategies.

To build a more transparent data marketplace, blockchain-based data trading systems are studied in [30, 3, 31, 32, 33], and [34]. Liu et al. [30] introduced an optimal pricing mechanism for data trading in the IoT environment adopted by the two-stage Stackelberg competition based on the blockchain. The model presented a pricing and purchasing mechanism between the data consumer and the market-agency to maximize the profits of both parties. The work in [3] proposes a decentralized fair data trading system, which guarantees the availability of data and fairness between the sellers and buyers. The model implements homomorphic

encryption, double-authentication-preventing signatures, and smart contracts to improve data availability and achieve fairness in data trading between participants. In the work presented in [30], the authors propose a blockchain-data market framework and an optimal pricing mechanism. They designed an optimal pricing mechanism to support efficient data trading in an IoT environment using a two-stage Stackelberg game. Sheng et al. [31] studied a crowd-sourcing data trading system based on blockchain. The model implements a smart contract that enables sellers and buyers to conduct credible and truthful data trading while ensuring the copyright and quality of data. The authors also proposed a semantic-similarity-based auction mechanism to guarantee truthful data trading. Similarly, the authors in [32] investigated a blockchain-based data trading ecosystem that filters out dishonest buyers to guarantee the market's truthfulness. The security model in [32] includes a set of trading protocols based on asymmetric cryptography. The work in [33] proposes a trading model based on Ethereum smart contracts. It incorporates machine learning to guarantee fairness in data trading. All the participants in the blockchain network achieve a consensus on an authentication task, and any potential threats can be identified. Truong et al. [34] proposed a blockchain-based for sharing IoT data, in which data owners can sell their private data. In this framework, smart contracts evaluate access control requests to off-chain encrypted data. Table 2.2 summarizes the comparison of our work and previous studies.

Table 2.2: Comparison between our work and existing studies

Research studies	Decentralization	Smart contracts	Reputation computation	Utility maximization	Performance evaluation
[25], [26]	✗	✗	✗	✓	✗
[27], [28]	✗	✗	✗	✓	✗
[29]	✗	✗	✓	✓	✗
[30], [3]	✓	✓	✗	✓	✗
[31], [32]	✓	✓	✓	✗	✗
[33], [34]	✓	✓	✗	✗	✗
Our work	✓	✓	✓	✓	✓

2.4 Blockchain-based Reputation Systems

In this section, we review the recent studies that use blockchain technology to build transparent and secure reputation systems for IIoT data ecosystem. Then, we com-

pared our proposed system to the recent studies. Soska et al. [35] presented a decentralized anonymous reputation system based on ring linkable signatures and the ZKP method. Customers use the Zerocash anonymous payment method to purchase items. The reviews cannot be linked to previous or any transactions and prevent adversaries to find out the link. This model preserves users' privacy while resisting Sybil attacks. However, the ring signature resulted in a linear overhead when generating the anonymous review. Similarly, in other work, Liu et al. [36] proposed a blockchain-based anonymous reputation system for Industry 4.0 that utilizes the proof-of-stake consensus protocol by leveraging a randomized signature and non-interactive ZKP method. This model used an identity management entity that provides anonymous identities to consumers and retailers. Then, consumers can leave reviews anonymously using rating tokens. However, the centralized structure of the identity management entity may create the concern of a SPOF. Truong et al. [37] introduced a blockchain-based decentralized trust system. They presented a trust model based on an asymmetric relationship between two entities formed through the history of the previous transactions. They investigate the feasibility of their trust system as a bridge between a blockchain platform and decentralized applications. They built a proof-of-concept mechanism and integrated the trust model on top of the Ethereum blockchain. However, they do not examine the security of their model, which presents serious privacy concerns for the users.

In other domains such as energy trading, vehicular and IoT mobile devices, Weerapanpisit et al. in [38] proposed a blockchain-based reputation management system for the IoT systems based on their location. The reputation scores depend on the IoT device's geographical location, and locations are stored in smart contracts. Their approach keeps the reputation system fault-tolerant and consistent across blockchain networks using cloud and fog nodes. The work in [39], proposed a blockchain-based reputation management system for mobile applications through Hyperledger Fabric. They used a mobile application named Aptoide to evaluate their proposed model. Soojan et al. in [40], proposed a two-layered blockchain-based reputation for the vehicular network. The first layer consists of different nodes such as vehicle and roadside units that communicate with a blockchain network to store the transactions from traffic events daily. The global reputation blockchain network is deployed in the second layer to calculate and update the reputation score of member nodes in layer one.

The work in [41], designed a blockchain-based reputation system for energy trading. In this model, the reputation is derived from the behavior of each node according to its role in the peer-to-peer process. To link buyers and sellers and to determine

trading prices, a matchmaking method based on a k-double auction algorithm is used. The matchmaking method balances the fairness among sellers and buyers which is defined as a ratio between reputation score and the average income and cost. However, like the previous work, they do not analyze the security of their proposed model. Zhou et al. [42] proposed a blockchain-based reputation system in the e-commerce environment. In this model, users' reputation scores are generated and updated by all ratings of their transactions weighted by practical transaction characteristics such as transaction duration, amount, and previous reputation scores to prevent unfair rating and collusion attacks. However, they do not measure the performance of their system to determine whether or not it is usable in real-world applications. Li et al. [43] proposed a reputation system for e-commerce applications based on Ethereum blockchain. They built anonymous credentials constructed from two-step blind signatures and the ZKP method. The claimed results show that the system resisted multiple and abnormal rating attacks. The model satisfies the rating and identity privacy, and unlinkability properties. This model relied on a single certificate authority entity to register the users and provide them with identities. However, the certificate authority's structure raises the concern of a SPOF. Table 2.3 provides a summary of the comparison between our work and existing studies.

Table 2.3: Comparison between our work and existing studies

Properties	Work [35]	Work [36]	Work [37, 38, 39]	Work [40, 41, 42]	Work [43]	Our system
Decentralization	✓	✓	✓	✓	✓	✓
Authentication	✓	✓	✓	✗	✗	✓
Anonymity	✓	✓	✗	✗	✓	✓
Reputation computation	✗	✗	✓	✓	✓	✓
Resist to SPOF	✗	✗	✗	✗	✗	✓
Performance evaluation	✗	✓	✗	✗	✓	✓

2.5 Summary

Blockchain technology is gaining significant attention from individuals and organizations of nearly all kinds and dimensions. It is capable of transforming the traditional industry with its features, which include decentralization, anonymity, persistency, and auditability. This chapter highlighted the importance of blockchain technology,

and it's core concepts. We presented a literature review pertinent to our research work dealing with IoT data trading and reputation systems.

Chapter 3

Towards a Trustful Game-Theoretic Mechanism for Data Trading in the Blockchain-IoT Ecosystem

In this chapter, we formulate the data market structure in section 3.1, followed by non-cooperative game theoretic approach for data trading in section 3.2. Section 3.3 discusses the Nash equilibrium solutions. Section 3.4 discusses the mechanism design, while section 3.5 presents experimental results of the model. Finally, conclusion is drawn in section 3.6.

3.1 Data Market Structure

Figure 3.1 shows the high-level architecture of the proposed data market. At a high level, data buyers and a data aggregator (DA) register themselves to the certificate authority (CA) to obtain a legal identity. The CA issues certificates (digital identities contained in X.509 digital certificates) to each entity. In this market, continuous data records are generated through IoT devices and made for sale by data owners (DOs). The latter grant access permission to the DA to aggregate, package, and sell data records on their behalf according to a smart contract-based agreement. The DA informs all the buyers about the packages available for sale through the blockchain network. Buyers simultaneously reply with their bids, which include the

bidding price and required data records from a specific package. Buyers will compete with each other to obtain desired records and learn from the outcome. Afterward, through a trustful auctioning process, data records will be awarded to one winner at a given time. Later, DA can leave a review score for a winner buyer for the current transaction. All the transactions will be added to the ledger. Our auction mechanism (winner determination and payment allocation) and review score are implemented in the form of smart contracts on the blockchain network.

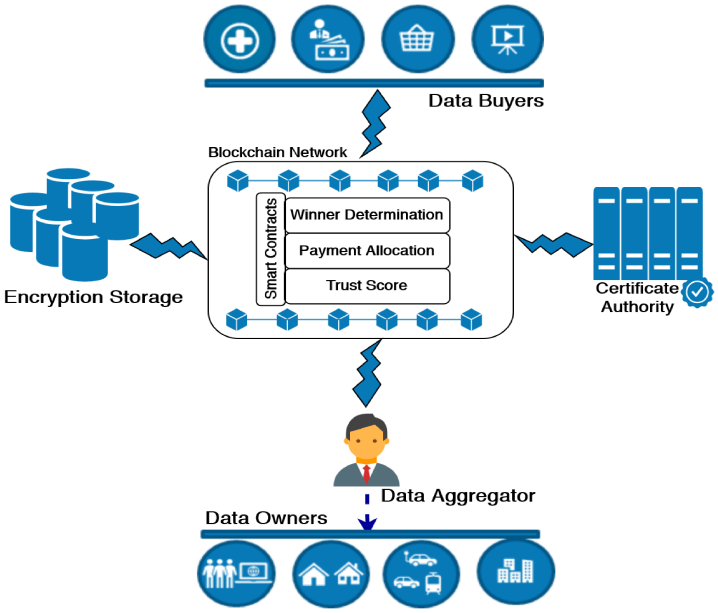


Figure 3.1: High-level architecture of the proposed system.

3.1.1 Assumptions

Before describing the detailed process of the proposed system, the following assumptions are made:

1. Infinite data records: we assume that DO $w \in W = \{1, 2, \ldots, m\}$ produces infinite data records r_w^t from IoT devices, such as wearable devices and smart appliances at time t. This is reasonable since 41.6 billion IoT devices will be connected to Internet by 2025 [2].

2. Data aggregator: we assume that DA is a trusted entity, acting on behalf of DOs, involves in technical and operational tasks, such as deriving data records value based on DOs' privacy risk, data encryption, coding a smart contract, and blockchain operations. This is reasonable because performing these technical tasks for senior citizens equipped with IoT devices would be extremely difficult.

3. Certificate authority: we assume the CA to be fully trusted. This is reasonable since the CA is a government agency responsible for managing the identities and credentials of a data aggregator and buyers.

4. Data buyers: we assume that the data buyers do not share their bidding values among each other, and their behavior is non-cooperative with the goal of maximizing their benefit.

3.1.2 Process details of the proposed system

Figure 3.2 describes the sequence diagram of the interaction between DA and buyers. Once DA and buyers obtained their certificates, the DA creates different packages based on data types received from DOs. For example, a package $\mathcal{D} = \{r_1^t, r_2^t, \ldots, r_m^t\}$ may contain smart TV records or energy records. The DA encrypts and stores data records in a secure indexed database, and generates a decryption key. Once this is completed, the DA sends the index of the records to the blockchain. Afterward, the DA publicizes the packages to the blockchain network. Data buyers are the end-users who purchase the data. Let $B = \{1, 2, \ldots, b\}$ be the set of data buyers in our system. Each data buyer is indexed by $i \in B$. Each data buyer i submits its bid $\beta_i^t = \big(g_i, v_i^t(x)\big)$ to the blockchain network, where g_i and v_i^t are total required quantity and reserved value, respectively. We denote x as the traded amount of records from package $\mathcal{D}$ in the market at time t. In this model, a limited number of records will be traded at each time t, until buyers fulfill their total demand g_i. The traded amount of records x can be defined by the CA or can be based on an agreement between players inside the market. For example, assume that a package consists of 10 million energy records about TV usage. Utility companies (i.e, buyers) are usually interested in different quantities, maybe one company is interested in one thousand records, while another company is interested in one million records, and in each auction period a hundred number of records are going to be sold. Thus, companies are going to keep competing with each other and biding simultaneously at each period (stage) until obtaining the desired quantity. Finally, after receiving the asking price for data records to be traded and bids from buyers, the smart contracts run as follow:

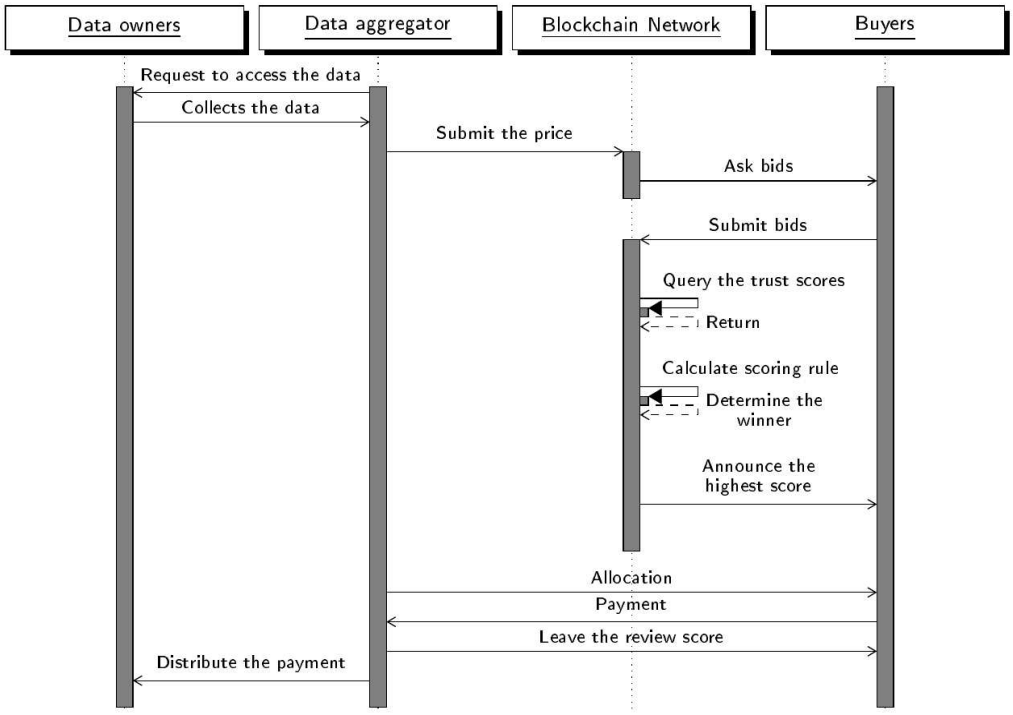


Figure 3.2: Sequence diagram describes the interaction between DA and buyers.

1. In the first sub-stage, the winner determination smart contract retrieves the trust score of buyers from blockchain. The trust score is determined by DA and buyer i's previous trading experiences. Then, the smart contract will remove the bids which are less than the trust threshold. This is done to ensure untrustworthy buyers will not have a chance to get the data. Then, smart contract will run the scoring rule and announce the winner.

2. In the second sub-stage, the payment allocation smart contract will run to determine the payment. In the payment stage, we impose a penalty on the payment of the winning buyer with respect to his/her trust score. The winner i receives data records and a decryption key. Simultaneously, the DA receives a payment amount p_i.

3.1.3 Data Value

The value of data will be derived based on DOs' privacy risk. DOs may have different privacy attitudes, and as a result, they may set different values for their data records. For instance, some DOs may be concerned about their privacy and would allow a user to access a small portion of data in exchange for a few dollars, whereas others may not be concerned about privacy and they ask for a higher price. The DA derives the privacy risk $\Omega_w(r_w^t)$ of a DO w as follows [13]:

$$\Omega_w(r_w^t) = PC(r_w^t) \times SL(r_w^t) \quad \forall PC, SL \in [0,1] \tag{3.1}$$

where $PC(r_w^t)$ denotes privacy concern of DO w, and $SL(r_w^t)$ denotes the sensitivity level of data [13]. The DA derives the privacy risk values $\Omega_{w,q}(r_w^t)$ of each w. Each DO w is described by a privacy risk value $\Omega_w(r_w^t)$ as well as a value of data $\mathcal{V}_w(r_w^t)$. Therefore, there is mapping $\mathcal{Z}$ between the privacy risk and the value of data that $\mathcal{Z} = [\Omega_w(r_w^t) \rightarrow \mathcal{V}_w(r_w^t)]$. The data values may vary for each DO. In order to find the final value of data records for each data type, we calculate the average value of data records as follows:

$$\bar{\mathcal{V}}^t = \left(\frac{\sum_{w=1}^{m} \mathcal{V}_w(r_w^t)}{m} \right) \tag{3.2}$$

where m is the total number of DOs which participate in selling data for a specific data type. Once the data aggregator announces the final value of the data to DOs, they can either accept or reject it $\langle$*Accept, Reject*$\rangle$. If the DO w decides to accept the final value then DA collects the data for further processing.

3.1.4 Data Aggregator Utility

For DA j, we define a cost function $C_j(r_w^t)$ representing the total cost incurring from operation, maintenance and electricity bill for the data records r_w^t at period t. It can be noticed that such cost increases with the size of data records, yielding an increasing and strictly convex cost function. We choose a quadratic function to model the cost function as follows [44]:

$$C_j(r_w^t) = a(r_w^t)^2 + b(r_w^t) + c \tag{3.3}$$

where $a, b, c \geq 0$ are constants. These parameters are dependent on the type of operation, maintenance, and electricity bill incurred to the DA. The utility function of DA is modeled by revenue of selling data records minus the cost:

$$U_j = \sum_{t=1}^{T} R_j(r_w^t) - C_j(r_w^t) \tag{3.4}$$

$$\text{subject to } x_w^t \leq r_w^t \tag{3.5}$$

Eq. (3.5) ensures that the DA trades no more than the agreed upon amount of records. Significant notation is summarized in Table 3.1 for the clarity of readers.

Table 3.1: Notations.

Notation	Description
w	Data owner (DO) $w \in W = \{1, 2, \ldots, m\}$
j	Data aggregator (DA)
r_w^t	Infinite data records r_w^t of Do w at time t
i	Buyer $i \in B = \{1, 2, \ldots, b\}$
x	Trading amount of records x
$\Omega_w(r_w^t)$	Privacy risk
$\mathcal{V}_w(r_w^t)$	Value of data
$\bar{\mathcal{V}}^t$	Average value of data records
$C_j(r_w^t)$	Cost function
U_j	Utility function for the DA
$u_i(\beta_i^t, \beta_{-i}^t)$	Utility function for buyer i for one-shot game
$v_i^t(x(\beta_i^t, \beta_{-i}^t))$	The buyers' valuation functions v_i^t
U_i	Overall utility of buyer i
G	One-shot game
$\mathcal{S}(\beta_i)$	Scoring rule funtion for buyer i bid
G_ψ^T	Finite repeated game with discount factor ψ
$Tr_n^t(j, i)$	Total trust DA has about a given buyer i
$T_{indirect}^{yi}$	Indirect trust
$p_i(\bar{\beta}_i)$	Payment of the winner buyer i

3.2 A Repeated Game Theoretic Approach for Data Trading Between Buyers

In this section, we present a non-cooperative game for data buyers in the infinite repeated horizon. A repeated game is one where the buyers repeatedly play the

same one-shot game in each time period (called a stage game) in which they play simultaneously [45]. We first formulate the utility function for a one-shot game G. Then, using the one-shot game definition as a building block, we then proceed to define finitely G^T and infinitely repeated games G^∞ that capture repeated interactions among the different buyers. We consider a data market setting for one-shot game $G = \langle B, A_i, u_i \rangle$, where B is set of buyers. Each buyer i has an action set A_i. An action profile $\beta = (\beta_i, \beta_{-i})$ consists of the bid of buyer i and bids of other buyers, denoted by $\beta_{-i} = (\beta_1, ..., \beta_{i-1}, \beta_{i+1}, ..., \beta_b) \in A_{-i}$. In addition, each buyer i has a real-valued, one-shot game utility function $u_i : A_i \to \mathbb{R}$, which maps every action profile $\beta \in A$ into a utility for buyer i, where A denotes the cartesian product of the action spaces A_i, written as $A = \prod_{i=1}^{B} A_i$.

3.2.1 Data Buyers Utility

We assume a buyer i, who needs a number of records from package $\mathcal{D}$ of a specific type, knows his own valuation of the current traded amount of records, but not those of his opponents. On receiving the required amount of records, the buyers pay the price $p_i^t\big(x(\beta_i^t, \beta_{-i}^t)\big)$, conditional on winning records, given the other buyers bid β_{-i}^t. If the game G is played only once, the utility function for the buyer i is the difference between valuation for traded amount of records and payment. The utility function u_i of buyer i for one-shot game is:

$$u_i(\beta_i^t, \beta_{-i}^t) = \sum_t v_i^t\big(x(\beta_i^t, \beta_{-i}^t)\big) - p_i^t\big(x(\beta_i^t, \beta_{-i}^t)\big) \tag{3.6}$$

where v_i^t is buyer i's valuation for the trading amount of records x. It represents how much the requested records are worth to the buyer i. The buyer i hopes to pay a smaller price p_i then his estimated value v_i. The buyers' valuation functions v_i^t are drawn independently from the following equation:

$$v_i^t(x(\beta_i^t, \beta_{-i}^t)) = v_i^t\left(x\left(1 + \log\left(\beta_i^t(\varphi), \beta_{-i}^t(\varphi)\right)\right)\right) \tag{3.7}$$

where φ denotes the satisfaction rate of buyer i ($0 \leq \varphi \leq 1$). The log function modifies the buyers' valuation in proportion to their satisfaction rate. This means that if buyer i is not satisfied with the quality of obtained records at stage t, the valuation of the buyer i in next stage decreases as show in in Figure 3.3. We assume

that the satisfaction rate of buyers $i \in B$ is 1 at the beginning. After receiving records at stage t, the buyer measures the quality of the records and updates the satisfaction rate. The buyers use the average of satisfaction rates as their valuation for the next stage. Figure 3.3 describes they way valuations are affected by the satisfaction rates.

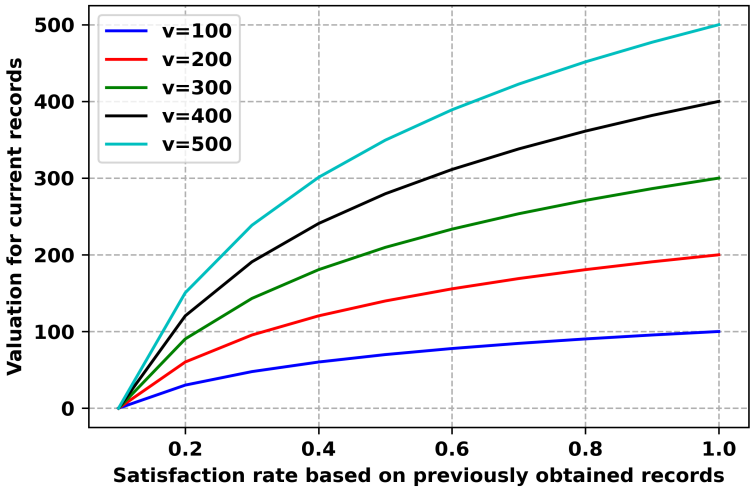


Figure 3.3: Valuation function. This example shows that the valuation v_i of buyers B decreases on the basis of the satisfaction rate φ. Note that this example does not take into account the average of all satisfaction rates.

In the next stage, the game G structure stage does not change. Buyers will continue bidding until they obtain the total quantity they needed. The overall utility of buyer i in the repeated game G_ψ^T is:

$$U_i = (1-\psi)\sum_{t=1}^{T} \psi^{t-1} u_i(\beta_i^t, \beta_{-i}^t) \tag{3.8}$$

where ψ is a discounted factor and β_{-i}^t denotes the set of bids submitted by the buyers other than i at stage t. We assume that future utilities are discounted proportionally at some rate ($0 \leq \psi \leq 1$). $\psi = \frac{1}{1+r}$, where r is the interest rate. We used fictitious play as type of learning for the buyers [46, 47]. Each buyer i starts with some belief about what are the bids of other buyers. Each buyer i updates his/her beliefs based on what he/she observed in the iteration of G_ψ^T. More formally, let $\eta_i^t(\beta_{-i})$ denotes the number of times buyer i has observed (β_{-i}) in the previous stages. So, buyer i assesses other buyers bid using fictitious learning as follow [46, 47]:

$$\sigma_i^t(\beta_i) = \frac{\eta_i^t(\beta_{-i})}{\sum_{\beta - i \in A - i} \eta_i^t(\beta_{-i})} \tag{3.9}$$

where $\sigma_i^t(\beta_i)$ is the probability that is proportional to the time it was played in the past. This means that buyer i forecasts buyer $-i$'s bid at time t to be the empirical frequency distribution of past G. Given buyer i's belief about other buyers play, he/she chooses the bid at time t to maximize his/her utility [46, 47]:

$$\beta_i^t \in \arg\max_{\beta_i \in A_{-i}} (\beta_i, \sigma_i^t) \tag{3.10}$$

3.3 Nash Equilibrium Solutions

The Nash equilibrium (NE) of a game is an action profile (list of actions — one for each buyer) with the property that no player can increase his utility to achieve higher benefits by choosing a different action given the other buyers' actions. To maximize the utilities, the buyers adjust their bids to reach the equilibrium. This means that if a NE exists for the game, then all buyers $i \in B$ are expected to converge to the state represented by the equilibrium. So, each buyer i aims to choose the strategy or action that maximizes its utility function to determine the best outcome. In addition, the players in the one-shot game choose their own bids independently and simultaneously and try to maximize their expected utility. There are two types of strategies or actions available for players: pure strategies and mixed strategies. Pure strategy defines an action that a player wants to take with positive probability from a given set of strategies in the game. In contrast, a mixed strategy for a player is a probability distribution over his/her pure-strategy choices. In our model, we will prove that the pure strategy equilibrium exists for the proposed one-shot game. The objective function of the players is to maximize their utilities. Before finding the NE of our one-shot game G, we first define formally the best response and NE. For the sake of clarity, we are dropping t notations, referring to the time, since we are dealing with a one-shot game.

Definition 1 (Best response [48]). *Assuming all the buyers $i \in B$ are rational, a buyer i played his/her bid $\left(\beta_i^*\right)$ as best response to the other buyers' β_{-i} played action $\left(\beta_{-i}^*\right)$ such that:*

$$\beta_i^* \in BR\left(\beta_{-i}\right) \text{ iff } \forall \beta_i \in A_i, u_i\left(\beta_i^*, \beta_{-i}\right) \geq u_i\left(\beta_i, \beta_{-i}\right) \tag{3.11}$$

Definition 2 (Nash equilibrium [48]). *The NE is a profile of actions, one for each buyer, such that each action is the best response to the other buyers actions. Specifically, an action profile β is said to be NE, if:*

$$\beta^* = \langle \beta_1^*, \beta_2^*, ..., \beta_n^* \rangle \text{ is a NE iff } \forall i,\ \beta_i^* \in BR\left(\beta_{-i}^*\right) \tag{3.12}$$

We will first define the existence of NE for the finite repeated game, which can be viewed as a generalization of the equilibrium concept for the one-shot game. We should point out here that we won't be able to construct subgame perfect nash equilibrium (SPNE) i.e., induced normal form - backward induction, which is the standard solution for finding NE. SPNE works only when the utility function is continuous and only applies to finite games. However, in our model, the utility function of buyers in (3.6) introduces a discontinuity in utilities. This means that the u_i could be zero at some stage t for buyer i, or it could have non-zero value. Hence, we will use the following approach to finding the NE in every one-shot game with a discontinuous utility setting. Next, we will leverage the results by using the Folk theorem in the infinitely horizon setup to find NE.

Theorem 1. *A Nash equilibrium exists in the proposed non-cooperative game $G = \langle B, A_i, u_i \rangle$.*

Proof. The Nash equilibrium exists only when the following conditions are satisfied [49]:

1. $A_i \subseteq \mathbb{R}^m, (i = 1, ..., b)$ is a non-empty, compact and convex subset of Euclidean space.
2. $u_i = A_i \to \mathbb{R}^b$ is upper semi-continuous in β and quasi-concave in $\beta_i\ \forall i$.

Obviously, the first condition can be satisfied since A_i is defined by a set of bidding vectors in which all the values are between zero and the maximum bidding of buyers. So, it is a nonempty, compact and convex subset of the Euclidean space R^b. To show that $u_i = A_i \to \mathbb{R}^b$ is upper semi-continuous, we first define the following property [50]:

Definition 3. *$u_i(\beta_i, \beta_{-i})$ is upper semi-continuous at β_{i0} if $\exists \beta_i$ as a neighborhood such that:*

$$\lim_{\beta_i \to \beta_{i0}} \sup u_i(\beta_i, \beta_{-i}) \leq u_i(\beta_{i0}, \beta_{-i}) \tag{3.13}$$

For a jump point of u_i in a given range $\Delta\beta$, we define [50]: $\beta_{i0} = \beta_i + \Delta\beta$ such that $p_1 \leq p_2$, $p_t(\beta_i, \beta_{-i}) = p_1$, and $p_t(\beta_{i0}, \beta_{-i}) = p_2$. This means that u_i function is upper semi-continuous because rational buyers $i \in B$ attempt for a higher utility around the discontinuity point. Only the quasi-concave property remains to be proved. Taking the derivatives of (3.6) with respect to β_i, we get:

$$\frac{\partial u_i}{\partial \beta_i} = \frac{v\big(x(\beta_{-i})\big)}{ln(10)\beta_i} - 1 \tag{3.14}$$

$$\frac{\partial^2 u_i}{\partial \beta_i^2} = -\frac{v\big(x(\beta_{-i})\big)}{ln(10)\beta_i^2} \tag{3.15}$$

Since $\frac{\partial u_i}{\partial \beta_i} = \frac{v\left(x(\beta_{-i})\right)}{ln(10)\beta_i} - 1 > 0$ and $\frac{\partial^2 u_i}{\partial \beta_i^2} = -\frac{v\left(x(\beta_{-i})\right)}{ln(10)\beta_i^2} < 0$, the utility function u_i is concave with respect to β_i, hence it is quasi-concave in β_i [44], thus we get:

$$u_i((1-\lambda)\beta_i^x + \lambda\beta_i^y, \beta_{-i}) \geq \min\{u_i(\beta_i^x), u_i(\beta_i^y), \beta_{-i}\} \tag{3.16}$$

where β_i^x and β_i^y belong to the buyer i action set A_i. Therefore, u_i is a quasi-concave in β_i $\forall i$. Thus, we have proved the existence of the NE. □

Theorem 2. *The NE of game $G = \langle B, A_i, u_i \rangle$ is unique.*

Proof. The uniqueness proof is to show that the best response function of each buyer β_i^* is a standard function. Based on best response Definition 3.11 and using Eq. (3.14), the best-response is achieved when the first derivative of u_i is equal to 0, thus we have:

$$\frac{\partial u_i}{\partial \beta_i} = \frac{v\big(x(\beta_{-i})\big)}{ln(10)\beta_i} - 1 = 0 \tag{3.17}$$

and we obtain:

$$\beta_i^* = f(\beta) = \frac{v\big(x(\beta_{-i})\big)}{ln(10)} \tag{3.18}$$

A function $f(\beta)$ is a standard function [51], if the following properties are satisfied:

1. Positivity: $f(\beta) \geq 0$;
2. Monotonicity: For all β and $\hat{\beta}$, if $\beta \geq \hat{\beta}$, then $f(\beta) \geq f(\hat{\beta})$;
3. Scalability: For all $\mu > 1, \mu f(\beta) \geq f(\mu\beta)$;

$f(\beta)$ satisfies the three above properties of a standard function.
Positivity: The best-response function in (19) is always positive, so $f(\beta) \geq 0$ positivity property is set up.
Monotonicity: Assuming $\beta \geq \hat{\beta}$, then

$$f(\beta) - f(\hat{\beta}) = \frac{v\big(x(\beta - \hat{\beta})\big)}{ln(10)} \geq 0 \tag{3.19}$$

we have $f(\beta) - f(\hat{\beta}) \geq 0$, in which $f(\beta)$ is monotonically increasing function.
Scalability: For all $\mu > 1$ we have,

$$f(\beta) = \mu \frac{v\big(x(\beta_{-i})\big)}{ln(10)} \qquad\qquad f(\beta) = \frac{\mu v\big(x(\beta_{-i})\big)}{ln(10)} \tag{3.20}$$

So, for all $\mu > 1, \mu f(\beta) \geq f(\mu\beta)$ thus, scalability property holds. Therefore, there exists a unique NE in the above one-shot game G, which can be viewed as the finite repeated game G_{ψ}^{T}. □

If the stage-game of a finitely repeated game has a unique NE, then we can consider that constant action for each buyer i, always play the stage-game best response irrespective of the past history. The infinitely repeated games requires different setup than finitely repeated games since it dose not have a terminal point. Before finding the NE of infinitely repeated game G_{ψ}^{∞}, we need to formally define the minmax value, enforceable and feasible utility as follows:

Definition 4 (Minmax value [48]). *Considering stage-game $G = \langle B, A_i, u_i \rangle$, the minmax value ν_i for each buyer i is:*

$$\nu_i = \min_{\beta_{-i}} \max_{\beta_i} u_i(\beta_i, \beta_{-i}) \tag{3.21}$$

It represents the amount of utility buyer i receives when the other buyers play minmax strategies and buyer i plays the best response.

Definition 5 (Feasible [48]). *Given a set of utility vector $U = (u_1, u_2, ..., u_n)$, U is said to be feasible if the convex hull of U is expressed as:*

$$\mathcal{H} = Conv\{u \in \mathbb{R}_+ \mid \exists \beta \in \mathbb{R}_+, U_i = \sum_{t=1}^{\infty} u_i\} \tag{3.22}$$

First, we need to apply Definition 5 to the set of utilities in a stage game G. Then, the convex hull of U will be determined by the convex combination between all utility vectors. Note that convex hull $\mathcal{H}$ of the vector utilities is achievable with pure strategies. In other words, a utility profile is feasible if it is a convex, and convex combination of the outcomes in G.

Definition 6 (Enforceable [48])**.** *A utility vector U is said to be enforceable, if:*

$$\mathcal{U} = \{u_i \geq \nu_i, \forall i \in B\} \tag{3.23}$$

The set of feasible and enforceable utilities is $\mathcal{E} = \mathcal{H} \cap \mathcal{U}$. Therefore, any set of feasible and enforceable utilities in $\mathcal{E}_\infty$ (Infinitely repeated game), $\mathcal{E}_T$ (Finitely repeated game), and $\mathcal{E}_\psi$ (Discounted repeated game) are always included in $\mathcal{E}$.

Theorem 3. *A Nash equilibrium exists in infinitely repeated game $G_\psi^\infty = \langle B, A_i, U_i \rangle$, if U is enforceable and feasible in $\mathcal{E}$, such that for each buyer i, we have $u_i \geq \nu_i$ [52]. Then $\mathcal{E}_\psi \xrightarrow[\psi \to 0]{} \mathcal{E}$.*

Proof. According to [53], there exists NE in discounted infinitely repeated game. There can be many NE in the infinitely repeated games G^∞ even if the stage game only has a unique NE. □

3.4 Mechanism Design

In this section, we design a truthful mechanism to determine the winner allocation and corresponding payment for the proposed one-shot game G (possibly $G \to G^T$) and implement it in form of smart contracts. The process of developing a mechanism design faces two primary challenges. One is how to determine the winner buyer and allocation. The other is how much the winner buyer should pay for the records. This section addresses these two issues by using a scoring function based on the trust score to evaluate the buyer's bid and announce the winner. Furthermore, we consider a penalty for the winner if he/she is not fully trusted with respect to his/her trust score. Given the reservation price $p_j(x)$ and the submitted bids, the smart contracts will return the winner allocation and payment rules.

3.4.1 Winner allocation stage

In winner allocation stage, each data buyer $i \in B$ submits his/her bid $\beta_i = \big(g_i, v_i(x)\big)$ simultaneously to the blockchain at stage t. The valuation $v_i(x)$ for the traded amount of records offered in all stages is unknown to the DA. In this model the winner allocation stage includes two steps. In the first step, after receiving bids from buyers, the smart contract collects the trust score of buyers who participated in the bidding process and eliminates buyers whose trust score is less than a threshold $\mathcal{T}$. The $\mathcal{T}$ is determined based on the average of data sensitivity $SL(r_w^t)\ \forall w \in W$, which is obtained through Eq. (3.1). In the second step, the scoring function $\mathcal{S}(\beta_i)$ is calculated for each buyer $i \in B$ according to the following scoring rule:

$$\mathcal{S}(\beta_i) = \beta_i \times Tr_n^t(j,i) \tag{3.24}$$

$$\text{subject to } \beta_i \geq p_i \tag{3.25}$$

where $Tr_n^t(j,i)$ measures the total trust DA has about a given buyer i, which is computed using the current satisfaction Tr_c and previous trust score $Tr_{n-1}^t(j,i)$ as shown in Eq. (5.10). In case DA does not have a prior trust for buyer i, we take indirect trust $T_{indirect}^{yi}$ into account. The total trust function is defined as follows:

$$Tr_n^t(j,i) = \begin{cases} \alpha \times Tr_c + (1-\alpha) \times Tr_{n-1}^t(j,i), \text{ if } Tr(.) > 0 \\ \alpha \times Tr_c + (1-\alpha) \times T_{indirect}^{yi}, \text{ if } Tr(.) = 0 \end{cases} \tag{3.26}$$

Here α is a relative weight that changes based on the accumulated deviation defined in Eq. (3.30, 3.31) and (3.32). The Tr_c function measures how much DA j is satisfied about data buyer i. It represents the satisfaction score for the most recent transaction between j and i $(0 \leq Tr_c \leq 1)$.

$$Tr_c = Sat_c \times \frac{(1 - e^{-\lambda Val_n^v(j,i)})}{1 + \xi_n^v(j,i)} \tag{3.27}$$

Here Sat_c is a feedback-based factor (e.g., review score) for the current transaction n reflecting the way DA j rates data buyer i [54].

$$Sat_c = \begin{cases} 0, & \text{if } j \text{ is totally unstatisied with } i, \\ 1, & \text{if } j \text{ is totally statisied with } i, \\ \in (0,1), & \text{otherwise.} \end{cases}$$

$Val_n^v(j,i)$ is a recent value fluctuation between the previous and current value Val_c. Here, λ is the decay constant and it controls the trust value. The Tr_c value reaches 1.0 with larger $Val_n^v(j,i)$ and decreases slowly with smaller $Val_n^v(j,i)$. For example, if the transaction's value is insignificant and current satisfaction is high, this will have little effect on overall trust. On the other hand, if the value of the transaction is high, and current satisfaction is high, the overall trust will be increased significantly.

$$Val_n^v(j,i) = |Val_{n-1}^v(j,i) - Val_c| \tag{3.28}$$

$$\xi_n^v(j,i) = \mathcal{K} \times Val_n^v(j,i) + (1-\mathcal{K}) \times \xi_{n-1}^v(j,i) \tag{3.29}$$

Here $\xi_n^v(j,i)$ represents the accumulated value deviation for the history of all transactions. The α is relative weight which gives higher weight to the recent n [54]. The weight of α changes based on the accumulated deviation $\xi_t^n(j,i)$ [54].

$$\alpha = threshold + \mathcal{K} \times \frac{\delta_n^t(j,i)}{1+\xi_n^t(j,i)} \tag{3.30}$$

$$\delta_n^t(j,i) = |Tr_{n-1}^t(j,i) - Tr_c| \tag{3.31}$$

$$\xi_n^t(j,i) = \mathcal{K} \times \delta_n^t(j,i) + (1-\mathcal{K}) \times \xi_n^t(j,i) \tag{3.32}$$

Here $\mathcal{K}$ is some user-defined constant factor which controls to what extent we will react to the recent error $\delta_n^t(j,i)$ [54]. So, if we increase the value of $\mathcal{K}$, then we give more significance to the recent deviation than accumulated deviation [54]. The *threshold* is used to prevent α from saturating to a constant value. $T_{indirect}^{yi}$ value is computed when DA j does not have a prior trust relationship and experience with buyer i. The DA requests other entities' $y \in Y$ to provide their rating about the target buyer i. So, the DA will have the capability and experience to truthfully judge the data buyer for the first transaction. The indirect trust function is:

$$T_{indirect}^{yi} = \sum_{y=1}^{Y} \frac{\mathcal{P}_{yi}}{\mathcal{P}_{yi} + \mathcal{N}_{yi}} \tag{3.33}$$

where, $\mathcal{P}_{yi}$ denotes positive feedback of entity y ($0.5 \leq \mathcal{P}_{yi} < 1$), and $\mathcal{N}_{yi}$ termed as negative feedback ($0 \leq \mathcal{N}_{yi} \leq 0.5$). So, $T^{yi}_{indirect}$ represents the total number of positive and negative feedbacks for buyer i. Based on Eq. (3.24), the buyer with the highest score wins the game at stage t. If the buyers have an equally high trust score, we randomly selected the winner. After that, allocation rule will apply $\mathcal{X} : \mathbb{R}_+ \rightarrow [0, 1]$, meaning that with the score bid profile $\mathcal{S}(\beta_i)$, buyer i gets the records with probability $\mathcal{X}\big(\mathcal{S}(\beta_i)\big)$ and makes a payment of $p_i(\beta) \in \mathbb{R}$, which indicates the amount that buyer i must pay. Furthermore, allocation rules have to satisfy the feasibility constraint as follows:

$$\sum_{i \in B}^{t} \mathcal{X}\big(\mathcal{S}(\beta_i)\big) \leq x \;\; \forall \beta \tag{3.34}$$

Eq. (3.34) restricts the allocation of records for the winner not to be more than the traded amount at stage t. Other buyers will modify their bids accordingly for the next stage of the game. Buyer $i \in B$ will continue bidding until obtaining the total quantity he/she requested.

$$g_i \geq \sum_{t}^{\infty} x \tag{3.35}$$

3.4.2 Payment Stage

In the payment stage, we consider a penalty for the winner if he/she is not fully trusted i.e., a trust score less than 1. If the winner is fully trusted which implies ($Tr^t_n(j, i) = 1$), he/she will not be punished. The trust is calculated based on Eq. (5.10). The payment rule of winning buyer $\bar{\beta}_i$ is:

$$p_i(\bar{\beta}_i) = \bar{\beta}_i - u_i\Big(1 + \log\big(Tr^t_n(j, i)\big)\Big) \tag{3.36}$$

The above equation provides assurances that buyer i is punished only on its stage utility u_i and will not be charged more than its bid. Algorithm 1 describes the winner allocation as well as the payment stage.

Algorithm 1 Winner allocation and payment stages

Input: Submitted bids β_i and reservation price p_j for stage t

Output: Winner allocation $\mathcal{X}\big(\mathcal{S}(\bar{\beta}_i)\big)$, payment $p_i(\bar{\beta}_i)$, current satisfaction score Sat_c

Stage 1: Winner allocation stage

for each $\beta_i(g_i, v_i(x))$ and p_j **do**
    Calculate the $Tr_n^t(j,i)$
    if $Tr_n^t(j,i) \geq \mathcal{T}$ **then**
        Calculate the $\mathcal{S}(\beta_i) = \beta_i \times Tr_n^t$
    else
        Remove β_i Eliminating bids less than thershold $\mathcal{T}$
    end if
end for
$\mathcal{S}(\bar{\beta}_i) = \max\big(\mathcal{S}(\beta_i), i = 1 \text{ to } b\big)$
$\mathcal{X}\big(\mathcal{S}(\bar{\beta}_i)\big) = 1$ Allocation for the highest score
$\mathcal{X}\big(\mathcal{S}(\beta_i)\big) = 0, i = 1 \text{ to } b$

Stage 2: Payment stage

Calculate the stage utility u_i for winner i
$p_i(\bar{\beta}_i) \leftarrow \bar{\beta}_i - u_i\Big(1 + \log\big(Tr_n^t(j,i)\big)\Big)$
Leave current satisfaction score (Sat_c) for winner i

Return $\Big(\mathcal{X}\big(\mathcal{S}(\bar{\beta}_i)\big), p_i(\bar{\beta}_i), Sat_c\Big)$

3.5 Evaluation of Results

In this section, we evaluate the model and analyze the results using different properties such as computational efficiency, bidding learning process, truthfulness, individual rationality, and budget balance. In our experiments, all the results were conducted using a Windows 10, 3.78 GHz Intel Core 7 with 6 GB RAM. For the evaluation of the model, since the record price is decided by the number of DOs $w \in W$ based on their privacy risk, we choose reasonable values for our experiments. We assigned a privacy risk value that is uniformly distributed between 0 and 1 to reflect the privacy attitude of the different DOs. Then, we calculate the average value of the data. We assumed that the data records cost varies between 0 and 1, which

is incurred by DA. We vary the number of buyers for evaluating the performance of our proposed data market.

3.5.1 Computational Efficiency Analysis

We evaluate the property of computational efficiency property, which means that winning determination and the payment stages in Algorithm 1 must be solved within a polynomial time. The computation complexity of the Algorithm 1 is $O(n)$, where n is the number of bidders. In the winning allocation stage, the for loop runs for all submitted bids and then calculates the trust score and scoring rule. The computational complexity of the for loop takes $O(n)$. The max operation will take $O(n)$, and allocations will take $O(1)$. In the payment stage, each statement will take $O(1)$ to finish. Therefore, the computational complexity of the proposed system is bounded by $O(n)$ time complexity at the most. We select 10, 50, 100, 150, 200, 250 and 300 buyers for the experiment, respectively. Figure 3.4 shows the running time of Algorithm 1 under various numbers of buyers. These results indicate that Algorithm 1 completes the computation in almost linear time, as demonstrated in the time complexity analysis given above. Therefore the computational efficiency property is satisfied.

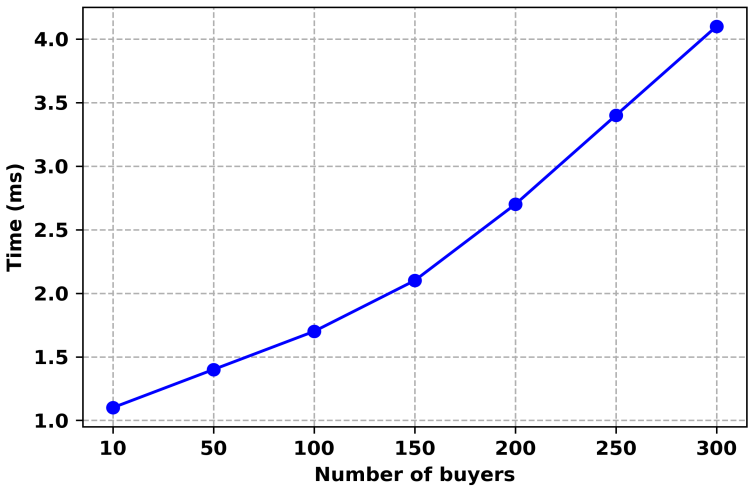


Figure 3.4: Computationally efficient property.

3.5.2 Budget Balance Analysis

We verify the budget balance property. Budget balance means at each stage t, the buyer's payment is higher than the reservation price (asked-price) of the data aggregator. As shown in the payment stage described in Algorithm 1, the buying price for the winner, taking into account the penalty, is greater than the selling price $\beta_i \geq p_i$. To verify the property of budget balance, we repeatedly run the game until stage 20 as shown in Figure 3.5. We can see that the curve line representing the buyer's payment is higher than of the asking price. Since we are imposing a penalty in the payment stage, definitely we satisfy the property of budget balance.

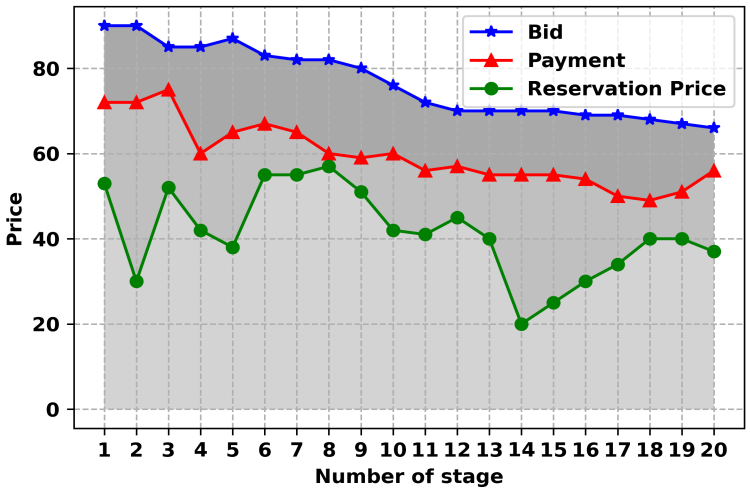


Figure 3.5: Budget balance property.

3.5.3 Individual Rationality Analysis

We evaluate the property of individual rationality in which each winning buyer i must receive a non-negative stage utility $u_i \geq 0$. Similarly, DA utility must be non-negative $U_j \geq 0$. The stage utility is $u_i = 0$ for buyers who are not selected in the winning determination stage. Even by imposing a penalty on the winning buyer in the payment stage, the winning buyer has non-negative u_i. For example, in the worst-case scenario, let's assume that the buyer trust score is 0.1. The payment would be the same as the bidding price. So, the buyer's utility is non-negative. Figure 3.6 shows the average utility for the buyers and DA. We can observe that the winning buyer receives non-negative utility considering penalty in each stage. It can

also be observed that DA has a non-negative utility. From the above, we can verify the individual rationality property of our proposed system.

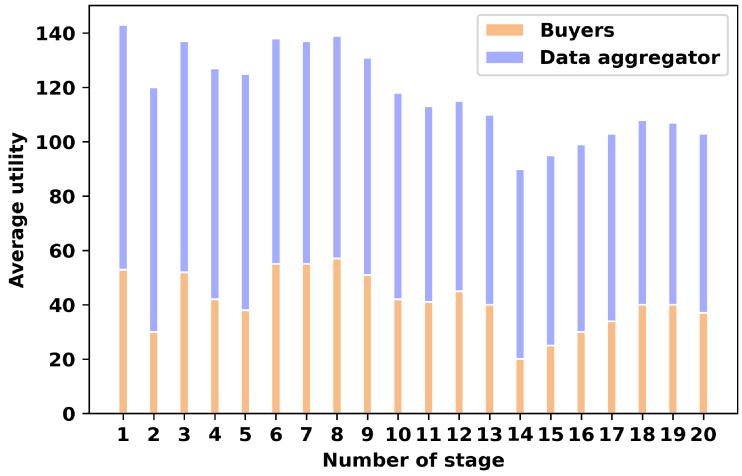


Figure 3.6: Individual rationality property.

3.5.4 Bidding Learning Analysis

As mentioned in section 3.2, we used fictitious play for the bidding learning process. After some arbitrary initial bidding at the first stage of the game, the buyers myopically choose their best responses against the empirical action distribution of other buyers' bids at every subsequent stage. Buyers hope that such a bidding learning process will converge and lead to a NE to increase their utility. Figure 3.7 shows the bidding learning process among 10, 100, and 500 buyers without considering trust. After each stage t, buyers update their actions (bids) based on the outcome of previous stages and observation of other bidders. We can notice that by increasing the number of buyers, the expected utilities increase as well. This will lead to the point where buyers can learn bidding strategies swiftly and converge to a NE to maximize their expected utilities. From Figure 3.7, we can see that buyers are learning from the outcome and observation of other buyers in the previous stage and are increasing their stage utility. Furthermore, by increasing the number of buyers, we can see that buyers converge to their NE profile.

We conducted a box-plot presentation to show the learning process of different data buyers using fictitious play. Figure 3.8 shows the utility with 10, 100, and 500

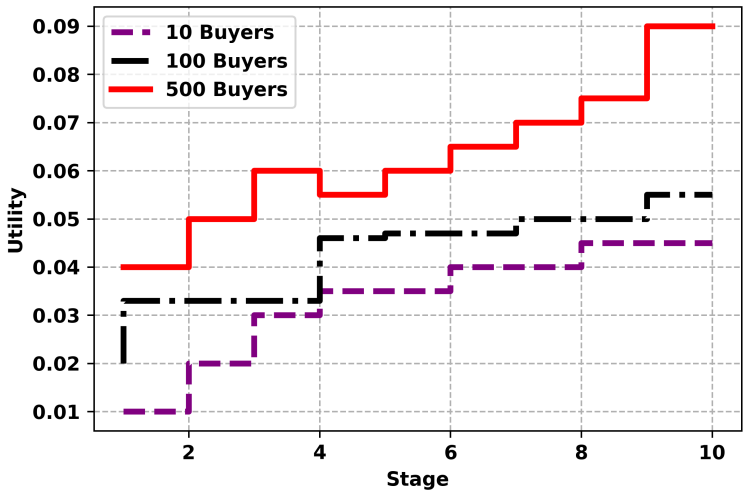
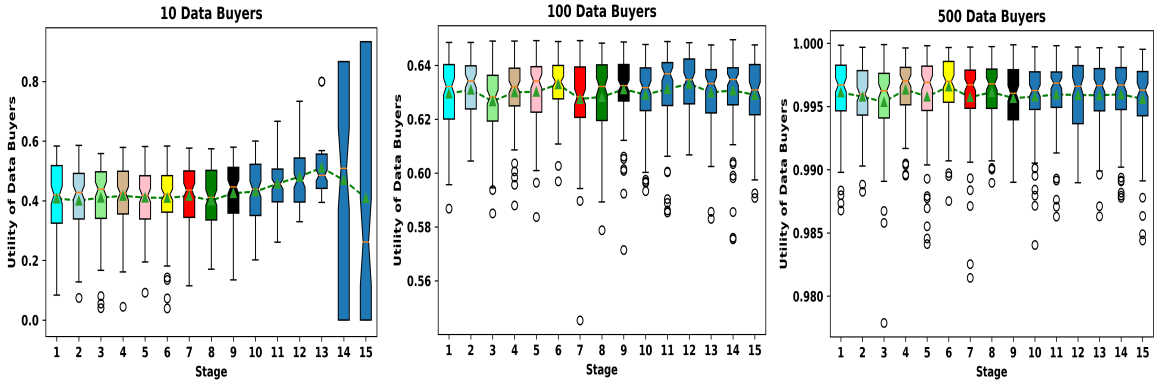


Figure 3.7: Learning process.

data buyers competing with each other repeatedly. For each box plot, the central mark indicates a median of utilities. The upper whiskers show the highest utility and the lowest whiskers show the lowest utility at each stage. The outliers are indicated by a (∘) symbol.



(a) (b) (c)

Figure 3.8: Boxplot presenting the bid learning processing using fictitious play.

3.5.5 Truthfulness Analysis

We evaluate the property of truthfulness in which each winning buyer i must bid their true valuation. Let's assume the following two cases:

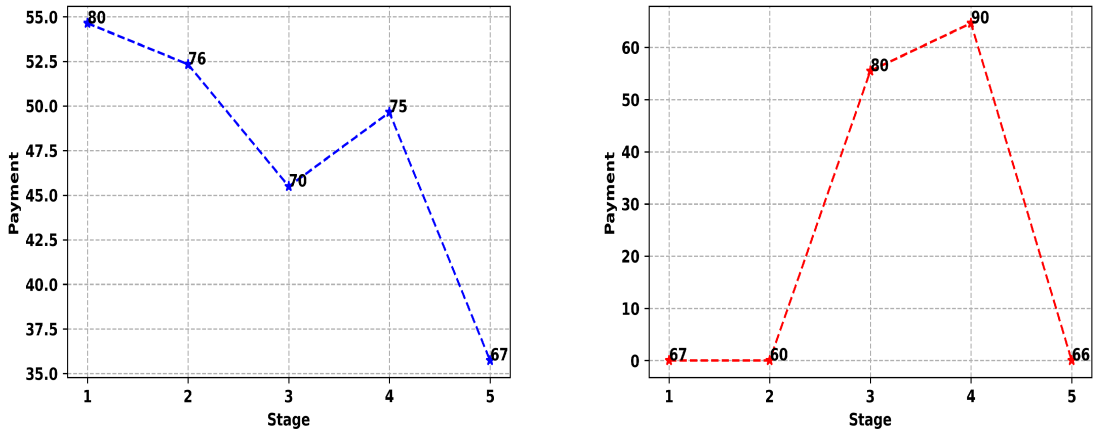
1. We define $\tilde{\beta}_i$ as overbid from the valuation, and β_i^* as the best response, and $\tilde{u_i}$ and $\bar{u_i}$ as their stage utilities, respectively. Buyer i is the winner when submitting either $\tilde{\beta}_i$ or β_i^* at stage t. However, overbidding creates extreme penalties with respect to trust scores in the payment stage in our model, which leads the winning buyer to pay more and gain less utility than if he/she plays their best response. The best response scheme given by Eq. 3.11 is incentive-compatible, when buyer i is repeatedly best-responding, in which case other buyers are incentivized to do the same to maximize their utilities. In other words, a buyer $i \in B$ cannot increase its utility by overbidding, since he/she will be punished severely.

2. Buyer i will lose the game G if he/she bids lower than their valuation (under-bidding), otherwise would win if he/she reported the true valuation and played their best response.

Therefore, buyer i cannot increase its utility by providing untruthful bidding (overbidding and under-bidding), no matter what the other buyers' bid. For the experiment, we select two buyers; the first buyer bids truthfully while the other bids untruthfully. To provide a consistent environment for comparison, we set the trust score for both buyers to 0.7 and ask price to 50, 48, 41, 40, and 30, respectively. Figure 3.9 (a) is the result when winner buyer i is biding his/her true valuation and pays the price p_i at stage t. Figure 3.9 (b) is the result when buyer i bids untruthfully. We can see that buyer i receives zero payment when he/she is underbidding, which means that he/she receives zero utility. At stages 3 and 4, he/she is overbidding, and he/she is the winner. However, we can see that the winner is paying much more than if he bids truthfully. Truthfulness property provides the best possible utility for the buyers and ensures there is no incentive for a buyer to bid untruthfully.

3.6 Summary

While the economic value of IoT data is increasing, it is not very well known how these data can be conceptualized, measured, and monetized in IoT data markets that



(a) (b)

Figure 3.9: Truthfulness property.

enable data owners to trade their data. Unfortunately, the existing IoT data markets are insufficient for capitalizing on the full value of the data in a trusted and transparent way. To address these challenges, we proposed a trustful data trading framework using the game theory approach in an infinitely repeated horizon to enable secure and efficient data trading between buyers and sellers. To model the data market, this chapter proposed a non-cooperative infinity repeated game model between rational data buyers. In each stage of the game, buyers hold a bid for a traded amount of records and seek to maximize their expected utility through learning from the outcome of previous stages considering discounted rates for the future utility. We proved NE and the uniqueness of our model, which is derived theoretically for the one-shot game, finite, and infinite horizon games, respectively. Besides, this model imposes a penalty on those buyers who do not have a good reputation and decreases their chance of winning to preserve the data owner's privacy. Through theoretical and security analysis, we showed that the proposed system is computationally efficient, truthful, budget balance, and individually rational.

Chapter 4

Performance Analysis of Blockchain-based IoT Data Trading Systems

In this chapter, we discuss the existing challenges of blockchain-based data trading systems in section 4.1. Section 4.2 discusses the data trading model based on blockchain. In section 4.3, the implementation setup using Hyperledger Fabric is presented followed by the system performance analysis and results in section 4.4. Finally, conclusion is discussed in section 4.5.

4.1 Introduction

According to a report from MarketsandMarkets [14], the data market will grow to 229.4 billion dollars by 2025. With massive number of IoT devices and applications, the total amount of data created by IoT devices will reach 847 ZB per year by 2021 [10]. Indeed, data are becoming the most valuable asset in use as well as in trade. Such valuable data received considerable attention from organizations to tailor their services to potential consumers and make a profit. As a result, various IoT applications and devices such as smartphones, wearable devices have brought great convenience to people, while producing huge amounts of data. Data have become a valuable asset, which creates a new business called data trading. Currently, there are several data trading systems, e.g., CitizenMe, Datacoup, DataExchange, Factual, and Terbine, to name a few. These systems suffer from two main concerns. First,

they fail to clearly explain where or with whom the owner's data are being shared, and often data change ownership illegally [6]. Second, aggregating large-scale data in storage platforms is subject to cyber attacks [55]. To address these issues, researchers propose blockchain systems as a means of providing people the ability to track and control their data securely [56, 57, 58]. In blockchain systems, the transactions are secure, tamper proof as well as transparent to all data owners and brokers [6]. Furthermore, several studies propose data encryption methods to reduce the risk of data breaches [59, 60, 61].

While all of the above-mentioned studies consider blockchain and encryption methods to achieve private and secure data trading, researchers paid little attention to the performance measures of these systems. This research aims at evaluating the performance of the blockchain-based data trading systems based on throughput (i.e., number of completed transactions per second in blockchain), latency, elapsed time (amount of time needed to query the assets), and resource consumption analysis. Transactions are an important part of the blockchain. To find out how well the system is adding the number of confirmed transactions to the blockchain network, we need to measure and analyze the throughput metric. By default, the underlying data structure of a blockchain does not support an effective method of querying the stored data. To overcome this limitation, we modeled the data in JSON format through CouchDB. The latter supports deploying indexes within the smart contract to make queries more efficient in massive datasets. Indexes enable a database to be queried faster and more efficiently compared to regular queries without indexes. Applying blockchain in data trading systems is not a straightforward task due to high resource consumption. Hence, measuring resource consumption is also important for the efficient management of systems, such as CPU, memory, etc., as well as the successful execution of transactions in the blockchain. Therefore, this research demonstrates a scenario of a blockchain-based data trading system based on Hyperledger Fabric and analyzes its performance measures using Hyperledger Caliper. The main contributions of this chapter are as follows:

- We analyze and evaluate the performance of the blockchain-based data trading system using Hyperledger. Our work measures and analyzes transaction throughput, latency, elapsed time, and resource consumption (memory consumption, CPU utilization, and disc read/write operations) unlike existing research.
- We used a real dataset to emulate a real-life data trading scenario and show that the proposed system can be easily deployed on IoT devices at a low cost.

4.2 Data trading model based on blockchain

Figure 4.1 illustrates an example of a blockchain-based data trading system, in which the market includes data sellers, auctioneers, and data buyers. The three stakeholders continuously interact with each other while trading data. In the data market, the sellers (data owners) generate data from various sources, such as smart devices and social networks, and are willing to sell their data in exchange for money. A seller $j \in \{1, 2, \ldots, N\}$ encrypts and stores its data in a secure indexed database. Since blockchain is suitable for storing small quantities of data, we considered such a secure database for each seller to store their large volume of data. Next, data seller j creates the smart contract $\mathcal{SC}_j$ to specify access control policies for the data and the way the buyer can search the index. Once this is completed, the data seller sends both the smart contract $\mathcal{SC}_j$ and the index to the blockchain. Next, seller j submits the asking price of their data records to the auctioneer. The auctioneer administers the auctioning process to avoid unfair trading, matches between the data sellers and the data buyers, and determines the winner and the payment. The auctioneer may execute an auction mechanism such as a sealed bid auction, double auction to complete the trading process. It should be noted that this study does not focus on auction mechanisms, but rather presents a blockchain-based trading scenario to analyze its performance as indicated previously.

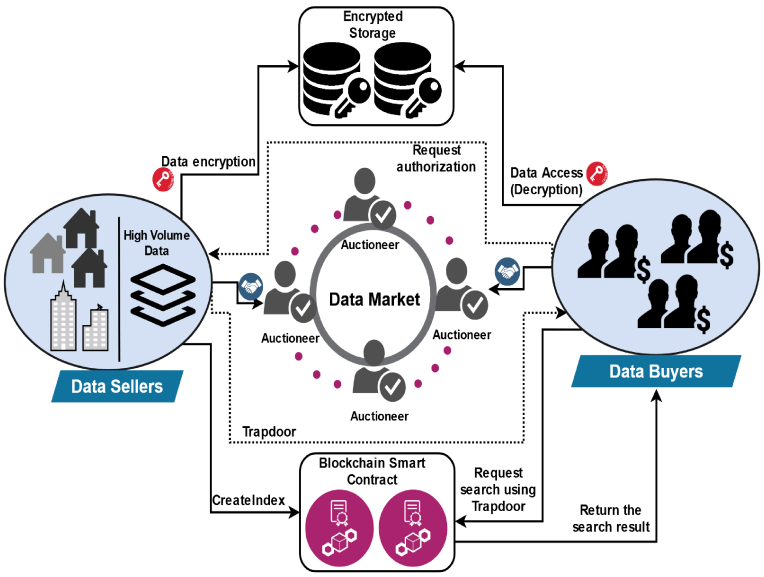


Figure 4.1: An example of blockchain-based data trading system.

Each buyer $i \in \{1, 2, \ldots, B\}$ submits its bid to the auctioneer who tries to match buyers and sellers. If buyer i is declared a winner, it receives the search token and decryption key from seller j to obtain the required data. Then, seller j receives the agreed upon payment amount. The winning buyer i will utilize the token to request a search through the blockchain. The smart contract, which is stored in the blockchain, will use the token and return the search result to buyer i. A smart contract specifies rules and conditions in a digital format, containing protocols within which the participants perform on these promises. A smart contract is a script (e.g., Solidity in Ethereum and Golang in Fabric) which defines the set of assets available to transfer and the type of transactions permitted. All smart contracts have a unique address and are stored in the blockchain. The model assumes that the data sellers have the skill and ability to perform these operations on the blockchain network. More formally, blockchain-based data trading systems is composed of following polynomial algorithms [62, 63]:

1. **KeyGenSetup**$(1^\lambda) \longrightarrow \mathcal{SK}$: It is run by seller j and takes security parameter λ. It outputs the encryption key $\mathcal{SK}_j$.

2. **Encrypt**$(\mathcal{SK}, \mathcal{D}) \longrightarrow \mathcal{C}$: It is run by the seller j to encrypt data. Given the encryption key of seller $\mathcal{SK}_j$, and data set $\mathcal{D}$, it outputs a ciphertext data set $\mathcal{C}$.

3. **DataPolicyRules**$([r_1, r_2, ..., r_n]) \longrightarrow \mathcal{P}$: It is run by the seller j to define data access control policy rules. Given the set of rules r_n, and data set $\mathcal{D}$, it outputs the data policy $\mathcal{P}$. Inside the smart contract, the seller j can specify access control policy $\mathcal{P}$ for his/her data such as what data can be used for, and when it can be used.

4. **CreateIndex**$(\mathcal{SK}, \mathcal{D}, \mathcal{C}, \mathcal{P}) \longrightarrow \mathcal{I}$: It is run by the seller j to create the index $\mathcal{I}$. Given the encryption key of seller $\mathcal{SK}_j$, and encrypted data set $\mathcal{C}$, and access policy rules $\mathcal{P}$, it outputs the searchable index $\mathcal{I}$.

5. **Trapdoor**$(\mathcal{SK}, \mathcal{Q}) \longrightarrow \mathcal{T}_Q$: It is run by the seller j to create trapdoor for the authorized buyer i. Given the encryption key of seller $\mathcal{SK}_j$, and query (e.g., keyword) $\mathcal{Q}$, it outputs the trapdoor $\mathcal{T}_Q$.

6. **Search**$(\mathcal{I}, \mathcal{T}_Q) \longrightarrow \mathcal{S}$: It is run by buyer i and evaluated by smart contract (1 or 0). Given the search index $\mathcal{I}$, and trapdoor $\mathcal{T}_Q$, it outputs the search result $\mathcal{S}$, including the list of the encrypted data set $\mathcal{C}$.

7. **Eval**$(\mathcal{I}, \mathcal{T}_Q) \longrightarrow \pi$: Given index $\mathcal{I}$, and trapdoor $\mathcal{T}_Q$, it evaluates the search function and outputs the correctness proof π.

8. **Verify**$(\mathcal{T}_Q, \mathcal{S}, \pi) \longrightarrow$ *True* or *False*: Given trapdoor $\mathcal{T}_Q$, search $\mathcal{S}$, and proof π, it outputs the (inculding ciphertext $\mathcal{C}$), and correctness proof π. It outputs *True* if the result is valid (correct) and *False* otherwise.

9. **Decrypt**$(\mathcal{SK}, \mathcal{C}) \longrightarrow \mathcal{D}$: Given the encryption key $\mathcal{SK}$, and ciphertext data set $\mathcal{C}$, it outputs the decrypted data set $\mathcal{D}$ to buyer i.

The key principles of the proposed data market are data ownership, transparency, and access control. Algorithm 2 presents the scheme construction pseudo-code for the blockchain-based data trading system.

4.3 Implementation Setup Using Hyperledger Fabric

To implement the data trading system we use the Hyperledger platform. Hyperledger was established to address some of the concerns posed by permissionless blockchains. Under the Hyperledger umbrella, various purpose-specific systems and tools are being built for use-cases ranging from finance to the IoT, manufacturing, and supply chain management [64]. Hyperledger Fabric is one of the tools that offer permissioned distributed ledger technology allowing specific entities to participate [64]. Some of the main components of Hyperledger Fabric are as follows:

- Ledger: Immutable data storage tool that keeps the records of the transactions.
- Organization: A Fabric network might contain one or multiple organizations. Organizations host peers and other components of the network and each maintains a copy of the ledger. A single organization in a Fabric called *Org1* and uses the domain name *org1.example.com*.
- Peer: A Fabric network might contain one or multiple peers. Peers execute transactions and host in the Docker containers. Peers can be committer or endorser. All peers are committers by default. Peers receive orders in form of a block of transactions from the Orderer. Upon receiving a new block, the peer validates the transactions. Peers can take up the additional responsibility

Algorithm 2 Scheme construction for blockchain-based data trading system

Input: Encryption key $\mathcal{SK}$, document set $\mathcal{D}$, cipher-text data set $\mathcal{C}$, submitted bids and asking price, query $\mathcal{Q}$

Output: Winning buyer, payment, index $\mathcal{I}$, document set $\mathcal{D}$, smart contract

for each $j \in N$ **do**
  $\mathcal{SK} \leftarrow (1^{\lambda})$ // Calculate KeyGenSetup function and create the encryption key $\mathcal{SK}$
  $\mathcal{C} \leftarrow (\mathcal{SK}, \mathcal{D})$ // Encrypt the data set $\mathcal{D}$
  $\mathcal{P} \leftarrow ([r_1, r_2, ..., r_n])$ // Define the access policy rules for the data set $\mathcal{D}$
  $Sign_j$ the data access policy $\mathcal{P}$
  $\mathcal{I} \leftarrow (\mathcal{SK}, \mathcal{D}, \mathcal{C})$ // Create the index $\mathcal{I}$ for the data set $\mathcal{D}$
  Add the encrypted data set $\mathcal{D}$ to database
  if SC_j exists **then**
    Update new information for seller j in SC_j
  else
    Add/append SC_j and index to the blockchain for seller j
  end if
  Submit the ask price for data $\mathcal{D}$ to the blockchain
end for
for each $i \in B$ **do**
  Submit the bidding price to the blockchain
  Smart contract runs the auction mechansim
  Smart contract announces the winners and payments
end for
for each matching pair i and j **do**
  Seller j receives the payment
  $\mathcal{T}_Q \leftarrow (\mathcal{SK}, \mathcal{Q})$ // Create the trapdoor $\mathcal{T}_Q$ for the authorized buyer i
  Add the $\mathcal{T}_Q$ to the smart contract SC_j
  Share the $\mathcal{T}_Q$ and index $\mathcal{I}$ to buyer i
  $\mathcal{S} \leftarrow (\mathcal{I}, \mathcal{T}_Q)$ // Buyer i requests search using the $\mathcal{T}_Q$ and index $\mathcal{I}$
  if Verify $= 1$ **then**
    $\mathcal{D} \leftarrow (\mathcal{SK}, \mathcal{C})$ // Decryption process
  else
    Invalid
  end if
end for

of endorsing transactions (endorsers). An endorser simulates the transaction by running the chain code and appending the results with its cryptographic signature (referred to as endorsement) before returning it to the application.

- Orderer: The component responsible for ordering all the transactions, proposes new blocks, and seeks consensus (using one of the consensus protocols available in Hyperledger, e.g., Kafka, Solo, Raft) in the network. Peers are linked to orderers through channels so that when a new block is created, all peers linked to the orderer receive a copy of the new block.
- Chaincode (Smart contract): A piece of computer program which runs on the blockchain and enforces how applications communicate with the ledger.
- Database: LevelDB and CouchDB are the two existing state database solutions in Fabric to store data. LevelDB stores the data as key-value pairs and CouchDB stores the data as JSON.
- Channel: A component of the Fabric that allows private communication among different participants/organizations in the network. This will allow for data isolation (private subnet) and confidentiality among members.

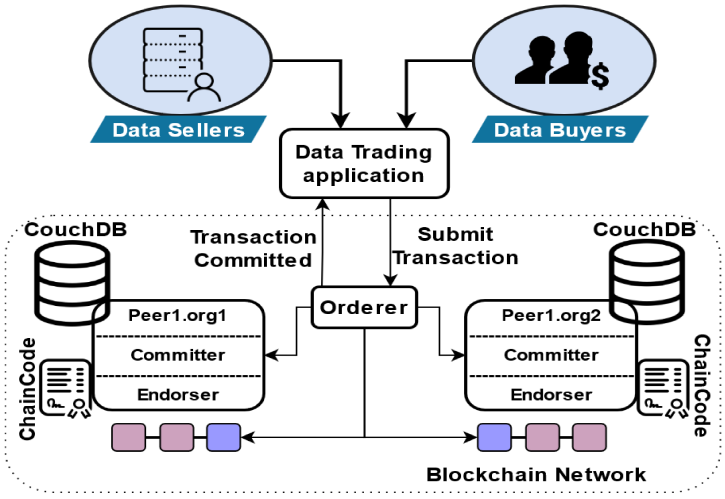


Figure 4.2: The blockchain transaction flow on Hyperledger Fabric V2.2 consisting of 2 organizations, each operating one running peer and a database inside a Docker container and orderer running a single channel.

Let us consider the Fabric network in Figure 4.2. In this example, sellers and buyers request a transaction through a data trading application (SDK). A transaction is a process of receiving the asks and bids from the sellers and buyers, respectively. The data trading system generates a transaction proposal and sends it to the orderer. Orderer broadcasts the transaction to the peers inside the channel. The peers

verify the transactions based on the predefined endorsement policy. Every chaincode contains an endorsement policy. It specifies which peers to execute chaincode and endorse (verify) the results for the transaction to be deemed valid. For example, it checks if the transaction proposed is in the correct format and the signature is valid, etc. The endorsed transaction is then returned to the application containing the result. Then, the application verifies the results if the endorsement policy is satisfied. Then, the orderer finalizes all the transactions in the network, seeks consensus (Solo consensus protocol), and creates a new block. Next, the orderer distributes the new block to all peers for the final validation. Finally, the peers validate the transactions and append them to the block. Since we only used one ordering node, Solo was chosen. Furthermore, we utilized CouchDB to model data on the ledger as JSON format. CouchDB enables us to deploy indexes in the chaincode to make queries efficient, thus empowering us to query large datasets.

4.4 Performance Analysis

The Fabric network we deployed for the experiment consists of two organizations (Org0, Org1), each consisting of one peer (peer0.org1.example.com, and peer0.org2.example.com), each consisting of one database (couchdb.org1.example.com, and couchdb.org2.example.com), and one orderer node (orderer.example.com) hosted inside a Docker container. We run the Hyperledger Caliper benchmark framework [65] over the implementation of Fabric to analyze our system performance on Ubuntu Linux Intel Core(TM) i7-3610QM CPU @2.30GHz with 6 GB RAM. Hyperledger Caliper allows users to execute, measure, and verify the performance of blockchain networks with predefined use cases [65]. The Hyperledger Caliper report file in an HTML format contains the performance of the blockchain network. Data is visualized via Python from various viewpoints for the readers' better understanding. Fig. 4.3 shows the data generation process. We used a real dataset UK-DALE [66] to emulate a real-life data trading scenario. The UK-DALE dataset contains five houses' energy usage consumption, measured by an IoT-based smart meter device. To make the data trading happen on the blockchain network, we converted the IoT data (energy consumption records) to a JSON file (key-value). We created three samples (assets) from UK-Dale dataset and stored them in CouchDB for our experiments. The results may change over different runs under different workloads. Therefore, the experiment itself must be repeatable. We run our experiments for eight rounds on the blockchain network to ensure the repeatability of the experiment. We measured our system performance

by using the following metrics:

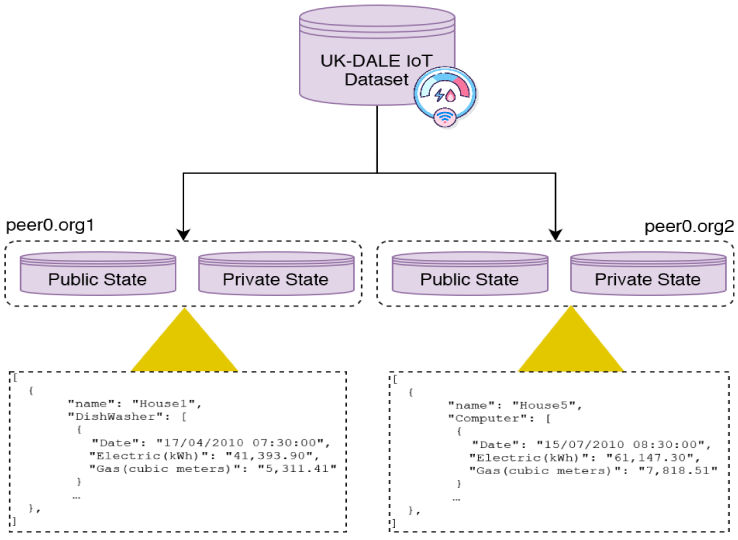


Figure 4.3: The data generation phase.

Transaction throughput: The average number of transactions per second (tps) that can be processed (i.e., written on the ledger) in the blockchain network. This metric measures how many transactions successfully processed on the blockchain. In the data trading systems, it is essential to find out the number of confirmed successful transactions in a unit of time to measure the performance of the system. Therefore, it is important to measure the system's capability of processing the transactions. The transaction throughput of the network calculated as follows [67]:

$$Transaction\ throughput = \frac{Total\ transcations}{Total\ time\ in\ seconds} \tag{4.1}$$

Transaction latency: The amount of time taken from the moment when a transaction is submitted till the moment when it is confirmed and available on the blockchain. This includes the propagation time and the processing time due to the consensus/ordering mechanism.

Resource consumption: The amount of computing resources (e.g., CPU usage, memory consumption) consumed by the blockchain network through different operations. This metric measures the cost efficiency of our system. Furthermore, based on this measure, users will be able to adapt algorithms to reduce resource consumption usage.

Elapsed time: The amount of time that each buyer needs to interact with the blockchain network to query data from chaincode. This metric measures how much time it takes for the buyer to get query result from database. Therefore, it is necessary to know the availability of the system in different requests.

It is important to test the system performance under different workloads. The workload is representative of the actual production usage. It is related to the amount of time and computational power used to complete a given task (e.g., amount of time and computing resources used to create a new transaction on the blockchain). We define the following workloads for our proposed system:

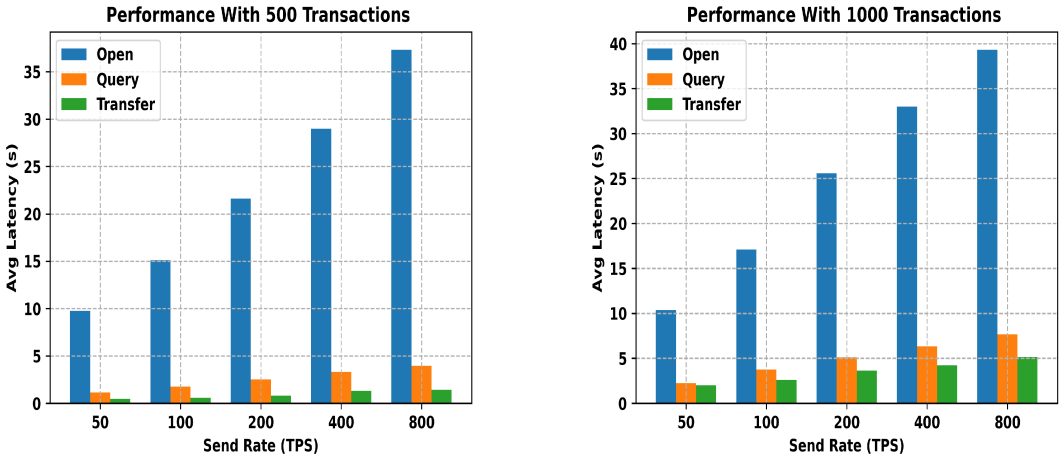
- Open workload: Opening accounts and testing the writing performance of the ledger.
- Query workload: Querying accounts and testing the reading performance of the ledger.
- Transfer workload: Data trading between accounts and testing the transaction performance of the ledger.

Table 4.1: Summary of Performance with 500 and 1000 Transactions

Name	Succ	Fail	Send Rate (TPS)	Max Latency (s)	Min Latency (s)	Avg Latency (s)	Throughput (TPS)
Open	500	0	50	6.38	3.37	9.75	38.5
Query	500	0	50	1.99	0.30	1.14	48
Transfer	500	0	50	0.79	0.14	0.46	50
Open	1000	0	50	12.41	8.32	10.36	48
Query	1000	0	50	4.12	0.78	2.24	38.2
Transfer	1000	0	50	3.67	0.34	2.00	50

Table 4.1 summarize the performance of the network with 500 and 1000 transactions using 50 transactions per second (TPS) for opening, querying, and transferring workloads, respectively. From the results in Table 4.1, we see that the average latency results for the 500 and 1000 transaction under the open workload is 9.75 and 10.36 seconds, respectively. The latency timer starts from the point of submitting the transaction and the result is available in our blockchain network. To evaluate the average latency blockchain network in more detail, we consider different sending

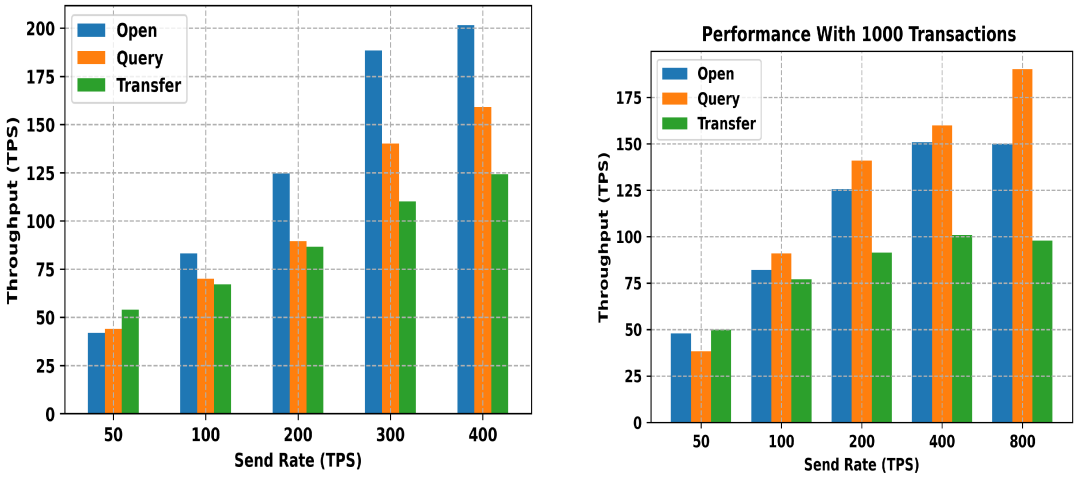
rates for analyzing the average latency of the proposed system. Figure 4.4 indicates the average latency for 500 and 1000 transactions with different sending rates 50, 100, 200, 400, 800 TPS, respectively. From Figure 4.4, we can see that by increasing the sending rate, the average latency of the open workload increases with the sending rate for both 500 and 1000 transactions. Sending a large number of transactions with higher sending rates would cause a failure in the network. Many other things can also cause transaction failures in the blockchain network, such as chaincode logic, version errors, peer resources, policy failures, network resources, consensus errors, and repeated transactions, to name a few. For query and transfer workload, from Figure 4.4, we can see that when the sending rate increases, the average latency increases slightly. Figure 4.5 shows the transaction throughput with different sending rates. The transaction throughput metric only measures valid committed transactions per second in the blockchain network. From the results in Figure 4.5 (a), we observe that by increasing the sending rate, the throughput of the query workload increased linearly, and the throughput of the open and transfer workload increases slightly too. However, in Figure 4.5 (b) the open and transfer workload for 1000 transactions will reaches a bottleneck when throughput gets close to 400, thus resulting into a higher latency average per transaction, and the throughput cannot be further improved.



(a) (b)

Figure 4.4: Average latency varying workloads and sending rates.

To create an asset in CouchDB the data trading application submits the *createAsset* method to the Hyperledger Fabric. Then, peers are required to verify the request through the Chaincode. Next, the orderer appends the transaction to the ledger and



(a) (b)

Figure 4.5: Transaction throughput under open, query, and transfer workloads with varied sending rates.

the asset is written in the database. On the other hand, to retrieve the asset, the data trading application submits the *getPrivateAsset* to the Fabric. Then, peers are required to verify the request through the Chaincode and allow the application to read the data from CouchDB. Asset retrieval will not have an interaction with the orderer. Tables 4.2 and 4.3 show the benchmark results of creating and retrieval assets within CouchDB, respectively. We can see the average latency increases with the size of the asset.

Table 4.2: Creating the assets in CouchDB

Asset Size (MB)	Max Latency (s)	Avg Latency (s)	Throughput
3.2MB	9.36	6.50	703.3
5.4MB	15.47	8.13	819.9
7.4MB	21.75	9.75	980.5

We ran the test queries on assets to calculate the elapsed time. The elapsed time begins when the query is executed, and ends when the query is returned. The following is an example of test query for the third asset (7.4Mb). The first query is not supported by the index while the second one is supposed by the index.

Query 1:

Table 4.3: Retrieval the assets from CouchDB

Asset Size (MB)	Max Latency (s)	Avg Latency (s)	Throughput
3.2MB	8.24	5.11	600.13
5.4MB	12.47	6.44	719.9
7.4MB	17.75	8.01	815.7

```
peer chaincode query -C $CHANNEL_NAME -n
ledger -c '{"Args":["QueryAssets",
"{\"selector\":{\"owner\":\"user3\"}]}'
```

Query 2:

```
peer chaincode query -C $CHANNEL_NAME -n
ledger -c '{"Args":["QueryAssets",
"{\"selector\":{\"docType\":\"asset3\",
\"owner\":\"user3\"}, \"use_index\":
[\"indexOwnerDoc\", \"indexOwner\"]}"]}'
```

Figure 4.6 shows running times of test queries without index and with index. From Figure 4.6, we can observe that querying with index takes less elapsed time compared with querying without index. In general, queries without index will have a longer elapsed time. Indexes allow a database to be queried without having to examine every row with every query, thus making them run faster and more efficiently.

Finally, tables 4.4, 4.5, 4.6 and 4.7, 4.8, 4.9 show the resource consumption for open, query, and transfer workloads with 500, and 1000 transactions, respectively. Memory displays the amount of memory used by the docker container on each test round. Memory(MAX) measures the maximum resources spent on a transaction, and memory(AVG) measures the average resources spent on all transactions. CPU displays the amount of CPU used by the docker containers during the test round. CPU(MAX) measures the maximum resources spent on a transaction, and CPU(AVG) measures the average resources spent on all transactions. For both 500 and 100 transactions, the resource consumption for open and query workloads reveals that CouchDB consumes the most memory and CPU, followed by the peers. The memory and CPU usage metrics indicate that with the increasing number of transactions from 500 to 1000, memory and CPU usage increases relatively. The network usage is conducted based on the traffic input and traffic output parameters. The majority of the network traffic is occupied by the orderer since it seeks the consensus in

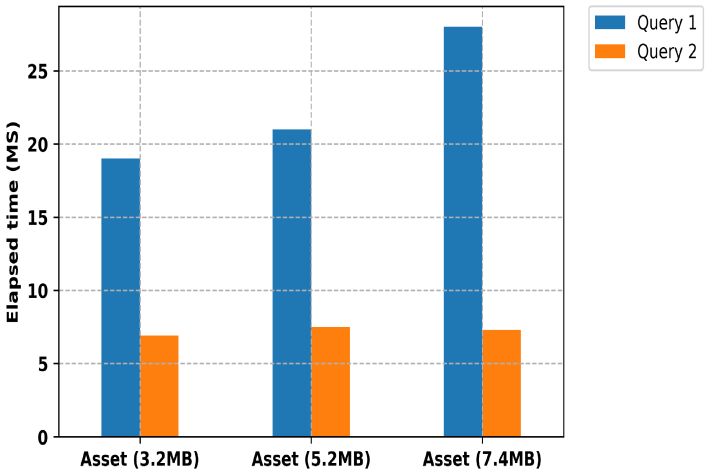


Figure 4.6: Running times of test queries.

the network in open workload. During our experiments, we observed that disk read is zero bytes, as there is no need to perform read operations on the ledger in open, and transfer workloads. The query transaction reads the data from the CouchDB in query workload. The performance analysis shows considerable low memory and CPU consumption for 500 and 1000 transactions. The peer node consumes an average of 54.1MB and 100.2MB for memory and 2.58% and 4.36% for CPU in 500 and 1000 transactions, respectively. This depicts that this blockchain network can be easily deployed in real-world applications with low-cost hardware. Focusing on IoT devices with the decentralized network. This will enable companies to deploy such a blockchain network to cut down the costs of expensive hardware and lead to successful secure data trading. Furthermore, since the average consumption is low it can be easily deployed on low-cost IoT devices such as Banana Pi, Raspberry Pi, VoCore, and Arduino.

Table 4.4: Resource Consumption for Open Workload with 500 Transactions

Type	Name	Memory (Max)	Memory (Avg)	CPU% (Max)	CPU% (Avg)	Traffic Input	Traffic Out	Disc Read/Write
Docker	peer0.org2.example.com	56.5MB	54.1MB	3.12	2.58	2.6MB	1.2MB	0B & 7.6MB
Docker	peer0.org1.example.com	27.1MB	21.2MB	3.8	1.45	2.2MB	1.1MB	0B & 7.2MB
Docker	orderer.example.com	6MB	5.4MB	1.22	0.14	2.1M	3.4M	0B & 4.6MB
Docker	couchdb.org2.example.com	71.1MB	67.9MB	35.3	22.9	3.1MB	2.9MB	0B & 3.4MB
Docker	couchdb.org1.example.com	68.5MB	65.7MB	31.1	20.4	3.1MB	2.9MB	0B & 3.1MB

Table 4.5: Resource Consumption for Query Workload with 500 Transactions

Type	Name	Memory (Max)	Memory (Avg)	CPU% (Max)	CPU% (Avg)	Traffic Input	Traffic Out	Disc Read/Write
Docker	peer0.org2.example.com	32.1MB	28.1MB	2.52	1.18	1.7MB	1.0MB	0B
Docker	peer0.org1.example.com	13.1MB	7.4MB	1.8	0.68	1.2MB	0.652B	0B
Docker	orderer.example.com	4.1MB	4.1MB	1.01	0.26	1.1MB	0.319B	0B
Docker	couchdb.org2.example.com	88.2MB	81.3MB	43.9	30.5	4.4MB	4.1MB	1.2M & 0B
Docker	couchdb.org1.example.com	86.1MB	79.4MB	40.8	29.1	4.2MB	3.9MB	1.0M & 0B

Table 4.6: Resource Consumption for Transfer Workload with 500 Transactions

Type	Name	Memory (Max)	Memory (Avg)	CPU% (Max)	CPU% (Avg)	Traffic Input	Traffic Out	Disc Read/Write
Docker	peer0.org2.example.com	32.1MB	28.1MB	2.52	1.18	1.7MB	1.0MB	0B & 7.6MB
Docker	peer0.org1.example.com	13.1MB	7.4MB	1.8	0.68	1.2MB	0.652B	0B & 7.2MB
Docker	orderer.example.com	9MB	7.1MB	1.04	0.54	614B	153B	0B & 4.6MB
Docker	couchdb.org2.example.com	70.6MB	66.7MB	34.3	21.0	2.9MB	2.4MB	0B & 3.3MB
Docker	couchdb.org1.example.com	68.1MB	65.3MB	31.1	19.2	2.9MB	2.4MB	0B & 3.1MB

Table 4.7: Resource Consumption for Open Workload with 1000 transactions

Type	Name	Memory (Max)	Memory (Avg)	CPU% (Max)	CPU% (Avg)	Traffic Input	Traffic Out	Disc Read/Write
Docker	peer0.org2.example.com	101.9MB	100.2MB	6.8	4.12	3.7MB	3.5MB	0B & 11.7MB
Docker	peer0.org1.example.com	91.2MB	88.1MB	6.1	3.39	3.7MB	3.5MB	0B & 11.1MB
Docker	orderer.example.com	14.7MB	12.7MB	2.98	1.19	3.4M	7.1M	0B & 8.6MB
Docker	couchdb.org2.example.com	107.1MB	100.9MB	47.9	32.6	4.4MB	6.2MB	0B & 5.1MB
Docker	couchdb.org1.example.com	101.8MB	98.5MB	46.1	31.2	4.4MB	6.2MB	0B & 4.7MB

Table 4.8: Resource Consumption for Query Workload with 1000 transactions

Type	Name	Memory (Max)	Memory (Avg)	CPU% (Max)	CPU% (Avg)	Traffic Input	Traffic Out	Disc Read/Write
Docker	peer0.org2.example.com	91.8MB	87.1MB	5.41	4.36	2.4MB	2.1MB	0B
Docker	peer0.org1.example.com	78.9MB	73.2MB	5.11	4.08	2.2MB	1.8MB	0B
Docker	orderer.example.com	12.1MB	10.9MB	2.11	1.51	3.1MB	6.6MB	0B
Docker	couchdb.org2.example.com	88.2MB	81.3MB	43.9	30.5	4.4MB	4.1MB	1.4M & 0B
Docker	couchdb.org1.example.com	86.1MB	79.4MB	40.8	29.1	4.2MB	3.9MB	1.1M & 0B

Table 4.9: Resource Consumption for Transfer Workload with 1000 transactions

Type	Name	Memory (Max)	Memory (Avg)	CPU% (Max)	CPU% (Avg)	Traffic Input	Traffic Out	Disc Read/Write
Docker	peer0.org2.example.com	91.8MB	87.1MB	5.41	4.36	2.4MB	2.1MB	0B & 11.7MB
Docker	peer0.org1.example.com	78.9MB	73.2MB	5.11	4.08	2.2MB	1.8MB	0B & 11.1MB
Docker	orderer.example.com	15.1MB	15.0MB	1.03	0.54	2.6MB	3.3MB	0B & 8.6MB
Docker	couchdb.org2.example.com	77.2MB	71.3MB	38.9	29.6	4.1MB	3.8MB	0B & 5.1MB
Docker	couchdb.org1.example.com	75.5MB	70.1MB	36.2	27.5	4.1MB	3.8MB	0B & 4.7MB

4.5 Summary

IoT devices and data-driven applications are generating huge data. Such valuable data received considerable attention from organizations to tailor their services to potential consumers and make a profit. On the other hand, data owners can sell their private data for profit. Data trading can be used to achieve efficient use of data resources and increase data valuation. In recent years, several studies focused on designing data trading systems based on blockchain in which sellers and buyers can interact securely with each other. However, conflicting issues have been raised regarding the performance of data trading systems in the blockchain network. This chapter presents a performance analysis of a blockchain-based data trading system using Hyperledger Caliper. Specifically, we highlighted the importance of transaction throughput, elapsed time, and CPU usage as performance parameters for system design efficiency. The experimental results show that the proposed system can easily be deployed on IoT devices at a low cost.

Chapter 5

A Blockchain-based Reputation System for IIoT Data Ecosystem

In this chapter, we discuss the existing challenges of blockchain-based reputation system for IIoT data ecosystem in section 5.1. In section 5.2, the blockchain-based reputation system for IIoT data ecosystem is presented, followed by security analysis in section 5.3. Section 5.4 presents implementation and experimental results of the proposed system. Finally, conclusions are drawn and future research directions are discussed in section 5.5.

5.1 Introduction

In 21st century's Industry 4.0, IIoT data is considered the most valuable asset in use as well as in trade [68]. The radically increasing amount of data generated by IIoT devices has led to the emergence of a new IIoT data ecosystem. In such an ecosystem, data providers generate an infinite number of data records from various IIoT applications (e.g., autonomous systems, smart healthcare, etc.) and sell them to data consumers [69]. Such valuable data received considerable attention from data consumers to tailor their services to potential consumers and make a profit [70]. Several firms, like Terbine and Dfintech, have developed online data trading systems for monetizing IIoT data. With the tremendous growth of online marketplaces ranging from e-commerce companies to data marketplaces, the need for a robust intelligent reputation systems for Industry 4.0 is becoming increasingly vital as more people and services interact online. The important components of a reputation system are

product ratings and reviews [71]. These components not only assist consumers with their purchasing decisions but are also valuable for data providers to build their trust [72].

However, current IIoT-based data trading systems suffer from three main concerns. To begin, current systems such as CitizenMe, Terbine, Datacoup, DataExchange, and Factual, to name a few, use a centralized-based reputation system, which is vulnerable to data leakage [73], and hence, a distributed blockchain system may alleviate this problem. Furthermore, because user accounts are not anonymous and may be traced, the process of posting reviews may reveal personal information about consumers [36]. Thus, consumers may be reluctant to post a review. On top of that, reputation systems are vulnerable to a variety of attacks, including Whitewashing, Self-promotion, Sybil, Slandering, and Bad-Mouthing, to mention a few. To address these issues, researchers propose blockchain-based reputation systems [35, 36, 37, 41, 42, 43] as a means of providing anonymity, transparency, increase mutual trust, and security for both providers and consumers in Industry 4.0. These studies focus on the decentralized reputation system with a single certificate authority, which creates the concern of a SPOF. They utilized a consensus mechanism that is not crash fault-tolerant and cannot continue to operate despite the failures. Moreover, they paid little attention to the performance measures of these blockchain-based reputation systems to demonstrate their usability in a real IIoT data ecosystem. In this chapter, the proposed blockchain-based reputation system is capable of avoiding a SPOF, by allowing the operation to proceed despite the failures. We provide a detailed performance analysis to ensure the feasibility and usability of the proposed system in real-world applications. The main contributions of this chapter are as follows:

- To the best of our knowledge, this is the first work to design a blockchain-based reputation system for the IIoT data ecosystem for Industry 4.0. We propose an anonymous reputation system for the IIoT data ecosystem by leveraging a blind Elliptic Curve Digital Signature (ECDSA) and a non-interactive zero-knowledge proof (ZKP) technique.
- We build a blockchain network based on the Raft consensus algorithm. With Raft, the proposed system is a crash fault-tolerant, which allows the operation to proceed as planned rather than failing. We further improve the Raft consensus algorithm to avoid SPOF, and link failures. We built a new policy called RepGossip based on the gossip protocol, which withstands the link failures.

- We demonstrate and provide a comprehensive security analysis of the proposed system which satisfies review completeness, review soundness, review anonymity, and review unlinkability properties.
- We analyze and evaluate the performance of the blockchain-based reputation for IIoT data ecosystem using Hyperledger Fabric. Unlike existing research, our work measures and analyzes transaction throughput, latency, memory usage, and CPU consumption.

5.2 System Model

In this section, we describe the blockchain-based reputation system for the IIoT data ecosystem which is shown in Figure 5.1. To give the reader an idea about the involved parties in this system, consider the example in [74] where a transportation department (i.e., data consumer) in a city would like to acquire information about traffic conditions, logistics, etc. The department would subscribe to Amazon Data Exchange which collects IIoT data from different data providers (owners of traffic IIoT devices with intelligent edge computing). In this example, the data consumer would benefit from the data to improve decision-making and services while the data providers trade their data for monetary benefit. The proposed system comprises several components that interact with one another, as explained below.

1. Data providers: A data provider $i \in \{1, 2, \ldots, B\}$ sells data records, which is generated by IIoT devices for a monetary value. Data records are any type of records sold by a provider i in a IIoT data ecosystem. Let $\mathcal{X}_i$ be the set of data records, and let x_i be a data package $x_i \in \mathcal{X}_i$. The reputation of a data provider i is built on the feedback of data consumers.
2. Data consumers: A data consumer $j \in \{1, 2, \ldots, N\}$ make purchases from data provider i, and may leave feedback on his/her purchase.
3. Certificate authority (CA): It is a government agency responsible for authenticating data providers and consumers by registering and issuing certificates to them. CA maintains a public ledger $\mathcal{L}$ based on the Raft consensus protocol.

At a high level, our proposed system works as follows. Data providers and consumers register themselves to the CA. The CA issues certificates (digital identities

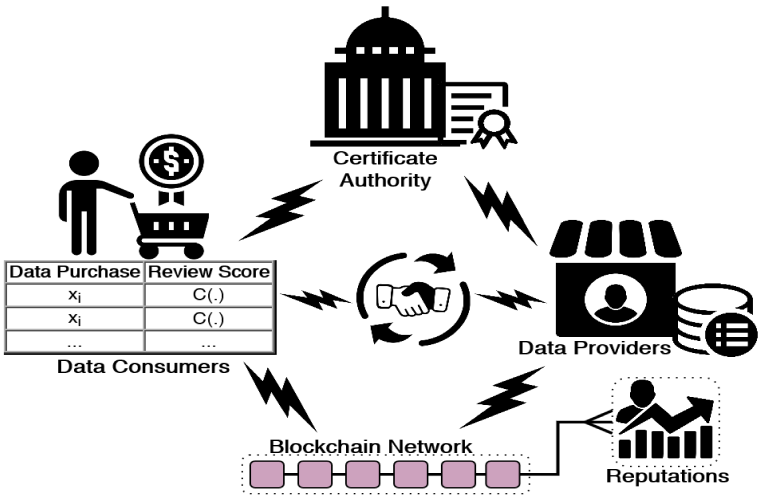


Figure 5.1: High-level architecture of the proposed system.

contained in X.509 digital certificates) to each entity. Afterward, data consumer j can make purchases from provider i. CA issues a rating token to the consumer j after the purchase is completed and verified. Later, a consumer j can leave a review score for a provider i. A review will be added to the $\mathcal{L}$. Finally, the CA accumulates and calculates the review scores and updates j's overall reputation score of the provider $i \in B$.

5.2.1 Design Goals

There are six significant security properties that our proposed system aims to achieve as follows:

1. Review anonymity: The consumer should be anonymous in the rating score process to protect the consumer's privacy. Moreover, review scores in the ledger $\mathcal{L}$ will not expose a data consumer j's identity.

2. Review unlinkability: Given two or more payment transactions, unlinkability guarantees that no entity can determine whether two or more valid reviews are from the same data consumer.

3. Review correctness: A data consumer j who performs a payment for a data package x_i must be able to leave a review. This property guarantees no consumer j can leave a review score without making the payment.

4. Review soundness: A data consumer j is only able to leave exactly one review per purchase with a valid payment transaction. This ensures that dishonest consumers can not leave a review.

5. Review completeness: All the reviews must be publicly available and visible in $\mathcal{L}$ for all consumers N and providers B without revealing identities.

5.2.2 System Setup

In this sub-section, we briefly describe the basic definition of bilinear pairing, non-interactive ZKP technique, and blind ECDSA schema as building blocks of the proposed system. Let $\mathbb{G} = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T)$ be multiplicative cyclic groups with a bilinear paring $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$, g_1 and g_2 are generators of $\mathbb{G}_1$ and $\mathbb{G}_2$, and a prime order q. CA defines a security parameter λ and generates public parameters $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, q, e, \mathcal{H})$ for consumers and providers, where $\mathcal{H}$ is a collision-resistant hash function. CA selects $\mathcal{H} : \{0,1\}^* \rightarrow \mathbb{Z}_q$, where $\mathbb{Z}_q$ is a finite field of order q. Then, CA chooses a random numbers $s_1, s_2 \in \mathbb{Z}_q^2$ as private key and computes $g_2^{s_1}, g_2^{s_2} \rightarrow (\tilde{Q}, \tilde{U})$. The public system parameters are $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, q, e, g_1, g_2, \tilde{Q}, \tilde{U}, \mathcal{H})$.

Definition 7 (Non-interactive ZKP [75]). *The non-interactive ZKP is the safest method of authentication of the entities. It allows entities to validate transactions on the blockchain network without reviling sensitive information (e.g., password) associated with the transaction (e.g., Zcash crypto platform is a real-world application that utilizes ZKP). Formally, given an element e of a language $L \in NP$ language, an entity (prover) is able to convince other entity (verifier) that e belongs to L, i.e. w is witness for e.*

Definition 8 (Blind ECDSA schema [76]). *ECDSA is a cryptographic algorithm used in many blockchain-based such as bitcoin other blockchain systems to secure transactions and increase efficiency. Blind ECDSA adds an extra layer of security and anonymity to the standard ECDSA, allowing users to acquire a signature from signers while preventing the signers from learning any information about the message. More formally, the blind ECDSA between the recipient r and the signer s is defined as follows:*

- *Signer s randomly selects an integer n_1 from 2 to $q-1$.*
- *Signer s computes $K_1 = n_1 g_1$ and send is it to r.*

- *Recipient r randomly selects an integer n_2 from 2 to $q-1$.*
- *Recipient r computes $K = (k_x, k_y) = n_2 K_1$.*
- *Recipient r selects two distinct primes t_1, t_2 and computes public key (N, g).*
- *Recipient r encrypts k_x and k_y and randomly selects $r_1, r_2 \in \mathbb{Z}^*_{N^2}$.*
- *Recipient r computes $C_1 = g^{k_x} r_1^N (mod N^2)$ and $C_2 = g^{k_y} r_2^N (mod N^2)$.*
- *Recipient r submits (N, g, C_1, C_2) to singer s.*
- *Signer s selects r from 2 to N and computes $C = (C_1 C_2^{sk})^{k_1^{-1}(modq)} r^N (mod N^2)$ and C to recipient r.*
- *Recipient r computes $s = k_2^1 D\big(C, (p,t)\big)(modq)\big(D(.)$ decryption algorithm$\big)$.*
- *Recipient r obtains a blind signature $\sigma = (k_x, s)$.*

Significant notation is summarized in Table 5.1 for the clarity of readers.

Table 5.1: Notations.

Notation	Description
CA	Certificate authority
i	Data provider $i \in \{1, 2, \ldots, B\}$
j	Data consumer $j \in \{1, 2, \ldots, N\}$
$\mathcal{X}_i$	Set of data records
x_i	A data package
σ_i	Data provider i's signature
σ_j	Data consumer j's signature
σ_{CA}	CA j's signature
$\mathcal{L}$	Ledger
$C^t_{j,n}(x_i)$	Review score of consumer j on provider i's data package
$\xi^t_{j,n}$	Accumulated value deviation for the history of all transactions
$\mathcal{R}_i$	Total reputation score of data provider i

5.2.3 System Registration

Each data provider i and data consumer j register themselves to CA to obtain anonymous credentials. Provider i and consumer j choose a secret key $s_i \in \mathbb{Z}_q^2$, $s_j \in \mathbb{Z}_q^2$ and computes $g_1^{s_i}, \tilde{Q}^{s_i} \rightarrow (M_i, \bar{M}_i)$, $g_2^{s_j}, \tilde{U}^{s_j} \rightarrow (N_j, \bar{N}_j)$, respectively. Then, provider i and consumer j generate a ZKP [75] π_{s_i}, π_{s_j}, respectively as follows:

$$s_i : M_i = g_1^{s_i} \wedge \bar{M}_i = \tilde{Q}^{s_i} \tag{5.1}$$

$$s_j : N_j = g_2^{s_j} \wedge \bar{N}_j = \tilde{U}^{s_j} \tag{5.2}$$

Then, provider i and consumer j send their public keys, signatures and proofs $(M_i, \bar{M}_i, \sigma_i, \pi_{s_i})$, $(N_j, \bar{N}_j, \sigma_j, \pi_{s_j})$ to CA, respectively. Afterward, CA checks the validity of proofs π_{s_i}, π_{s_j}, and $e(M_i, \tilde{Q}) \stackrel{?}{=} e(g_1, \bar{M}_i)$, $e(N_j, \tilde{U}) \stackrel{?}{=} e(g_2, \bar{N}_j)$ for provider i and consumer j, respectively. CA ignores when the proofs are invalid. Otherwise, CA issues and signs the certificates for provider i and consumer j as follows:

$$\sigma_{CA}(M_i, \bar{M}_i, \sigma_i, \pi_{s_i}) \rightarrow Cert_i \tag{5.3}$$

$$\sigma_{CA}(N_j, \bar{N}_j, \sigma_j, \pi_{s_j}) \rightarrow Cert_j \tag{5.4}$$

The CA stores $\sigma_{CA}(M_i, \bar{M}_i, \sigma_i, \pi_{s_i})$, $\sigma_{CA}(N_j, \bar{N}_j, \sigma_j, \pi_{s_j})$ in the blockchain and send the $Cert_i$, $Cert_j$ to provider i and consumer j, respectively.

5.2.4 Payments

In the payment stage, the provider i sends its account (e.g., wallet address) and the price of data package x_i to the consumer j. When a data consumer j decides to purchase the data package x_i from provider i, consumer j can transfer the money anonymously via secure payment channels such as Bolt, and Zerocash. For example, Bolt anonymous payment channel scheme includes a tuple of probabilistic algorithms $(KeyGen, Init_i, Init_j, Refund, Refute, Resolve)$ [77]. At beginning, data provider i and consumer j generate their public and private keys using bilinear pairing cryptography, as discussed in section 5.2.2. Then, both execute the $Init_i, Init_j$ algorithms to initiate the private channel to drive tokens. Next, they send the tokens alongside a transaction to the payment network. If the transaction succeeds, the provider i receives $Establish$ protocol, and customer j receives Pay protocol. The data consumer j runs the Pay protocol to process the payment. Afterward, data consumer j runs the $Refund$ algorithm to end the payment channel, and data provider i runs

Refute algorithm to revoke the tokens. Then, provider i generates a transaction proposal (TxProposal) that contains of the payload (includes all of the metadata about a transaction), and txID, certificates, along with a cryptographic signature of himself and consumer j on the transaction header, and eventually submits the transaction to the CA. A payment transaction proposal $\mathcal{P}$ looks as follow:

$$\mathcal{P} = \Big(txID, \sigma_i, \sigma_j, Cert_i, Cert_j, pType, payload \Big) \tag{5.5}$$

where $TxID$ is the unique transactionID, $\sigma_i, \sigma_j, Cert_i, Cert_j$ are signatures, and certificates of provider i and consumer j, respectively. $pType$ is the type of payment (Zerocash, Bolt, etc.). Afterward, once CA receives the transaction proposal, CA checks $TxID \notin \mathcal{L}$, and verifies the $Cert_i$, and $Cert_j$ and its corresponding σ_i, and σ_j digital signatures from $\mathcal{L}$. If the transaction satisfies the requirements then, CA adds the $\mathcal{P}$ to the ledger $\mathcal{L}$.

5.2.5 Reputation Computation and Verification

Once the payment transaction p is appended to the blockchain, and the consumer j received the anonymous rating token to_j from CA. Consumer j has the option to leave a review score for the data package x_i. $C^t_{j,n}(x_i)$ represents the review score of consumer j has upon provider i based on its recent transaction n in the time t. The review score function is defined as follows:

$$C^t_{j,n}(x_i) = Sat_c \times \frac{(1 - e^{-\Omega Val^t_{j,n}})}{1 + \xi^t_{j,n}} \tag{5.6}$$

$$Val^t_{j,n} = |Val^t_{j,\bar{n}} - Val_c| \tag{5.7}$$

$$\xi^t_{j,n} = \mathcal{K} \times Val^t_{j,n} + (1 - \mathcal{K}) \times \xi^t_{j,\bar{n}} \tag{5.8}$$

Sat_c is a rating score between $[0, 1]$ for the current transaction n reflecting the way consumer j rates provider i. If $Sat_c = 0$, then consumer j is totally unstatisied with provider i. If $Sat_c = 1$, then consumer j is totally statisied with provider i. $Val^t_{j,n}$ is a recent value fluctuation between the previous and current value Val_c. Here, Ω is the decay constant and it controls the review score. The $C^t_{j,n}(x_i)$ value reaches 1.0 with larger $Val^t_{j,n}$ and decreases slowly with smaller $Val^t_{j,n}$. We consider such a value fluctuation to make the review score fair. This makes sense because

private or sensitive data is more valuable than public data. For example, suppose the value of the transaction is trivial (e.g., data package contains public data), and the current satisfaction of consumer j is high. In that condition, the review score will not significantly impact the overall reputation. On the other side, if the value of the transaction is high (e.g., data package contains private data), and the current satisfaction of consumer j is high. Then, the review score of consumer j will have a considerable impact on the overall reputation data provider $i \in B$. $\xi_{j,n}^t$ represents the accumulated value deviation for the history of all transactions [54]. $\mathcal{K}$ is a constant parameter that determines how much to react with respect to the recent value fluctuations $Val_{j,n}^t$ [54]. Once consumer j calculate the review score $C_{j,n}^t(x_i)$. Then, consumer j signs the review score and generates a transaction proposal as follows:

$$\mathcal{T} = \left(txID, Cert_j, \sigma_j, to_j, \left(C_{j,n}^t(x_i) \right) \right) \tag{5.9}$$

Afterward, once CA receive the transaction proposal from consumer j, CA runs Algorithm 3 to verify the transaction. If the transaction satisfies the requirements then, CA adds the $\mathcal{T}$ to the ledger $\mathcal{L}$. The, CA calculate the review scores of all consumers N as follow:

$$\mathcal{R}_i = \sum_{j=1}^{N} \left(\frac{C_{j,n}^t(x_i)}{N} \right) \tag{5.10}$$

where $\mathcal{R}_i$ is the total reputation score of data provider i, which is computed using the current review score $C_{j,n}^t(x_i)$. CA announces or updates the total reputation $\mathcal{R}_i$ of data provider $i \in B$ and append it to the $\mathcal{L}$. After several transactions, the data provider $i \in B$ obtains a complete view of his/her total reputation.

5.3 Security Analysis

In this section, we provide a security analysis of the proposed system which satisfies the properties of the review anonymity, review unlinkability, correctness, soundness, and completeness.

Lemma 1. *The proposed system satisfies the review anonymity property.*

Proof. To achieve full anonymity in our system, we utilized an interactive identification protocol based on the ZKP. This strategy ensures that consumers who post review scores remain anonymous. Its security analysis has been proven in [75]. □

Algorithm 3 Reputation computation and verification stage

Input: $\mathcal{T} = \left(txID, Cert_j, \sigma_j, to_j, \left(C^t_{j,n}(x_i)\right)\right)$

Output: $\mathcal{L}$, R_i

1: **for** each $j \in N$ **do**
2: Calculate the $C^t_{j,n}(x_i) \leftarrow Sat_c \times \frac{(1-e^{-\Omega Val^t_{j,n}})}{1+\xi^t_{j,n}}$
3: Calculate the $Val^t_{j,n} \leftarrow |Val^t_{j,\bar{n}} - Val_c|$
4: Calculate the $\xi^t_{j,n} \leftarrow \mathcal{K} \times Val^t_{j,n} + (1-\mathcal{K}) \times \xi^t_{j,\bar{n}}$
5: **end for**
6: Submit $\mathcal{T} = \left(txID, Cert_j, \sigma_j, to_j, \left(C^t_{j,n}(x_i)\right)\right)$ to CA
7: Check $TxID \notin \mathcal{L}$
8: From $\mathcal{L}$, verify $Cert_j$
9: Verify the σ_i digital signature
10: Add $\mathcal{T} \rightarrow \mathcal{L}$
11: Calculate the $\mathcal{R}_i \leftarrow \sum_{j=1}^{N} \left(\frac{C^t_{j,n}(x_i)}{N}\right)$
12: Announce the $\mathcal{R}_i$
13: Add/update $\mathcal{R}_i \rightarrow \mathcal{L}$

Lemma 2. *The proposed system satisfies the review unlinkability property.*

Proof. Let's assume that there is an adversary $\mathcal{A}$ node in CA that can link two reviews of the same consumer. An adversary $\mathcal{A}$ receives transaction $\mathcal{T}$ to verify the review and add the review to the ledger $\mathcal{L}$. All these parameters in $\mathcal{T}$ are blinded because of the blind ECDSA schema and ZKP method. Therefore, the proposed system satisfies the review unlinkability property. Blind ECDSA security analysis has been proven in [76]. □

Lemma 3. *The proposed system satisfies the review correctness property.*

Proof. A payment is completed when the data provider $i \in N$ submits the payment transaction $\mathcal{P}$ to the CA. Upon the verification transaction $\mathcal{P}$, the CA will add the $\mathcal{P}$ to the $\mathcal{L}$. More formally, $Pr\left[\mathcal{P} \longrightarrow \mathcal{L}\right] = 1$. Then, CA generates the rating token for the consumer j. Thus, the proposed system satisfies the review correctness property. □

Lemma 4. *The proposed system satisfies the review soundness property.*

Proof. We assume two attack scenarios. First, adversary $\mathcal{A}$ is a third party attacker and does not have a legal identity in the system and wants to leave multiple reviews. Second, the adversary $\mathcal{A}$ is a dishonest entity against which wants to leave multiple reviews. In the first scenario, without a valid payment transaction in $\mathcal{L}$, valid certificate, and valid token, a third-party attacker $\mathcal{A}$ would not be able to leave reviews. In the second scenario, due to forgery resistance of ECDSA schema dishonest entity $\mathcal{A}$ cannot leave multiple reviews. If a $\mathcal{A}$ signs two reviews using the same token, then the two signatures will be linked, and CA can detect this misbehavior. Furthermore, tokens generated by CA can only be used once. If the identical token is provided to CA, the transaction will be revoked. □

Lemma 5. *The proposed system satisfies the review completeness property.*

Proof. A review is completed when the data consumer $i \in B$ submits the transaction $\mathcal{T}$ to the CA. Upon the verification transaction $\mathcal{T}$ using Algorithm 3, the CA will add the $\mathcal{T}$ to the ledger $\mathcal{L}$. More precisely, $Pr\big[\mathcal{T} \longrightarrow \mathcal{L}\big] = 1$. Then, CA calculates the aggregated review scores $\mathcal{R}_i$ and publish it in $\mathcal{L}$. Therefore, the proposed system satisfies the review completeness property. □

5.4 Implementation and Experimental Results

In this section, first, we present our experiment settings. Then, we describe the Raft consensus mechanism and its transaction workflow on Hyperledger Fabric. Afterward, we present our implementation and experiments. Finally, the performance of the proposed system is analyzed based on the transaction latency, throughput, and CPU consumption metrics. We used Hyperledger Fabric blockchain for our implementation. According to a penetration test report on Hyperledger Fabric from Tevora Threat Research Group [78], no severe vulnerabilities were discovered that may lead to an attack. They determined Fabric core security architecture other critical components were natively secure by both design and default. The Fabric uses a data isolation approach and secure channels to preserve users' data privacy. While the traditional technique protects privacy by requiring a login password, this method poses security problems. We instantiate one organization (.org1) consisting of a certificate authority (ca.org1) node, five orderer nodes, and two peers nodes on the Ubuntu Linux 20.04.3, Intel Core(TM) i7-3610QM CPU @2.30GHz with 16

GB RAM. The total transactions are 500, and we set transactions per second (tps) to 50, 100, 200, 300, 400tps. The H function is SHA256. The elliptic curve is $y^2 \equiv x^3 + ax + b$ over $\mathbb{Z}_q$ (prime256v1). The rating reviews are generated randomly.

5.4.1 Raft Consensus Mechanism

Figure 5.2 shows the flow diagram of the Raft consensus mechanism based on Hyperledger Fabric. It consists of the ledger(s), organization(s), certificate authority, peer nodes, orderer nodes, chain code (smart contract), database, and private channel components. We deployed the Raft consensus mechanism due to its crash fault-tolerant resiliency and SPOF avoidance. In the Raft mechanism, an orderer node can be in three states [79]: leader, candidate, and follower. The orderer node is responsible for seeking the consensus in the blockchain network among other orderer nodes. Peer nodes receive instructions (in the form of transactions) from the orderer nodes. The peer nodes (Peer0.org1 and Peer1.org1) execute a Raft consensus protocol to validate and endorse transactions [80]. Each peer node ensures that the ledger $\mathcal{L}$ is up-to-date on every single request (in collaboration with orderers) and keeps copies of ledgers $\mathcal{L}$ and smart contracts.

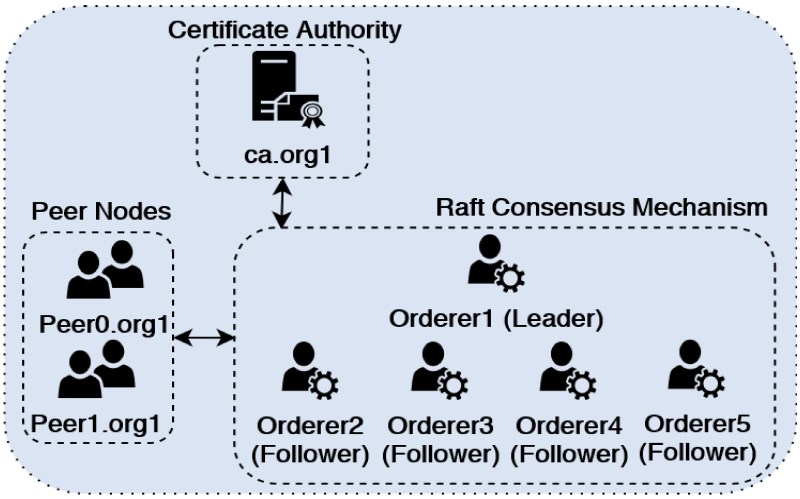


Figure 5.2: The transaction workflow on Hyperledger Fabric V2.2 using Raft consensus mechanism.

When the algorithm initializes, the existing orderer nodes in the blockchain network will vote for the candidates. A candidate becomes a leader node as soon as it

gets a quorum of the votes. A leader is always assumed to act honestly. When the existing leader fails, a new leader needs to be selected. Then, a new term starts in the network with the leader election. The raft algorithm divides time into terms. The terms are numbered sequentially with integers. If the election is completed successfully, the term keeps going with normal operations under leader supervision. Then, the leader is responsible for the log replication. It receives requests from clients. Each client request consists of transactions that the network will perform. The leader transmitted the transactions to the followers to process and add to the ledger $\mathcal{L}$ once the transactions were verified to be authentic.

5.4.2 Experimental Results

Link failures

The links between leader and followers can have a high failure rate. As result, leader elections fail in such a network environment because of a lack of received RequestsVotes (a message sent by a candidate to start an election and ask each node for their vote) and RequestVoteReplies, and the chances of the candidates to win the leader elections will decrease significantly. We built a new policy called RepGossip which withstands the link failures. In RepGossip, we used Fabric's private channel for the communication between leaders and followers to maintain confidentiality. Then, we implement a gossip protocol for each node on the private channel to ensure that messages are disseminated to all of the nodes. For example, if a candidate sets up an election procedure, and broadcasts a message to the nodes. Recipient nodes propagate that message through the whole network. As a result, a message is disseminated to all the nodes in the blockchain network and will significantly reduce the link failures. Figure 5.3 shows the only link failures of the network using RepGossip policy, ReplicaRV [81] policy, and without using any policy (Raft itself). To acquire an accurate result, we used 200 trials, as ReplicaRV [81]. Figure 5.3 depicts that with RepGossip policy number of link failures reduced significantly compared to ReplicaRV [81], and Raft itself.

Single point of failure

The Raft consensus mechanism is crash fault-tolerant, which provides high level of resiliency in the blockchain network. Even if some nodes fail, it correctly takes responsibility and reaches a consensus. It can tolerate $\mathcal{F}$ number of crash nodes

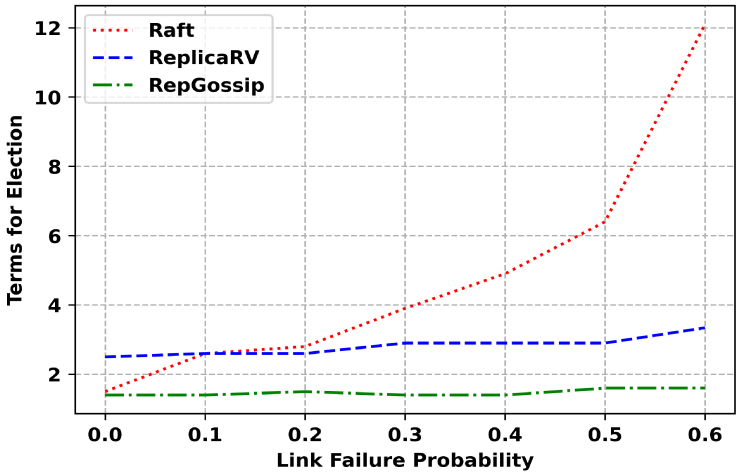


Figure 5.3: The result of running the RepGossip protocol during the election process with non-negligible link failures. A link failure of 0.1 indicates that every message sent over a particular link has a likelihood of 10% lost.

in $n = 2\mathcal{F} + 1$, where n is the total number of total nodes [79]. To the analysis of our blockchain network reliability, we consider the 10 nodes. In each period, we crashed a number of nodes to test the network against SPOF. To the analysis of our blockchain network reliability, we consider the 10 nodes with 200 trials. In each period, we crashed a number of nodes to test the network against SPOF. Figure 5.4 depicts the timeout intervals encountered nodes crashed. From Figure 5.4, we can see that there is no SPOF occurred and the timeout intervals are negligible. We observe that by increasing the crashed nodes, the timeout interval increases slightly too.

Computational costs

Figure 5.5 shows the computational cost of user registration (data provider and data consumer) and review generation and verification. These criteria assist us in measuring the efficiency of the algorithms in terms of execution time and estimating the execution-time budget for the real-time machines. We further compare our proposed system with the most recent studies: Beaver [35], ARS-PS [36], and Repchain [43]. Some of these studies do not provide the computational cost of the selected criteria. Beaver [35], and Repchain [43] do not provide computational cost for registration and review verification, respectively. These studies have built their models using the

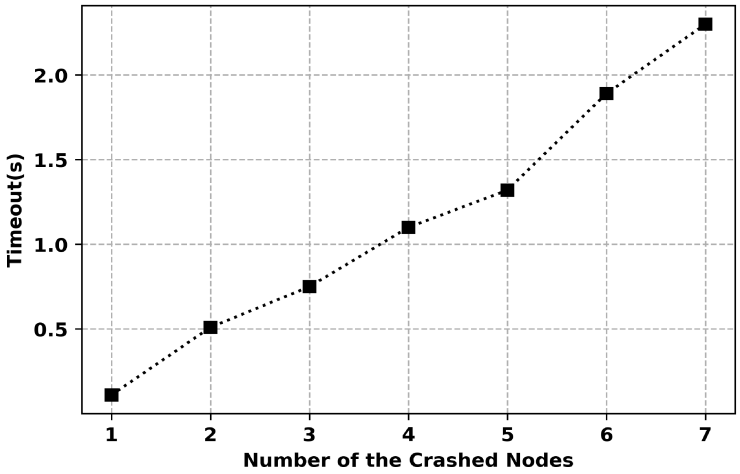
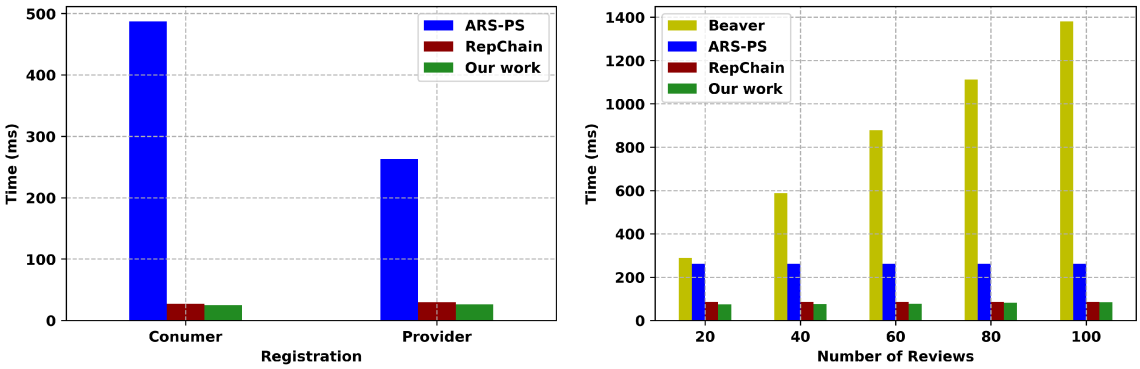


Figure 5.4: Comparison between different timeout intervals with respect to the crashed nodes.

Ethereum blockchain and noticed that there is no notion of gas in Hyperledger Fabric. Figure 5.5 (a) illustrates the data consumer's and data provider's computational cost for registration. We can see that only requires 25 milliseconds for a consumer i and 26.5 milliseconds (ms) for a provider j to register. The computational costs of the review generation and review verification for 20, 40, 60, 80, and 100 are depicted in Figure 5.5 (b) and Figure 5.5 (c), respectively. In comparison to existing systems, our system takes much less computational time for review generation and verification. Our system generated 100 reviews in 84.1 ms and verified 100 reviews in 364.5 ms. This is possible because Hyperledger Fabric supports parallel transactions, which allow transactions to be executed in parallel (by endorsers node) to boost the throughput. Figure 5.5 (c) shows the computational cost of the review verification for 20, 40, 60, 80, and 100, respectively.

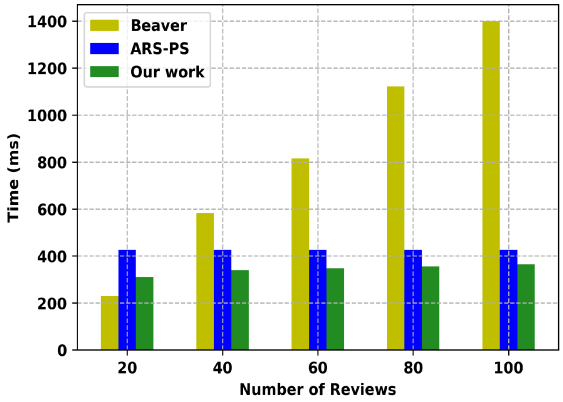
5.4.3 Performance Analysis

We run the Hyperledger Caliper benchmark framework [82] on top of the Hyperledger Fabric to analyze our system performance measurement. We used the *Open*, *Query*, and $Transfer$ workloads for our performance measurement. The *Open* workload measures the writing of the performance of the ledger (creating accounts). The *Query* workload measures the reading performance of the ledger. The $Transfer$ workload measures the performance of the ledger while data trading between accounts. Then,



(a) Computational cost for registration.

(b) Computational cost for review generation.



(c) Computational cost for review verification.

Figure 5.5: Computational costs.

we analyzed our system performance by using the following metrics:

Transaction throughput

This metric measures the average number of transactions that can be written in the ledger $\mathcal{L}$ successfully. The transaction throughput of the blockchain network calculated as follows [67]:

$$Transaction\ throughput = \frac{Total\ transcations}{Total\ time\ in\ seconds} \tag{5.11}$$

Figure 5.6 shows the transaction throughput for 500 transactions with 50, 100, 200, 300, 400tps sending rate. From the results in Figure 5.6, we observe that by increasing

the sending rate, the transaction throughput of the *Query* and *Transfer* workloads increased linearly. The transaction throughput of the *Open* increased linearly until 300tps and slightly increases from 300tps to 400tps.

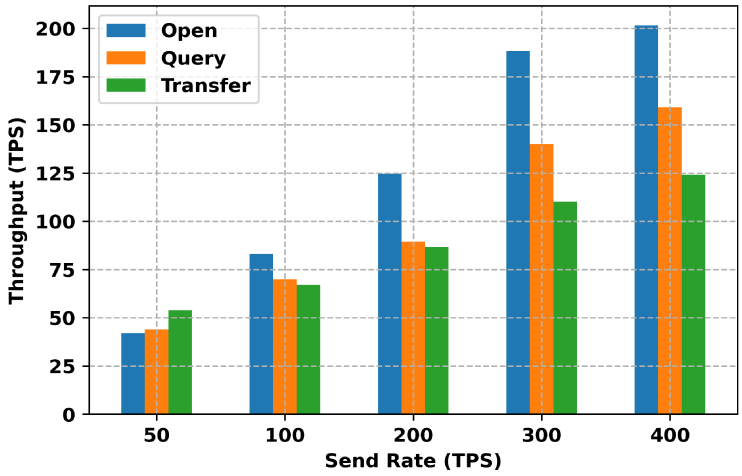


Figure 5.6: Transaction throughput under *Open*, *Query*, and *Transfer* workloads with varied sending rates.

Transaction latency

This metric measures the amount of time it takes from the time a transaction is submitted to the time it is added to the $\mathcal{L}$. Figure 5.7 indicates the average latency for 500 transactions with 50, 100, 200, 400tps sending rate. From Figure 5.7, we can see that by increasing the sending rate, the average latency increases in *Open* workload. We raise the sending rate slightly since higher sending rate would cause a network failure. For *Query* and *Transfer* workloads, from Figure 5.7, we can see that when the sending rate increases, the average latency increases slightly.

Resource consumption

This metric measures how much computing resources, such as CPU and memory, are utilized by the blockchain network during various processes. Figure 5.8 shows the average memory consumption of 500 transactions under *Open* workload. Memory depicts the amount of memory used by the Docker container. The performance

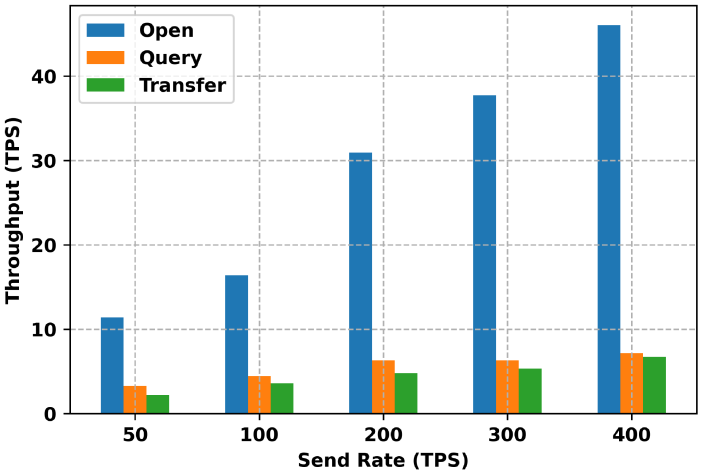


Figure 5.7: Transaction latency under *Open*, *Query*, and *Transfer* workloads with varied sending rates.

analysis shows considerable low memory consumption for 500 transactions. The peer nodes (peer0 and peer1) consume an average of 72.4MB and 58.77MB with 50tps, respectively. By increasing the number of transactions we can see that an average is 88.13MB for peer0.org1.example.com and 71.75MB for peer0.org1.example.com with 400tps, which is significantly low. The orderer.example.com consumer 35.21MB with 50tps and 53.04MB with 400tps. Compared to other orderers, orderer.example.com consumes slightly more memory since it is the leader. With 50tps and 400tps, CA (ca.org1) consumes 25.87MB and 40.88MB, respectively.

Figure 5.8 shows the average CPU consumption of 500 transactions under *Open* workload. CPU displays the amount of CPU used by the docker containers during the test round. CPU(AVG) measures the average resources spent on all transactions. The peer0.org1.example.com node consume an average of 6.4%, 7.6%, 8.8%, 9.7%, 11.13% CPU with 50, 100, 300, 400tps, respectively. The leader node consumes 3.21%, 4.59%, 5.20%, 6.61%, 7.44% CPU with 50, 100, 300, 400tps, respectively. The CA node consumes 2.98%, 3.98%, 5.12%, 6.44%, 7.98% CPU with 50, 100, 300, 400tps, respectively. The performance analysis shows significantly low CPU consumption for 500 transactions.

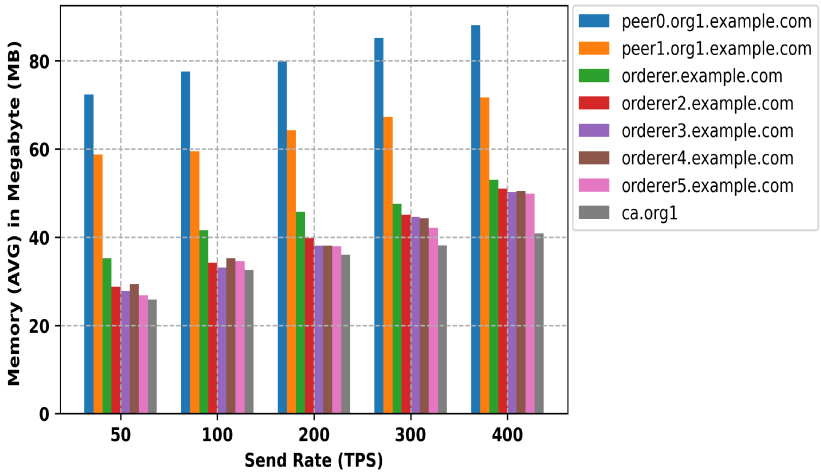


Figure 5.8: Memory consumption for *Open* workload with 500 transactions.

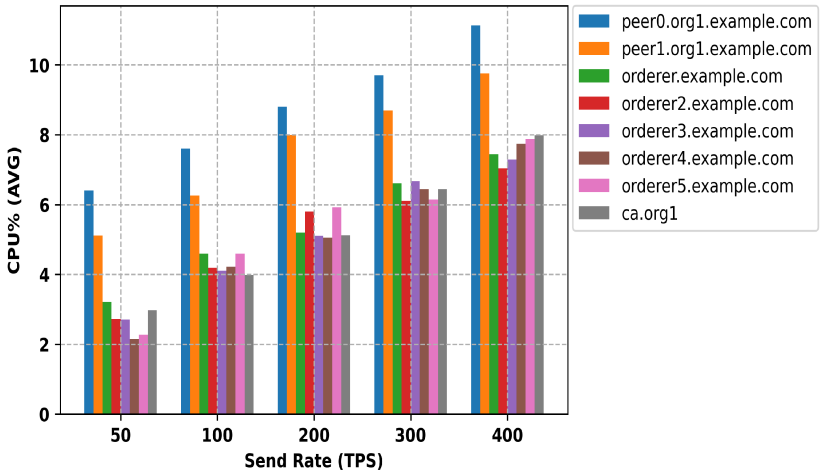


Figure 5.9: CPU consumption for *Open* workload with 500 transactions.

5.5 Summary

This chapter proposed a blockchain-based reputation system for the IIoT data ecosystem for Industry 4.0 based on the Raft consensus mechanism. We further improve the Raft consensus algorithm to avoid SPOF and link failures, which allows the operation to proceed as planned rather than failing. We used blind ECDSA with a

non-interactive ZKP technique to provide confidentiality and anonymity in the reputation system for the IIoT data ecosystem. Furthermore, we propose a reputation computation model which computes the review score of data consumer has upon data provider based on their recent transaction. The experimental results show that the proposed system reduces the computational cost significantly in user registration, review generation, and review verification phases compared to recent studies.

Chapter 6

Blockchain-based Reputation System For IoT Data Ecosystem: A Utility Maximization Approach

In this chapter, we introduce a utility maximization approach for blockchain-based reputation system for IoT data ecosystem in section 6.1. In section 6.2, a system evaluation is presented. Finally, we summarized this chapter in section 6.3.

6.1 System Model

To provide the reader an idea about the high-level overview of our system, we envision that most marketplaces, such as AWS data exchange, Amazon, eBay, etc., will soon integrate blockchain-based reputation systems into their platforms [83]. In such a case, users will use Web 3.0 applications to process their transactions (reviews, payments, etc.) on those platforms. With the rise of blockchain-based reputation systems, the need for interaction models becomes critical. Thus, we propose a decentralized blockchain-based reputation system for IoT data ecosystem as shown in Figure 6.1.

In this system, the users are the data providers (data seller) and the data consumers (data buyers). A data provider $j \in B$ sells IoT data records, which is generated by IoT devices for a monetary value. Let $\mathcal{D}_j$ be the set of data records, and let d_j be a data package $d_j \in \mathcal{D}_j$. For example, a data package consists of one million climate data records (e.g., temperature, humidity, atmospheric pressure,

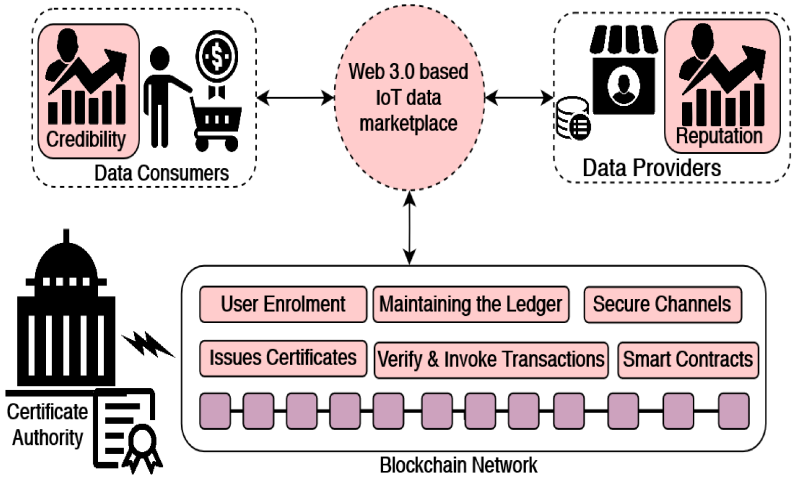


Figure 6.1: High-level architecture of the proposed system.

wind speed records, etc.). The reputation of a data provider j is built on the rating review scores of data consumers. A data consumer $i \in N$ make purchases from data provider j, and leave a rating score r_{ji} on his/her purchase. Both providers and consumers register themselves to the certificate authority (CA). CA is a a a trusted third-party responsible for enrolling and issuing certificates to users (providers and consumers). CA maintains a public ledger $\mathcal{L}$, verifies/invokes transactions, providing secure communication channels for data providers and consumers in the blockchain network. We assume that the data providers compete among their counterparts. This is reasonable because, in real-world scenarios such as Amazon, all vendors (sellers) compete to improve their reputation. As a result, top-rated sellers will have a higher selling rate and make more profit. Data consumers, on the other hand, can boost the credibility of their reviews. Review credibility is critical to ensuring that untrustworthy customers who leave negative reviews (low rating score) do not impact data providers' overall reputation. The smart contract returns the provider and consumer scores over time. Significant notations is summarized in Table 6.1 for the clarity of readers.

6.1.1 Blockchain Registration

This subsection presents how data providers and consumers obtain credentials and certificates to be eligible to trade data in the market. Let $\mathbb{G} = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T)$ be multiplicative cyclic groups with a bilinear paring $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$, g_1 and g_2 are gen-

Table 6.1: Notations.

Notation	Description
CA	Certificate authority
B	Set of data providers $\{1, 2, \ldots, j\}$
N	Set of data consumers $\{1, 2, \ldots, i\}$
λ	Security parameter
CE_i	Certificate for data consumer i
CE_j	Certificate for data provider j
$\mathcal{H}$	Collision-resistant hash function
$\mathcal{D}_j$	Set of data records
d_j	A data package
σ_j	Data provider j's signature
σ_i	Data consumer i's signature
π_{s_j}	Data provider j's ZKP proof
π_{s_j}	Data consumer i's ZKP proof
σ_{CA}	CA's signature
$\mathcal{L}$	Ledger
C_i	Strategy profile of consumer i
A_j	Strategy profile of provider j
U_i	Consumer i' utility (credibility score)
U_j	Provider j' utility (Reputation score)

erators of $\mathbb{G}_1$ and $\mathbb{G}_2$, and a prime order q. Each data provider j and data consumer i request to register in the blockchain network. Provider j and consumer i choose a secret key $s_j \in \mathbb{Z}_q^2$, $s_i \in \mathbb{Z}_q^2$ and CA computes $g_1^{s_j}, \bar{Z}^{s_j} \rightarrow (Y_j, \tilde{Y}_j)$, $g_2^{s_i}, \bar{W}^{s_i} \rightarrow (K_i, \tilde{K}_i)$, respectively. Then, data provider j and consumer i generate a non-interactive ZKP [75] π_{s_j}, π_{s_i}, respectively as follows:

$$s_j : Y_j = g_2^{s_j} \wedge \tilde{Y}_j = \bar{Z}^{s_j} \tag{6.1}$$

$$s_i : K_i = g_1^{s_i} \wedge \tilde{K}_i = \bar{W}^{s_i} \tag{6.2}$$

Then, provider j and consumer i send their public keys, signatures and proofs $(Y_j, \tilde{Y}_j, \sigma_j, \pi_{s_j})$, $(K_i, \tilde{K}_i, \sigma_i, \pi_{s_i})$ to CA, respectively. We used the ECDSA for our signature schema [84]. More formally, the signing operation on a message $m : \{0,1\}^*$ using ECDSA is defined as follows [84]:

- Select s random number $s \in \mathbb{Z}_q^2$ as private key.
- Compute the public key $g_1^{s_1} = r_x$.
- Compute $R = r_x \mod q$.
- Compute $\bar{m} = \mathcal{H}(m)$.
- Compute $S \leftarrow a^{-1}.(\bar{m} + r.x) \mod q$.
- Signature output $\sigma = (R, S)$.

Afterward, CA checks the validity of proofs π_{s_j}, π_{s_i}, and $e(Yj, \bar{Z}) \stackrel{?}{=} e(g_1, \tilde{Y_j})$, $e(K_i, \tilde{W}) \stackrel{?}{=} e(g_2, \tilde{K_i})$ for provider j and consumer i, respectively. CA ignores when the proofs are invalid. Otherwise, CA generates and signs σ_{CA} the certificates CE_j and CE_i for provider j and consumer i, respectively. Finally, the CA stores the information in the blockchain, and sends the certificate CE_j, and CE_i to data provider and data consumer, respectively. Algorithm 4 describes the registration process for providers and consumers on the blockchain.

Algorithm 4 Blockchain Registration

Input: Public parameters $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, q, e, g_1, g_2, \bar{Z}, \bar{W})$

Output: Certificates CE_j, CE_i, ledger $\mathcal{L}$

1: $s_j \in \mathbb{Z}_q^2 \rightarrow g_1^{s_j}, \bar{Z}^{s_j} \rightarrow (Y_j, \tilde{Y_j})$ //Provider j's key generation
2: $s_i \in \mathbb{Z}_q^2 \rightarrow g_2^{s_i}, \bar{W}^{s_i} \rightarrow (K_i, \tilde{K_i})$ //Consumer i's key generation
3: $\pi_{s_j} \equiv s_j : Y_j = g_2^{s_j} \wedge \tilde{Y_j} = \bar{Z}^{s_j}$ //Provider j's ZKP
4: $\pi_{s_i} \equiv s_i : K_i = g_1^{s_i} \wedge \tilde{K_i} = \bar{W}^{s_i}$ //Consumer i's ZKP
5: $e(Yj, \bar{Z}) \stackrel{?}{=} e(g_1, \tilde{Y_j})$ //Proof validation
6: $e(K_i, \tilde{W}) \stackrel{?}{=} e(g_2, \tilde{K_i})$ //Proof validation
7: Verify the σ_j, and σ_i digital signatures
8: $CE_j \leftarrow \sigma_{CA}(Y_j, \tilde{Y_j}, \sigma_j, \pi_{s_j})$ //Issuing certificate
9: $CE_i \leftarrow \sigma_{CA}(K_i, \tilde{K_i}, \sigma_i, \pi_{s_i})$ //Issuing certificate
10: CA adds $CE_j, CE_i \rightarrow \mathcal{L}$ //Storing certificates in ledger

6.1.2 Data Providers Utility

In this subsection, we formulate the utility function of data providers. Each data provider j chooses a strategy (action) $\vec{a_j} \in A_j$ to maximize its reputation score and to increase their competitive advantage. The reputation score is a rating value (score) given by the data consumer after finishing the transaction. In addition, each provider j has a utility function $U_j : A_j \rightarrow \mathbb{R}$, which maps every strategy $\vec{a_j} = [a_1, a_2, a_3, a_4, a_5]$ into a utility for data provider j. Each strategy consists of different factors such as product discount a_1, product quality a_2, product awareness a_3, service quality a_4, and product diversity a_5 [85]. The data providers compete with each other by choosing the strategy that maximizes their reputation and attract more consumers to their products. We drop the vector notation ($\rightarrow$) for the simplicity. Since the reputation score of a provider j depends on the rating score given by consumer i. Given a strategy choice a_j by any provider $j \in B$, the utility function is as follow:

$$U_j(a_j^t, a_{-j}^t) = \sum_t \left(r_{ji}^t(a_j^t, a_{-j}^t) \times c_i \right) \times \sqrt{\bar{r}_{jy}^t(a_j^t, a_{-j}^t)} \tag{6.3}$$

where r_{ij} is current rating score ($0 \leq r_{ij} \leq 1$) for data package purchased. $\overrightarrow{a_{-j}} = [a_1, a_2, a_3, a_4, a_5]$ is the strategies of other data providers. $\bar{r}_{jy}$ indicates a indirect rating score of other entities' $y \in Y$ upon target data provider j. The indirect rating score function is:

$$\bar{r}_{jy} = \sum_{y=1}^{Y} \frac{p_{jy}}{p_{jy} + n_{jy}} \tag{6.4}$$

where p_{jy}, denotes positive feedback of entity y ($0.5 \leq p_{jy} < 1$), and n_{jy} denotes as negative feedback ($0 \leq n_{jy} \leq 0.5$). We consider indirect rating score $\bar{r}_{jy}$, when the data consumer does not have previous experience (for the first transaction).

6.1.3 Data Consumers Utility

Review credibility measures the degree of accuracy of the rating score that consumer i provides to provider j. Generally, trustworthy consumers provide accurate ratings while untrustworthy consumers provide inaccurate ratings. The objective of each data consumer j is to maximize its credibility c_i^t to increase the impact of their rating score on the reputation of data provider i. We define a strategy profile $\vec{c_i} = [c_1, c_2, c_3] \in C_i$ for data consumers, which maps the strategy to the credibility score.

Each strategy consists of multiple factors such as credibility accuracy c_1, credibility completeness c_2, and credibility timeliness c_3 [86]. These factors are controlled, and assessed by CA. In addition, each consumer i has a utility function $U_i : C_i \rightarrow \mathbb{R}$, which maps every strategy $\vec{c_i}$ into a payoff. The utility function of consumer i is modeled as follows:

$$U_i(c_i^t, c_{-i}^t) = \alpha_{ji} \times \frac{c_i^t}{\sum_{i=1}^{N} c_{-i}^t} \tag{6.5}$$

where α_{ji} represents the provider's assessment of the consumer's credibility. In other words, how credible is the consumer's review from the provider perspective. The assessment of α_{ji} may take values such as low ($0 \leq \alpha_{ji} \leq 0.4$), medium ($0.4 \leq \alpha_{ji} \leq 0.6$), and high ($0.6 \leq \alpha_{ji} \leq 1$). The aim of each data consumer i is to maximize its credibility c_i^t to increase the impact of the his/he rating score on the reputation of data provider j.

6.1.4 Adaptive Learning Mechanism

In this subsection, we model the interaction between the data providers and the data consumers as an adaptive learning mechanism. Data providers can learn and update their strategies to maximize their benefits over time. Meanwhile, data consumers can update their strategies to increase their credibility scores. The main goal of the design is to maximize the social welfare between each data provider and the data consumer as follows:

$$U_{sw} = \max\{U_j + U_i\} \tag{6.6}$$

We propose an adaptive learning algorithm that is aware of utilities and the occurrence of strategies for data providers and data consumers, as shown in Algorithm 5. We calculate the number of occurrences of strategies and utilities at each time for data providers and data consumers, respectively as follows:

$$\bar{O}_j = \sum_{j=1}^{B} h(a_j^t, a_{-j}^t) + \sum_{j=1}^{B} U_j(a_j^t, a_{-j}^t) \tag{6.7}$$

$$\bar{O}_i = \sum_{i=1}^{N} h(c_i^t, c_{-i}^t) + \sum_{i=1}^{N} U_i(c_j^t, c_{-i}^t) \tag{6.8}$$

$h(.)$ is number of occurrences of strategies for data providers and data consumers. $\bar{O}_j$ and $\bar{O}_i$ represent the number of occurrences of strategies plus the utilities at the each time. These functions assist data providers and data consumers in identifying which strategies are frequently utilized alongside the utilities for these actions.

Algorithm 5 Adaptive Learning Algorithm

Input: A_j, C_i

Output: $\bar{O}_j, \bar{O}_i$

for each B and N **do**
    Calculate the utility of each provider $U_j(a_j^t, a_{-j}^t)$
    Calculate the utility of each consumer $U_i(c_i^t, c_{-i}^t)$
end for
Smart contract returns the review and credibility scores
Scores adds to ledger $\mathcal{L}$
for all $j \in B$ **do**
    Calculate $\bar{O}_j \leftarrow \sum_{j=1}^{B} h(.) + \sum_{j=1}^{B} U_j(.)$
    Observe the result of $\bar{O}_j$
    Update the $\vec{a_j} \in A_j$
end for
for all $j \in B$ **do**
    Calculate $\bar{O}_i \leftarrow \sum_{i=1}^{N} h(.) + \sum_{i=1}^{N} U_i(.)$
    Observe the result of $\bar{O}_i$
    Update the $\vec{c_j} \in C_i$
end for

6.2 Implementation and Experimental Results

In our experiments, all the results were conducted using Ubuntu 20.04.4 LTS, 4.70 GHz, Intel Core i7 with 16 GB RAM. We implemented our blockchain netwrok on the Hyperledger Fabric. On top of that, we used Hyperledger Caliper for the performance analysis of the proposed system. We used The Raft consensus mechanism due to its fault-tolerance and high performance. Readers may refer to [87] to gain a better understanding of Raft and its components. The security analysis of non-interactive ZKP is discussed in [75], which satisfies the completeness and soundness properties.

ECDSA can withstand multiple attacks on the elliptic curve, hash function, and other attacks discussed in [84]. The security analysis of Hyperledger Fabric is discussed in [88], which satisfies the accountability, fairness, completeness, and consistency properties. Table 6.2 shows the configuration parameters of our system.

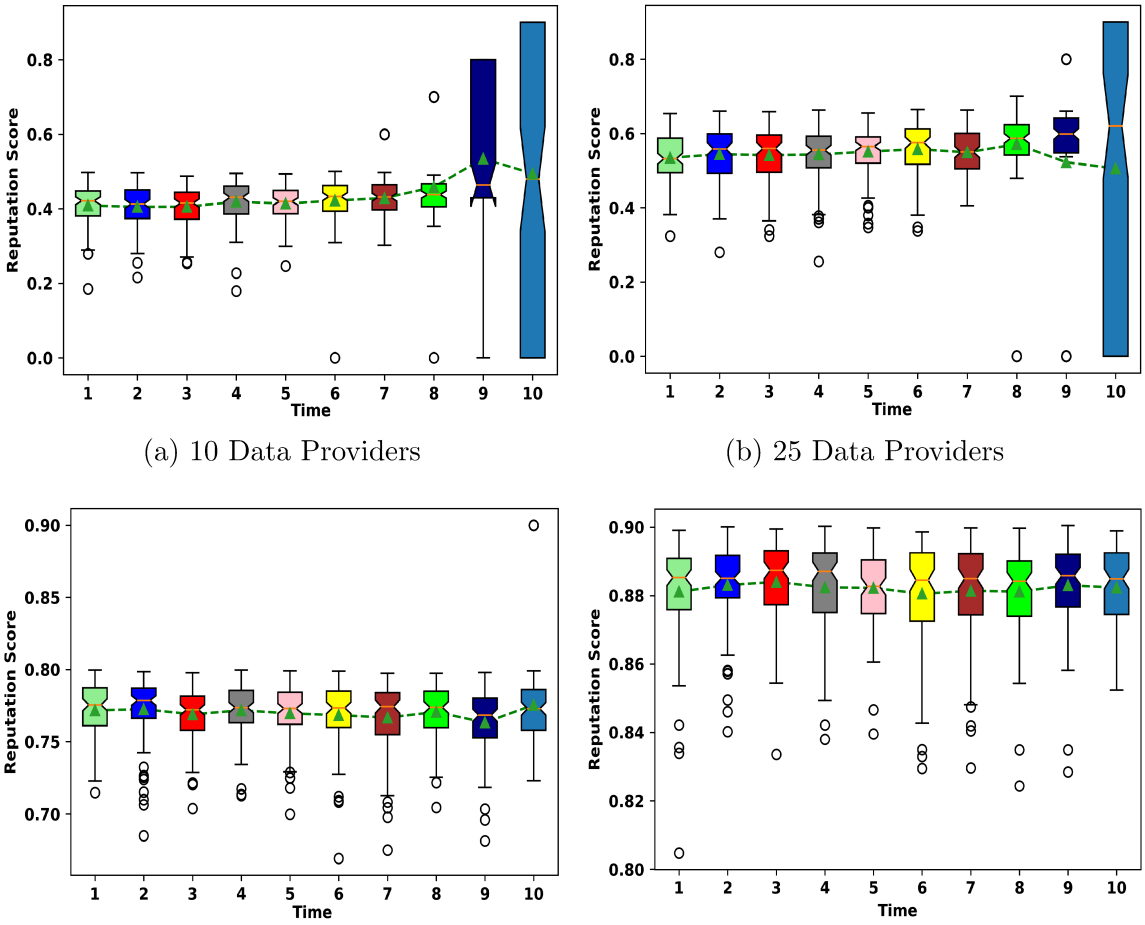
Table 6.2: Experimental setup.

Parameter(s)	Configuration
Transactions per second (TPS)	50, 100, 200, 300, 400, 800 TPS
$\mathcal{H}$ function	SHA256
Total transactions	1000
Elliptic curve	$y^2 \equiv x^3 + ax + b$ over $\mathbb{Z}_q$ (prime256v1)
Consensus	Raft
No. of the organization	One (Org0)
No. of the peer nodes	Two (Peer0, and Peer1) nodes
No. of the orderer	Five (Orderer 0, 1, 2, 3 and 4) nodes
Certificate Authority	One node

6.2.1 Evaluation of Results

We used our adaptive learning Algorithm 5 in repeated interactions between data providers and data consumers overtime. We selected some reasonable arbitrary values for the first interaction. We consider ten interactions with 100 trails. We used a box-plot presentation to demonstrate the data providers' learning process. The center mark in each box plot represents the median of data provider reputation scores. The upper whiskers depict the highest reputation score and the lowest whiskers show the reputation score at each interaction. Outliers are denoted by a ($\circ$) symbol. Figure 6.2 shows the adaptive learning process among 10, 25, 50, and 100 data providers. From Figure 6.2, we can see that after each repeated interactions, data providers update their strategies (increasing their reputation score) based on the previous interaction's outcome and observation of other providers $j \in B$. In Figure 6.2 (a), we can see the reputation score between 10 buyers is between 0.35 to 0.45. In Figure 6.2 (b), the reputation score rises gradually as the number of data consumers rises. In Figure 6.2 (c), and Figure 6.2 (d) reputation score reaches to a range between 0.75 to 0.80 with 50 data providers and reaches between 0.87 to 0.89 with 100 data providers, respectively. We can see that as the number of data suppliers grows, so do the reputation scores. With increased data providers, the number of strategies

will be increased significantly. As a result, data providers will better understand the outcomes and be able to adjust and reach an optimal strategy quickly.



(a) 10 Data Providers

(b) 25 Data Providers

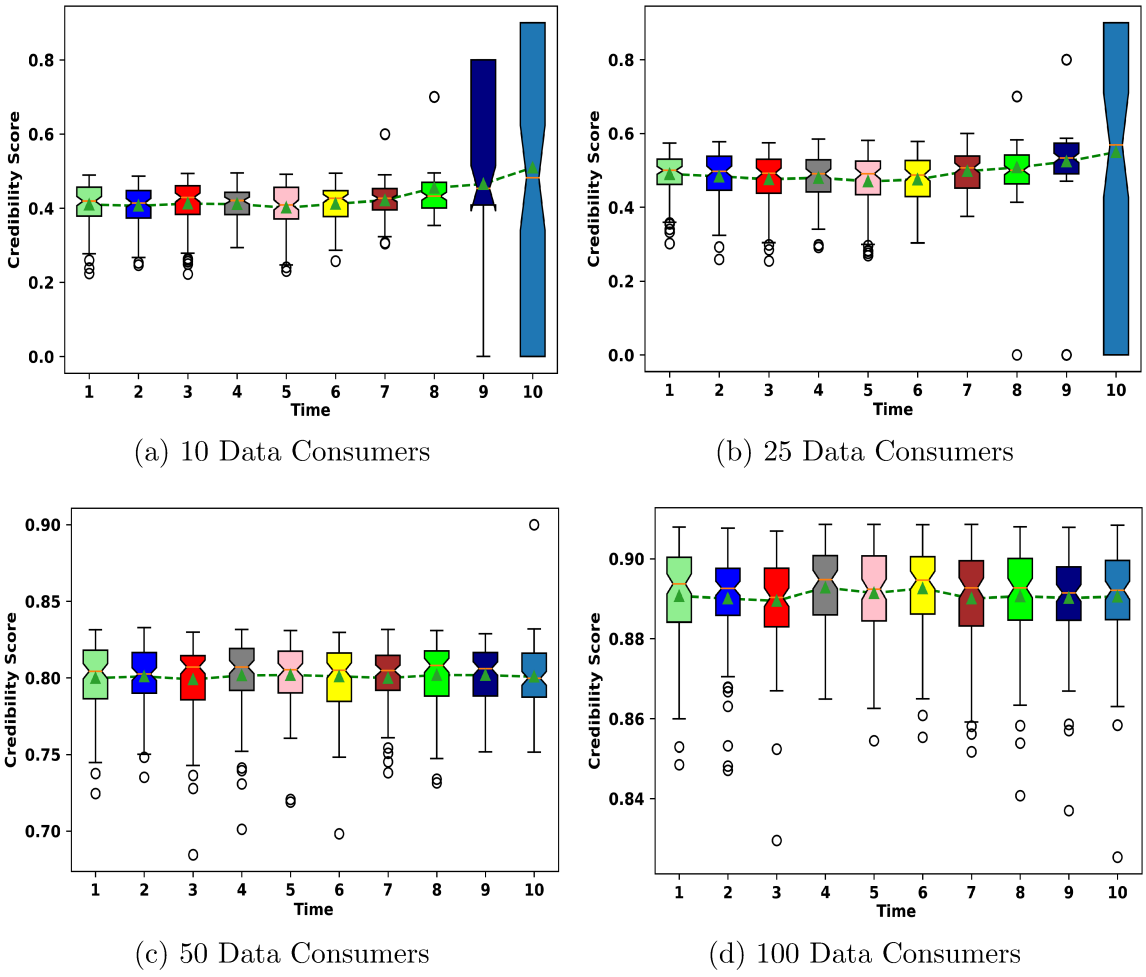
(c) 50 Data Providers

(d) 100 Data Providers

Figure 6.2: Data provider's utility in repeated interactions overtime.

We applied the same learning principle and settings for the of data consumers $i \in N$. Figure 6.3 shows the utilities among 10, 25, 50, and 100 data consumers. From Figure 6.3, we can see that after each period, data consumers can maximizing their credibility score based on the previous interaction's outcome and observation of other consumers $i \in N$. Similarly to data providers, we can see that as the number of data consumers grows, the credibility score increase as well. In Figure 6.3 (a), we can see that in ten repeated interactions, the highest credibility scores are laid down between 0.25 to 0.45. In Figure 6.3 (b), the credibility score increases slightly by

increasing the number of data consumers to 25. In Figure 6.3 (c), and Figure 6.3 (d) credibility score reaches between 0.75 to 0.85 with 50 data consumers and reaches between 0.85 to 0.90 with 100 data consumers, respectively.



(a) 10 Data Consumers

(b) 25 Data Consumers

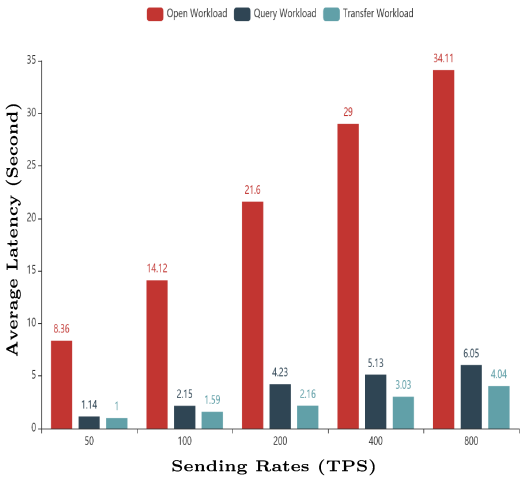
(c) 50 Data Consumers

(d) 100 Data Consumers

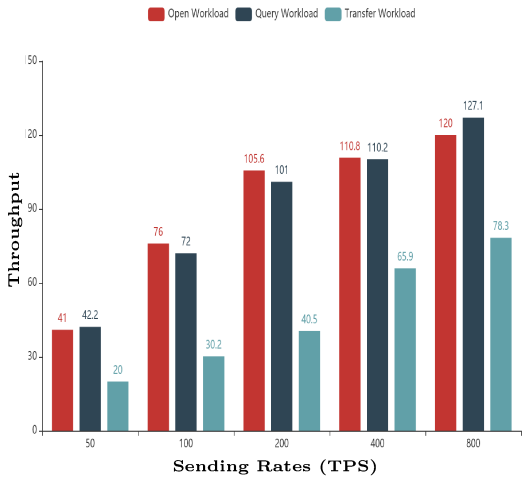
Figure 6.3: Data consumer's utility in repeated interactions overtime.

6.2.2 Performance Analysis

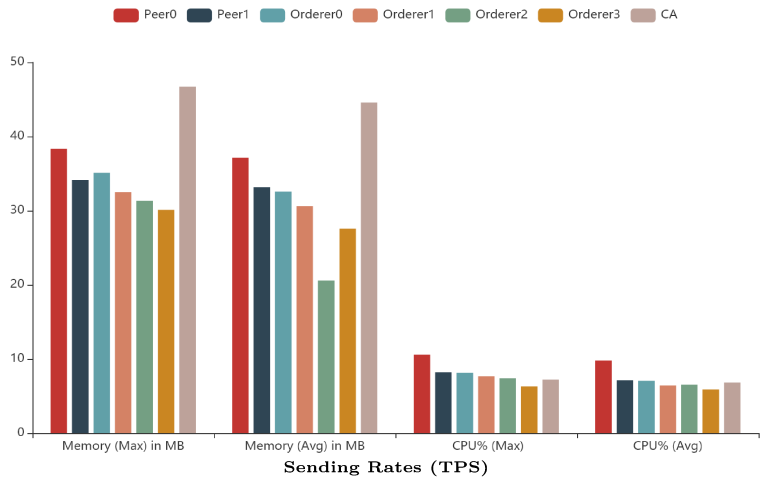
We measured our system performance using three metrics [65, 67]: transaction throughput, transaction latency, and resource consumption under open, query, and transfer workloads. Figure 6.4 (a) and (b) depict the performance analysis of average latency and transaction throughput metrics under varied workloads with different



(a) Average Latency



(b) Transaction Throughput



(c) Memory and CPU Consumption

Figure 6.4: Performance analysis.

sending rates, respectively. We can observe that in Figure 6.4 (a), the average latency is slightly increases with the increase of sending rates under the open workload. Furthermore, we can see that the average latency under query and transfer workloads rise lightly when the TPS increases. Figure 6.4 (b) shows that transaction throughput under open and query workloads are nearly close, whereas transaction throughput increases somewhat under transfer workload. Transaction throughput, respectively, reaches 120, 127.1, and 78.3 in 800 TPS under open, query, and transfer

workloads. Figure 6.4 (c) shows memory and CPU consumption in 800 TPS. In Figure 6.4 (c), we can see that the maximum memory that CA consumes is 46.71MB, and the average memory is 44.59MB. The peer0 and peer1 nodes consume 37.12MB and 33.15MB memory on average, respectively. The orderer nodes, respectively (orderer0, orderer1, orderer2, orderer3), consume 32.59MB, 30.59, 20.59, and 27.59. Peer nodes utilize somewhat more CPU than other nodes due to the fact that they host ledgers and smart contracts. Peer0 and peer1 use 9.81%, and 7.13% of CPU on average. Orderer0 uses 7.07%, orderer1 uses 6.43%, orderer2 uses 6.53%, orderer3 uses 5.91% of CPU on average. CA uses 6.83% of CPU on average. The performance analysis results show that our blockchain network consumes low resources in terms of CPU and memory.

6.3 Summary

With the rapid expansion of online data marketplaces, the necessity for a reputation system is becoming increasingly important. In recent years, researchers developed blockchain-based reputation systems to overcome the challenges of centralization, review anonymity, and cyber attacks. Unfortunately, they do not concentrate on the strategic interaction models between data consumers and providers in blockchain-based reputation systems. This chapter proposed a utility maximization approach between data consumers and providers to address the gap. We proposed an adaptive learning mechanism in which data providers seek to maximize their reputation scores by learning from the reviews' outcomes and adjusting their strategies. Meanwhile, data consumers seek to increase their credibility scores by updating their review accuracy, completeness, and timeliness strategies. We implemented our system on the Hyperledger Fabric blockchain. We analyzed and measured our system performance under transaction throughput, transaction latency, memory, and CPU metrics.

Chapter 7

Conclusions and Future Works

In this chapter, the conclusions of this research are drawn and the future works are summarized.

7.1 Conclusions

First, the research studies the trustful and transparent system for IoT data market place in chapter 3. Specifically, we proposed a trustful data trading framework using the game theory approach in an infinitely repeated horizon to enable secure and efficient data trading between buyers and sellers. To model the data market, this research proposed a non-cooperative infinity repeated game model between rational data buyers. In each stage of the game buyer participates, holds a bid for a traded amount of records, and seeks to maximize his expected utility through learning from the outcome of previous stages considering discounted rate for the future utility. We proved NE and uniqueness of our model which derived theoretically for one-shot (stage game), finite and infinite horizon, respectively. In addition, this model imposes a penalty on those buyers who do not have a good reputation and decrease their chance of winning to preserve the data owner's privacy.

Second, this research investigates the performance analysis of blockchain-based data trading systems in chapter 4. The radically increasing amount of data generated by IoT devices has led to the emergence of a new data trading market. In recent years, blockchain technology has been used to build private and secure modern data trading systems. Most of the research in this area describes models of blockchain-based trading systems for ensuring transparency, encryption, privacy, and security

of traded data. However, the literature lacks studies about performance models to demonstrate their usability in a real data trading market. This research addresses this shortcoming by providing a detailed performance analysis of blockchain-based data trading systems to ensure their feasibility, usability, and data availability, which are important aspects for their functional deployment in real data trading markets. Through experimental results, we evaluate the performance of the blockchain-based data trading system using Hyperledger Fabric and its benchmark framework Hyperledger caliper.

Third, this research investigates a blockchain-based reputation system for IIoT data ecosystem in chapter 5. IIoT devices generate and collect massive amounts of industrial data. Monetizing the flood of data generated by the IIoT devices has enabled the creation of the IIoT data ecosystem where individuals and businesses may sell and trade data. With the rapid expansion of the online data trading industry, the necessity for a reputation system is becoming increasingly important as more individuals and services connect online. In recent years, researchers have proposed blockchain-based reputation systems as a means of offering anonymity, security, transparency, and mutual trust for both providers and customers in Industry 4.0. Unfortunately, they focus on the decentralized reputation system with a single certificate authority, which creates the concern of a SPOF. Moreover, researchers paid little attention to the performance measures of these blockchain-based reputation systems to demonstrate their usability in a real IIoT data ecosystem. This research proposes a robust blockchain-based reputation system capable of avoiding failures by enhancing the Raft consensus mechanism. We provide extensive security analysis and simulation experiments to demonstrate the performance of the blockchain-based reputation system for the IIoT data ecosystem using different metrics, such as transaction throughput, latency, and resource consumption.

Finally, this research proposes a blockchain-based reputation system where market players can maximize their utility while engaged in the IoT data ecosystem in chapter 6. Blockchain technology has been used to build private and secure IoT data markets and trading systems. Reputation of the trading parties is an important attribute that affect their profitability and trading prosperity. However, current reputation systems are prone to malicious manipulation. Such an issue can be avoided by registering all reputation scores in the blockchain which guarantees transparency, encryption, privacy, and security. To this end, this research proposes a blockchain-based reputation system where market players can maximize their utility while engaged in the IoT data ecosystem. We also propose an adaptive learning mechanism that allows the data providers and consumers to enhance their reputation and review

credibility scores. We provide extensive experiments to ensure our system satisfies the feasibility, usability, and availability properties. These properties are essential for functional deployment in real IoT data markets.

7.2 Future Works

This section presents some promising ideas for improving and expanding this research.

- In chapter 3, there could be condition that same data buyer wins every time, because no other buyer has a higher score. This a limitation of our system. We completely understand that this might lead to "starvation" of other buyers who are eager to win deals but unable to reach the highest score. One suggestion would be to gradually increase their score based on the number of attempts that they failed to acquire the data. In other words, if the buyer keeps bidding and not successful, the system, for example, should consider increasing the buyer's score after a number of consecutive attempts. Our plan is develop efficient mechanism to tackle this limitation.
- In chapter 3, if the data buyers have an equally high trust score, our system randomly selects the winner. Although this is not necessary the best approach, it solves the contention such a scenario arise. However, a more suitable solution could be examined such as considering the history of transactions or other reviews that can be accounted. We plan to implement and examine such a approach to evaluate its effectiveness.
- There could be a condition in chapter 4, where data sellers sell the same data many times (e.g., in separate markets) and buyers resale the purchased data. We intend to create systems to prohibit data sellers and buyers from reselling the data.
- Improve System Performance: We plan to develop a more extensive blockchain network with more ordering nodes and peer nodes to reduce latency and increase throughput. We plan to compare our improved raft consensus mechanism to other existing consensus mechanisms such as proof-of-activity, proof-of-work, proof-of-stake, and proof-of-authority in terms of energy, network latency, and consensus time.

- Transaction Scalability: Processing transaction depends on different components such as hardware, block length, docker cluster, nodes, etc. In a real-life system, Bitcoin processes 7 tps and Ethereum 30 tps, whereas centralized databases process around 1,500 tps on average (e.g., visa). Hyperledger Fabric can process more than 3,000 tps because of its consensus mechanism. We plan to improve the transcation scalability by improving the Raft consensus mechanism.

- Additional Commenting: Our blockchain-based reputation system does not have the commenting feature in which consumers can leave comments alongside their review score. We plan to add additional commenting features to our system.

- Reward Mechanism: It is always vital to incentivize data users to post a review. Rewarding consumers encourage them to provide feedback on the product. We intend to develop an efficient and secure rewarding mechanism to encourage data consumers to leave a review score on the purchased data packages.

- Online Blockchain-based Reputation Systems for the Health Sector: People seeking medical advice and services frequently struggle to acquire credible information regarding the quality and competence of health-care providers such as doctor and hospital. We intend to apply our reputation systems and integrated into the health sector.

References

[1] Sujit Bebortta, Amit Kumar Singh, Bibudhendu Pati, and Dilip Senapati. A robust energy optimization and data reduction scheme for iot based indoor environments using local processing framework. *Journal of Network and Systems Management*, 29(1):1–28, 2021.

[2] The growth in connected iot devices is expected to generate 79.4zb of data in 2025, according to a new idc forecast. Available online: https://www.idc.com/getdoc.jsp?containerId=prUS45213219. (Accessed: 2020-12-14).

[3] Yan Li, Lingyan Li, Yanqi Zhao, Nadra Guizani, Yong Yu, and Xiaojiang Du. Toward decentralized fair data trading based on blockchain. *IEEE Network*, 2020.

[4] Albert Opher, Alex Chou, Andrew Onda, and Krishna Sounderrajan. The rise of the data economy: driving value through internet of things data monetization. *IBM Corporation: Somers, NY, USA*, 2016.

[5] Raul Castro Fernandez, Pranav Subramaniam, and Michael J Franklin. Data market platforms: Trading data assets to solve data problems [vision paper]. *arXiv*, pages arXiv–2002, 2020.

[6] David Lopez and Bilal Farooq. A multi-layered blockchain framework for smart mobility data-markets. *Transportation Research Part C: Emerging Technologies*, 111:588–615, 2020.

[7] Yang Xiao, Ning Zhang, Jin Li, Wenjing Lou, and Y Thomas Hou. Privacyguard: Enforcing private data usage control with blockchain and attested off-chain contract execution. In *European Symposium on Research in Computer Security*, pages 610–629. Springer, 2020.

[8] Md Moniruzzaman, Seyednima Khezr, Abdulsalam Yassine, and Rachid Benlamri. Blockchain for smart homes: Review of current trends and research challenges. *Computers & Electrical Engineering*, 83:106585, 2020.

[9] Karl Wüst and Arthur Gervais. Do you need a blockchain? In *2018 Crypto Valley Conference on Blockchain Technology (CVCBT)*, pages 45–54. IEEE, 2018.

[10] Cisco Global Cloud Index. Forecast and methodology, 2016–2021 white paper. *Updated: February*, 1, 2018.

[11] David Y Aharon and Mahmoud Qadan. When do retail investors pay attention to their trading platforms? *The North American Journal of Economics and Finance*, page 101209, 2020.

[12] Lindah Kotut, Timothy L Stelter, Michael Horning, and D Scott McCrickard. Willing buyer, willing seller: Personal data trade as a service. In *Companion of the 2020 ACM International Conference on Supporting Group Work*, pages 59–68, 2020.

[13] Abdulsalam Yassine, Ali Asghar Nazari Shirehjini, and Shervin Shirmohammadi. Smart meters big data: Game theoretic model for fair data sharing in deregulated smart grids. *IEEE access*, 3:2743–2754, 2015.

[14] Big Data Market by Component, Deployment Mode, Organization Size, Business Function (Operations, Finance, and Marketing and Sales), Industry Vertical (BFSI, Manufacturing, and Healthcare and Life Sciences), and Region - Global Forecast to 2025 . Available online: https://www.marketsandmarkets.com/Market-Reports/big-data-market-1068.html. (Accessed: 2020-12-12).

[15] JW MICHAEL, ALAN COHN, and JARED R BUTCHER. Blockchain technology. *The Journal*, 2018.

[16] Andrea Gaggioli. Blockchain technology: Living in a decentralized everything. *Cyberpsychology, Behavior, and Social Networking*, 21(1):65–66, 2018.

[17] Gavin Wood et al. Ethereum: A secure decentralised generalised transaction ledger. *Ethereum project yellow paper*, 151(2014):1–32, 2014.

[18] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. Technical report, Manubot, 2019.

[19] Satoshi Nakamoto et al. Bitcoin: A peer-to-peer electronic cash system. 2008.

[20] Iuon-Chang Lin and Tzu-Chun Liao. A survey of blockchain security issues and challenges. *IJ Network Security*, 19(5):653–659, 2017.

[21] Merlinda Andoni, Valentin Robu, David Flynn, Simone Abram, Dale Geach, David Jenkins, Peter McCallum, and Andrew Peacock. Blockchain technology in the energy sector: A systematic review of challenges and opportunities. *Renewable and Sustainable Energy Reviews*, 100:143–174, 2019.

[22] Xiaoqi Li, Peng Jiang, Ting Chen, Xiapu Luo, and Qiaoyan Wen. A survey on the security of blockchain systems. *Future Generation Computer Systems*, 107:841–853, 2020.

[23] Vitalik Buterin et al. A next-generation smart contract and decentralized application platform. *white paper*, 3(37), 2014.

[24] Seyednima Khezr, Rachid Benlamri, and Abdulsalam Yassine. Blockchain-based model for sharing activities of daily living in healthcare applications. In *2020 IEEE Intl Conf on Dependable, Autonomic and Secure Computing, Intl Conf on Pervasive Intelligence and Computing, Intl Conf on Cloud and Big Data Computing, Intl Conf on Cyber Science and Technology Congress (DASC/PiCom/CBDCom/CyberSciTech)*, pages 627–633. IEEE, 2020.

[25] Hyeontaek Oh, Sangdon Park, Gyu Myoung Lee, Jun Kyun Choi, and Sungkee Noh. Competitive data trading model with privacy valuation for multiple stakeholders in iot data markets. *IEEE Internet of Things Journal*, 7(4):3623–3639, 2020.

[26] Hyeontaek Oh, Sangdon Park, Gyu Myoung Lee, Hwanjo Heo, and Jun Kyun Choi. Personal data trading scheme for data brokers in iot data marketplaces. *IEEE Access*, 7:40120–40132, 2019.

[27] Ling Tian, Jiaxin Li, Wei Li, Balasubramaniam Ramesh, and Zhipeng Cai. Optimal contract-based mechanisms for online data trading markets. *IEEE Internet of Things Journal*, 6(5):7800–7810, 2019.

[28] Xuanyu Cao, Yan Chen, and KJ Ray Liu. Data trading with multiple owners, collectors, and users: An iterative auction mechanism. *IEEE Transactions on Signal and Information Processing over Networks*, 3(2):268–281, 2017.

[29] Rashid Hussain Khokhar, Farkhund Iqbal, Benjamin CM Fung, and Jamal Bentahar. Enabling secure trustworthiness assessment and privacy protection in integrating data for trading person-specific information. *IEEE Transactions on Engineering Management*, 2020.

[30] Kang Liu, Xiaoyu Qiu, Wuhui Chen, Xu Chen, and Zibin Zheng. Optimal pricing mechanism for data market in blockchain-enhanced internet of things. *IEEE Internet of Things Journal*, 6(6):9748–9761, 2019.

[31] Dingjie Sheng, Mingjun Xiao, An Liu, Xiang Zou, Baoyi An, and Sheng Zhang. Cpchain: A copyright-preserving crowdsourcing data trading framework based on blockchain. In *2020 29th International Conference on Computer Communications and Networks (ICCCN)*, pages 1–9. IEEE, 2020.

[32] Weiqi Dai, Chunkai Dai, Kim-Kwang Raymond Choo, Changze Cui, Deiqing Zou, and Hai Jin. Sdte: A secure blockchain-based data trading ecosystem. *IEEE Transactions on Information Forensics and Security*, 15:725–737, 2019.

[33] Wei Xiong and Li Xiong. Smart contract based data trading mode using blockchain and machine learning. *IEEE Access*, 7:102331–102344, 2019.

[34] Hien Thi Thu Truong, Miguel Almeida, Ghassan Karame, and Claudio Soriente. Towards secure and decentralized sharing of iot data. In *2019 IEEE International Conference on Blockchain (Blockchain)*, pages 176–183. IEEE, 2019.

[35] Kyle Soska, Albert Kwon, Nicolas Christin, and Srinivas Devadas. Beaver: A decentralized anonymous marketplace with secure reputation. *IACR Cryptol. ePrint Arch.*, 2016:464, 2016.

[36] Dongxiao Liu, Amal Alahmadi, Jianbing Ni, Xiaodong Lin, and Xuemin Shen. Anonymous reputation system for iiot-enabled retail marketing atop pos blockchain. *IEEE Transactions on Industrial Informatics*, 15(6):3527–3537, 2019.

[37] Nguyen Truong, Gyu Myoung Lee, Kai Sun, Florian Guitton, and Yike Guo. A blockchain-based trust system for decentralised applications: When trustless needs trust. *arXiv preprint arXiv:2101.10920*, 2021.

[38] Ponlawat Weerapanpisit, Sergio Trilles, Joaquín Huerta, and Marco Painho. A decentralised location-based reputation management system in the iot using blockchain. *IEEE Internet of Things Journal*, 2022.

[39] Goncalo Sousa Mendes, Daniel Chen, Bruno MC Silva, Carlos Serrao, and Joao Casal. A novel reputation system for mobile app stores using blockchain. *Computer*, 54(2):39–49, 2021.

[40] Soojin Lee and Seung-Hyun Seo. Design of a two layered blockchain-based reputation system in vehicular networks. *IEEE Transactions on Vehicular Technology*, 71(2):1209–1223, 2021.

[41] Tonghe Wang, Jian Guo, Songpu Ai, and Junwei Cao. Rbt: A distributed reputation system for blockchain-based peer-to-peer energy trading with fairness consideration. *Applied Energy*, 295:117056, 2021.

[42] Zhili Zhou, Meimin Wang, Ching-Nung Yang, Zhangjie Fu, Xingming Sun, and QM Jonathan Wu. Blockchain-based decentralized reputation system in e-commerce environment. *Future Generation Computer Systems*, 124:155–167, 2021.

[43] Meng Li, Liehuang Zhu, Zijian Zhang, Chhagan Lal, Mauro Conti, and Mamoun Alazab. Anonymous and verifiable reputation system for e-commerce platforms based on blockchain. *IEEE Transactions on Network and Service Management*, 18(4):4434–4449, 2021.

[44] Stephen Boyd, Stephen P Boyd, and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[45] Dinh Thai Hoang, Xiao Lu, Dusit Niyato, Ping Wang, Dong In Kim, and Zhu Han. Applications of repeated games in wireless networks: A survey. *IEEE Communications Surveys & Tutorials*, 17(4):2102–2135, 2015.

[46] David K Levine. Learning in games, 2001.

[47] Drew Fudenberg and David Levine. Learning in games. *European economic review*, 42(3-5):631–639, 1998.

[48] Kevin Leyton-Brown and Yoav Shoham. Essentials of game theory: A concise multidisciplinary introduction. *Synthesis lectures on artificial intelligence and machine learning*, 2(1):1–88, 2008.

[49] Partha Dasgupta and Eric Maskin. The existence of equilibrium in discontinuous economic games, i: Theory. *The Review of economic studies*, 53(1):1–26, 1986.

[50] Yunpeng Wang, Walid Saad, Zhu Han, H Vincent Poor, and Tamer Başar. A game-theoretic approach to energy trading in the smart grid. *IEEE Transactions on Smart Grid*, 5(3):1439–1450, 2014.

[51] Roy D. Yates. A framework for uplink power control in cellular radio systems. *IEEE Journal on selected areas in communications*, 13(7):1341–1347, 1995.

[52] Sylvain Sorin. On repeated games with complete information. *Mathematics of Operations Research*, 11(1):147–160, 1986.

[53] Rida Laraki, Jérôme Renault, and Sylvain Sorin. *Mathematical foundations of game theory*. Springer, 2019.

[54] Anupam Das and Mohammad Mahfuzul Islam. Securedtrust: a dynamic trust computation model for secured communication in multiagent systems. *IEEE Transactions on dependable and secure computing*, 9(2):261–274, 2011.

[55] Jiaxing Li, Jigang Wu, Guiyuan Jiang, and Thambipillai Srikanthan. Blockchain-based public auditing for big data in cloud storage. *Information Processing & Management*, 57(6):102382, 2020.

[56] Chuan Chen, Jiajing Wu, Hui Lin, Wuhui Chen, and Zibin Zheng. A secure and efficient blockchain-based data trading approach for internet of vehicles. *IEEE Transactions on Vehicular Technology*, 68(9):9110–9121, 2019.

[57] Shinsaku Kiyomoto, Mohammad Shahriar Rahman, and Anirban Basu. On blockchain-based anonymized dataset distribution platform. In *2017 IEEE 15th International Conference on Software Engineering Research, Management and Applications (SERA)*, pages 85–92. IEEE, 2017.

[58] Vikas Hassija, Vinay Chamola, Sahil Garg, Nanda Gopala Krishna Dara, Georges Kaddoum, and Dushantha Nalin K Jayakody. A blockchain-based framework for lightweight data sharing and energy trading in v2g network. *IEEE Transactions on Vehicular Technology*, 2020.

[59] Haiying Ma, Elmo X Huang, and Kwok-Yan Lam. Blockchain-based mechanism for fine-grained authorization in data crowdsourcing. *Future Generation Computer Systems*, 106:121–134, 2020.

[60] Huige Li, Haibo Tian, Fangguo Zhang, and Jiejie He. Blockchain-based searchable symmetric encryption scheme. *Computers & Electrical Engineering*, 73:32–45, 2019.

[61] Muzafer H Saračević, Saša Z Adamović, Vladislav A Miškovic, Mohamed Elhoseny, Nemanja D Maček, Mahmoud Mohamed Selim, and K Shankar. Data encryption for internet of things applications based on catalan objects and two combinatorial structures. *IEEE Transactions on Reliability*, 2020.

[62] Bryan Parno, Jon Howell, Craig Gentry, and Mariana Raykova. Pinocchio: Nearly practical verifiable computation. In *2013 IEEE Symposium on Security and Privacy*, pages 238–252. IEEE, 2013.

[63] Yinghui Zhang, Robert H Deng, Jiangang Shu, Kan Yang, and Dong Zheng. Tkse: Trustworthy keyword search over encrypted data with two-side verifiability via blockchain. *IEEE Access*, 6:31077–31087, 2018.

[64] Tamas Blummer, M Sean, and C Cachin. An introduction to hyperledger. *Hyperledger Under Linux Found., White Paper*, 2018.

[65] P.W.D. Charles. A blockchain benchmark framework. 2019. (Accessed: 2020-04-26).

[66] Jack Kelly and William Knottenbelt. The uk-dale dataset, domestic appliance-level electricity demand and whole-house demand from five uk homes. *Scientific data*, 2:150007, 2015.

[67] Hyperledger Performance, Scale Working Group, et al. Hyperledger blockchain performance metrics. *Hyperledger. org*, pages 1–17, 2018.

[68] Saiyu Qi, Youshui Lu, Yuanqing Zheng, Yumo Li, and Xiaofeng Chen. Cpds: enabling compressed and private data sharing for industrial internet of things over blockchain. *IEEE Transactions on Industrial Informatics*, 17(4):2376–2387, 2020.

[69] Qinya Li, Zun Li, Zhenzhe Zheng, Fan Wu, Shaojie Tang, Zhao Zhang, and Guihai Chen. Capitalize your data: Optimal selling mechanisms for iot data exchange. *IEEE Transactions on Mobile Computing*, 2021.

[70] Cheng Huang, Dongxiao Liu, Jianbing Ni, Rongxing Lu, and Xuemin Shen. Achieving accountable and efficient data sharing in industrial internet of things. *IEEE Transactions on Industrial Informatics*, 17(2):1416–1427, 2020.

[71] Chunli Huang, Wenjun Jiang, Jie Wu, and Guojun Wang. Personalized review recommendation based on users' aspect sentiment. *ACM Transactions on Internet Technology (TOIT)*, 20(4):1–26, 2020.

[72] Chenyu Huang, Zeyu Wang, Huangxun Chen, Qiwei Hu, Qian Zhang, Wei Wang, and Xia Guan. Repchain: A reputation-based secure, fast, and high incentive blockchain system via sharding. *IEEE Internet of Things Journal*, 8(6):4291–4304, 2020.

[73] Faiza Loukil, Chirine Ghedira-Guegan, Khouloud Boukadi, Aïcha-Nabila Benharkat, and Elhadj Benkhelifa. Data privacy based on iot device behavior control using blockchain. *ACM Transactions on Internet Technology (TOIT)*, 21(1):1–20, 2021.

[74] Xiaohan Zhang, Xinghua Li, Yinbin Miao, Xizhao Luo, Yunwei Wang, Siqi Ma, and Jian Weng. A data trading scheme with efficient data usage control for industrial iot. *IEEE Transactions on Industrial Informatics*, 2021.

[75] Manuel Blum, Paul Feldman, and Silvio Micali. Non-interactive zero-knowledge and its applications. In *Providing Sound Foundations for Cryptography: On the Work of Shafi Goldwasser and Silvio Micali*, pages 329–349. 2019.

[76] Xun Yi and Kwok-Yan Lam. A new blind ecdsa scheme for bitcoin transaction anonymity. In *Proceedings of the 2019 ACM Asia Conference on Computer and Communications Security*, pages 613–620, 2019.

[77] Matthew Green and Ian Miers. Bolt: Anonymous payment channels for decentralized currencies. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 473–489, 2017.

[78] Linux foundation 2021 hyperledger fabric penetration test. Available online: https://wiki.hyperledger.org/display/fabric/Audits, 2021. (Accessed: 2022-01-23).

[79] Diego Ongaro and John Ousterhout. In search of an understandable consensus algorithm (extended version), 2013.

[80] Hyperledger fabric. Available online: https://github.com/hyperledger/fabric/tree/release-2.2, 2021. (Accessed: 2021-10-26).

[81] Christian Fluri, Darya Melnyk, and Roger Wattenhofer. Improving raft when there are failures. In *2018 Eighth Latin-American Symposium on Dependable Computing (LADC)*, pages 167–170. IEEE, 2018.

[82] Hyperledger caliper. Available online: https://github.com/hyperledger/caliper, 2021. (Accessed: 2021-10-26).

[83] Seyednima Khezr, Abdulsalam Yassine, Rachid Benlamri, and M Shamim Hossain. An edge intelligent blockchain-based reputation system for iiot data ecosystem. *IEEE Transactions on Industrial Informatics*, 2022.

[84] Don Johnson, Alfred Menezes, and Scott Vanstone. The elliptic curve digital signature algorithm (ecdsa). *International journal of information security*, 1(1):36–63, 2001.

[85] Haonan Cheng, Yi-Ting Huang, and Jesheng Huang. The application of dematel-anp in livestream e-commerce based on the research of consumers' shopping motivation. *Scientific Programming*, 2022, 2022.

[86] Marc-Julian Thomas, Bernd W Wirtz, and Jan C Weyerer. Determinants of online review credibility and its impact on consumers'purchase intention. *Journal of Electronic Commerce Research*, 20(1):1–20, 2019.

[87] Diego Ongaro and John Ousterhout. In search of an understandable consensus algorithm. In *2014 USENIX Annual Technical Conference (Usenix ATC 14)*, pages 305–319, 2014.

[88] Mike Graf, Ralf Küsters, and Daniel Rausch. Accountability in a permissioned blockchain: formal analysis of hyperledger fabric. In *2020 IEEE European Symposium on Security and Privacy (EuroS&P)*, pages 236–255. IEEE, 2020.

Author's Publications

Published:

1. Khezr, S., Yassine, A., Benlamri, R., & Hossain, M. S. (2022). An Edge Intelligent Blockchain-based Reputation System for IIoT Data Ecosystem, *IEEE Transactions on Industrial Informatics*

2. Khezr, S., Benlamri, R., & Yassine, A. (2020, August). Blockchain-based Model for Sharing Activities of Daily Living in Healthcare Applications, *18th IEEE Intl Conf on Dependable, Autonomic & Secure Computing.*

3. Moniruzzaman, M., Khezr, S., Yassine, A., & Benlamri, R. (2020). Blockchain for smart homes: Review of current trends and research challenges, *Journal of Computers & Electrical Engineering, Elsevier.*

4. Khezr, S., Moniruzzaman, M., Yassine, A. and Benlamri, R., (2019). Blockchain Technology in Healthcare: A Comprehensive Review and Directions for Future Research, *Journal of Applied Sciences, MDPI.*

Submitted:

1. Towards a Trustful Game-Theoretic Mechanism for Data Trading in the Blockchain-IoT Ecosystem, Submitted to *Journal of Network and Systems Management, Springer*.
Status: Conditionally Accepted.

2. Towards a Secure and Dependable IoT Data Monetization using Blockchain and Fog Computing, Submitted to *Journal of Cluster Computing, Springer*.
Status: Accepted with Minor Revision.

3. Blockchain-based Reputation System For IoT Data Ecosystem: A Utility Maximization Approach, Submitted to *IEEE Global Communications Conference (GLOBECOM)*.
Status: Under Review.