UTS UNIVERSITY OF TECHNOLOGY SYDNEY

Utilizing Blockchain for Privacy Preservation in Internet of Things

by Minghao Wang

Thesis submitted in fulfilment of the requirements for the degree of

Doctor of Philosophy

under the supervision of A/Prof. Tianqing Zhu & Prof. Shui Yu

University of Technology Sydney
Faculty of Engineering and Information Technology

October 2023

CERTIFICATE OF ORIGINAL AUTHORSHIP

I, Minghao Wang, declare that this thesis is submitted in fulfilment of the requirements for the award of Doctor of Philosophy, in the *School of Computer Science*, *Faculty of Engineering and IT* at the University of Technology Sydney.

This thesis is wholly my own work unless otherwise referenced or acknowledged. In addition, I certify that all information sources and literature used are indicated in the thesis.

This document has not been submitted for qualifications at any other academic institution.

This research is supported by the Australian Government Research Training Program.

SIGNATURE: Production Note: Signature removed prior to publication.

[Minghao Wang]

DATE: 27th October, 2023

PLACE: Sydney, Australia

C02029: Doctor of Philosophy
CRICOS Code: 009469A
33874 PhD Thesis: Software Engineering
October 2023

Utilizing Blockchain for Privacy Preservation in Internet of Things

Minghao Wang

School of Computer Science
Faculty of Engg. & IT
University of Technology Sydney
NSW - 2007, Australia

Utilizing Blockchain for Privacy Preservation in Internet of Things

A thesis submitted in partial fulfilment of the requirements
for the degree of

Doctor of Philosophy
in
Software Engineering

by

Minghao Wang

to

School of Computer Science
Faculty of Engineering and Information Technology
University of Technology Sydney
NSW - 2007, Australia

October 2023

Abstract

The advent of the Internet of Things (IoT) has led to a highly interconnected digital ecosystem, revolutionizing sectors from healthcare to urban infrastructure. IoT devices amass large amounts of data, resulting in improved efficiency and quality of life. However, the consequential privacy challenges are immense. The decentralized and heterogeneous nature of IoT networks intensifies these concerns, making traditional privacy-preserving mechanisms insufficient.

This thesis advances a novel approach to enhance privacy preservation in the IoT, leveraging the unique attributes of blockchain technology, a decentralized, transparent, and immutable ledger system. Initially, we delve into the realm of crowdsourcing within IoT networks, where trust issues and privacy protection are paramount concerns. We explore the potential of integrating public and private blockchains for differentially private crowdsourcing, establishing a foundational application of blockchain in IoT privacy preservation. This groundwork sets the stage for further exploration of blockchain's broader applicability.

Building on the insights from our exploration of crowdsourcing, the research then extends to multi-agent systems in IoT. These dynamic, complex networks pose their own set of privacy and security challenges. Drawing on the principles established in our initial investigation, we propose a blockchain-based solution to fortify security, augment transparency, and enhance the resilience of multi-agent systems. The proposed approach encourages more effective, reliable, and secure multi-agent interactions within the IoT.

The stage set by our explorations of blockchain applications in crowdsourcing and multi-agent systems then paves the way for addressing federated learning, a form of machine learning ideal for decentralized networks. Despite its promise, federated learning presents unique privacy and computational challenges within IoT. Integrating the insights gained from the initial sections, the thesis proposes a pioneering strategy that fuses public and private blockchain networks. This approach facilitates secure, efficient, and privacy-preserving federated learning within the Internet of Everything (IoE), thus expanding the scope of blockchain's utility in IoT contexts.

In the culminating section, the research introduces a novel blockchain-based defense mechanism against gradient inversion and poisoning attacks on federated learning systems, a significant vulnerability in IoT. Drawing on the lessons from previous sections, this mechanism serves as a testament to the power of blockchain as a comprehensive defense against a spectrum of privacy and security threats.

In summary, this thesis underscores the immense potential of blockchain technology

for enhancing privacy preservation across various aspects of IoT, and advocates for its broader adoption. The study concludes by proposing future work directions, envisioning a more secure, efficient, and trusted IoT landscape by extending these foundational findings to a wider array of IoT scenarios.

Dedication

To Xuhan, whose unwavering love and encouragement powered me through this journey.
To my family, whose steadfast faith and support were my anchor during these transformative years.
And to myself, for remaining resolute in the face of both adversities and triumphs. This accomplishment stands testament to our collective will and determination. . . .

Acknowledgments

I am immensely grateful to those who have made this journey of pursuing a PhD from July 2019 to July 2023 possible and enlightening, despite the unique challenges posed by the COVID-19 pandemic.

Firstly, I extend my deepest gratitude to my advisors, Prof. Wanlei Zhou, A/Prof. Tianqing Zhu, and Prof. Shui Yu. Their unwavering support, mentorship, and expertise have been instrumental in shaping my research and personal growth. They provided me with invaluable guidance and feedback at each stage of this work, consistently challenging me to reach new academic heights.

I am also grateful for the support and insights of Dr. Dayong Ye and all the team members at the Center for Cyber Security and Privacy, UTS. Their collaborative spirit, shared wisdom, and camaraderie have enriched my research experience and made this journey an enjoyable one.

A special note of gratitude goes to my partner of ten years, Xuhan Zuo. Her unwavering support, understanding, and love have been my haven of peace during the challenging times. Xuhan's belief in my abilities and her constant encouragement have been instrumental in my journey.

Lastly, I owe a profound debt of gratitude to my family, who have been my pillars of support throughout my life. Their unconditional love, understanding, and sacrifices have fueled my aspirations and perseverance in completing this journey.

This thesis is a testament to all of your faith and confidence in me. Thank you for being a part of this remarkable journey.

List of Publications

Related to the Thesis :

1. **M. Wang**, T. Zhu, X. Zuo, M. Yang, S. Yu, W. Zhou, "Differentially private crowdsourcing with the public and private blockchain", IEEE Internet of Things Journal, 2023.

2. **M. Wang**, T. Zhu, X. Zuo, D. Ye, S. Yu, W. Zhou, "Public and Private Blockchain Infusion: A Novel Approach to Federated Learning", Under review in IEEE Internet of Things Journal.

3. **M. Wang**, T. Zhu, X. Zuo, D. Ye, S. Yu, W. Zhou, "Blockchain-based Gradient Inversion and Poisoning Defense for Federated Learning", Under review in IEEE Internet of Things Journal.

4. **M. Wang**, T. Zhu, X. Zuo, D. Ye, S. Yu, W. Zhou, "Advising Multi-agent System with Blockchain Network", Under review in IEEE Internet of Things Journal.

Others :

4. **M. Wang**, T. Zhu, T. Zhang, J. Zhang, S. Yu, W. Zhou, "Security and privacy in 6G networks: new areas and new challenges", Digital Communications and Networks, 2020, 6(3): 281-291.

5. **M. Wang**, X. Zuo, T. Zhu, "Blockchain in 5G and 6G networks." (2020): 137-173.

6. S. Shen, T. Zhu, D. Ye, **M. Wang**, X. Zuo, A. Zhou, "A novel differentially private advising framework in cloud server environment", Concurrency and Computation: Practice and Experience, 2022, 34(7): e5932.

Table of Contents

List of Figures

Figure **Page**

List of Tables

CHAPTER 1

INTRODUCTION

1.1 Background

The Internet of Things (IoT) is not just another technological innovation; it represents the next frontier in the digital revolution, with the potential to weave together an intricate web of interconnected devices, sensors, and systems [8]. As we've seen in recent years, the explosive growth of IoT has been driven by advances in wireless communication, miniaturization of hardware, and the rising power of edge computing. The applications of IoT have permeated multiple sectors such as healthcare, transportation, agriculture, and manufacturing, laying the foundation for smarter homes, self-driving vehicles, precision agriculture, and the dawn of Industry 4.0 [11].

In healthcare, the impact of IoT is palpable. Wearable fitness trackers offer personalized insights, and remote patient monitoring systems revolutionize patient care. These devices not only promise enhanced medical outcomes but also a marked reduction in healthcare costs [2]. In transportation, the integration of IoT is reshaping the landscape with more efficient traffic management, fewer emissions, improved safety standards, and the promise of fully autonomous driving. Agriculture is also in the midst of an IoT-driven revolution, with tech-savvy farming techniques enhancing crop health and maximizing yield [53]. In the sphere of manufacturing, IoT is ushering in a new era, streamlining processes such as predictive maintenance, inventory management, and quality control [93].

However, with this interconnectedness comes a looming shadow: the ever-present

concerns of security and privacy [10]. Every connected IoT device offers a gateway to improved functionality, but it also presents a potential avenue for cyber-attacks [4]. Attacks that threaten not just the sanctity and confidentiality of data but, in severe cases, could even wrest control of the devices themselves ,Äì a dire scenario in critical sectors like healthcare or transportation where the implications can be fatal.

These devices constantly collect, process, and transmit a deluge of sensitive data, making them a veritable treasure trove for any malicious entity. The lack of stringent security measures can result in this data being intercepted or accessed, leading to grave breaches of privacy.

Traditional IoT architectures further exacerbate these concerns. Their tendency to rely on centralized data storage creates a glaring vulnerability: a single point of failure. This not only makes them ripe targets for distributed denial of service (DDoS) attacks but also raises significant privacy concerns. Without stringent safeguards, this centralized trove of sensitive data can be misused, not just by external threats but potentially by service providers themselves [16].

Adding another layer of complexity is the sheer diversity of IoT devices [86]. The vast array of capabilities, operating systems, and built-in security protocols means there's no one-size-fits-all solution. This lack of standardization poses a formidable challenge, as ensuring airtight security across such a heterogeneous ecosystem is daunting [21]. Every unique device, with its specific set of vulnerabilities, can inadvertently become the Achilles heel of the network [38].

In summary, while the promise of IoT is undeniably transformative, it is essential to navigate its landscape with a keen awareness of the inherent security and privacy risks. Addressing these challenges is not just important but imperative, demanding a holistic and nuanced approach tailored to the idiosyncrasies of the IoT ecosystem.

1.2 Motivation

The global IoT landscape, characterized by unprecedented interconnectivity and data proliferation, brings to light pressing security and privacy issues that demand immediate attention. The extensive array of IoT applications, coupled with the continuous growth in IoT device deployment, has led to an exponential increase in data generation, much of which is of a sensitive nature. Without robust security and privacy measures in place, this data is left vulnerable to malicious attacks, leading to privacy breaches and severe consequences for individuals and organizations alike.

Traditional security solutions employed in the IoT domain often fail to provide adequate protection due to the unique challenges presented by IoT architectures. These systems are predominantly centralized, creating single points of failure that can be exploited by attackers. Moreover, the heterogeneous nature of IoT devices, stemming from various manufacturers, operating systems, and protocols, complicates the security landscape, making it challenging to implement uniform security measures.

Privacy issues in IoT systems are equally concerning. Many IoT devices are inherently personal and routinely handle sensitive data, such as health records from wearable devices, personal information from smart home appliances, and location data from smart vehicles. In the absence of stringent privacy protection measures, this data can be exploited, leading to privacy infringements that can have significant personal and societal implications.

The inherent limitations of traditional cryptographic methods in the context of resource-constrained IoT devices further exacerbate these concerns. Standard encryption techniques often demand substantial computational resources, rendering them unsuitable for many IoT devices. This results in weak or no encryption for data at rest and in transit, creating a ripe environment for cyber attacks.

In view of these pressing concerns, there is a clear motivation to explore innovative solutions to address the security and privacy issues in IoT systems. This motivation leads to the exploration of blockchain technology, renowned for its decentralization, immutability, and transparency, as a potential solution to these challenges. With its promising characteristics, blockchain has the potential to revolutionize security and privacy preservation in the IoT domain, forming the primary focus of this thesis.

1.3 Research Objectives

This thesis focuses on addressing a multitude of privacy issues associated with the integration of Internet of Things (IoT) systems with the techniques of crowdsourcing, multi-agent systems, and federated learning. These methods are deeply interconnected, each adding layers of complexity to the privacy landscape in IoT. The objective is extensive, aiming to understand, evaluate, and improve current privacy-preserving measures in this interconnected setting, and to anticipate future challenges in the evolving digital landscape. The specific goals of this research are as follows:

- Understanding the Interconnected Privacy Landscape in IoT: The research aims to provide a comprehensive understanding of privacy challenges in IoT systems

when integrated with the interconnected techniques of crowdsourcing, multi-agent systems, and federated learning. It involves an in-depth exploration of existing literature, highlighting the privacy concerns unique to each technique and those arising from their interaction, existing mitigation measures, and their limitations.

- Evaluating Current Privacy-Preserving Methods in Interconnected Systems: A significant portion of this research is dedicated to critically assessing current privacy-preserving methods employed in IoT systems incorporating crowdsourcing, multi-agent systems, and federated learning. The aim is to understand their strengths and weaknesses, their effectiveness in this complex, interconnected setting, and identify possible areas for improvement.

- Developing Enhanced Privacy Measures for Interconnected Systems: Based on the insights gained from the evaluation, this research aims to propose innovative and robust privacy measures that can effectively address the identified issues and shortcomings in this interconnected landscape. This involves the creation and validation of advanced methodologies, models, and techniques, with a significant emphasis on practical applicability and scalability in real-world IoT scenarios.

- Anticipating Future Privacy Challenges in Interconnected Systems: An essential aspect of this research is anticipating and preparing for future privacy challenges in the interconnected landscape of IoT, crowdsourcing, multi-agent systems, and federated learning. The study aims to identify potential threats and challenges, providing foresight that can guide the development of future-proof privacy measures and policies.

- Disseminating Knowledge about Privacy in Interconnected Systems: A fundamental goal of this research is to contribute to the pool of knowledge on privacy issues in this interconnected setting. This involves disseminating the research findings through scholarly publications and practical guidelines, thus assisting researchers, practitioners, and policymakers in understanding and addressing privacy challenges in IoT, crowdsourcing, multi-agent systems, and federated learning.

1.4 Major Contributions of This Thesis

This thesis makes numerous significant contributions to the realm of security and privacy in the Internet of Things (IoT), focusing specifically on the deployment of blockchain

technology. These contributions are encapsulated in four interconnected research articles that together address the primary challenges associated with maintaining privacy and ensuring security in IoT systems:

- Differentially Private Crowdsourcing with Public and Private Blockchain: The first piece of research presents an innovative differentially private crowdsourcing system, leveraging public and private blockchains. This research addresses the critical privacy and trust issues of traditional crowdsourcing systems and offers adjustable privacy protection levels. The solution forms a strong foundation for subsequent investigations in the thesis, given the shared characteristics of trust and privacy needs in multi-agent and federated learning scenarios.

- Blockchain Empowered Multi-Agent Systems: Advancing IoT Security and Transaction Efficiency: Building on the insights gained from the initial exploration, this study proposes a unique integration of multi-agent systems and blockchain technology in the IoT context. We present innovative algorithms that regulate agent activities, thereby enhancing the security, transparency, and robustness of IoT systems. This step brings in the complexity of agent interaction, which serves as a bridge to the next stage of research focusing on federated learning.

- Public and Private Blockchain Infusion: A Novel Approach to Federated Learning: Here, we introduce an innovative approach combining public and private blockchains to enhance federated learning performance while preserving data privacy and security within the Internet of Everything (IoE). This methodology reduces clients' computational demands and ensures trustworthiness in model migration, offering an advanced solution that echoes the privacy and security concerns discussed in the previous two articles.

- Blockchain-based Gradient Inversion and Poisoning Defense for Federated Learning: In the final stage of this research journey, we propose a robust blockchain-based defense mechanism that effectively protects federated learning systems from gradient inversion and poisoning attacks. It demonstrates the potential of our unique approach in enhancing the security and privacy of distributed machine learning systems in various IoT scenarios, thus culminating our research exploration and linking it back to the foundational principles discussed in the first paper.

Together, these research articles showcase the innovative ways that blockchain technology can be utilized to address existing challenges associated with privacy preservation

and security in IoT. In unison, they advance the field of IoT security and privacy, providing invaluable insights and setting a path for future research.

1.5 Thesis Organisation

This thesis is meticulously structured to provide a comprehensive view of the research undertaken. The organization of the thesis is as follows:

- Chapter 1 Introduction: This chapter sets the stage by presenting an overview of the IoT landscape, discussing the unique security and privacy challenges in the IoT domain, and establishing the need for novel solutions.
- Chapter 2 Preliminary and Related Work: The second chapter provides a detailed discussion on privacy preservation in IoT and related works. It also introduces the fundamentals of blockchain technology, providing the necessary knowledge to understand its application in the subsequent chapters.
- Chapter 3 Differentially Private Crowdsourcing with Public and Private Blockchain: This chapter presents the first research article of the thesis. It delves into the development of a differentially private crowdsourcing system that integrates public and private blockchains.
- Chapter 4 Blockchain Empowered Multi-Agent Systems: Advancing IoT Security and Transaction Efficiency: The fourth chapter features the second research article, which proposes a unique confluence of multi-agent systems and blockchain technology within the IoT domain.
- Chapter 5 Public and Private Blockchain Infusion: A Novel Approach to Federated Learning: The fifth chapter introduces an innovative methodology that integrates public and private blockchain networks to enhance the performance of federated learning while preserving data privacy and security within the IoE.
- Chapter 6 Blockchain-based Gradient Inversion and Poisoning Defense for Federated Learning: This chapter presents the final research article that proposes a comprehensive blockchain-based defense mechanism for federated learning systems against gradient inversion and poisoning attacks.
- Chapter 7 Discussion, Future Work, and Conclusion: The final chapter synthesizes the insights derived from the research articles, discusses potential avenues for

future research, and provides concluding remarks on the overall contributions of the thesis.

By providing a clear roadmap of the thesis, this organization enables readers to follow the logical progression of the research and understand the full extent of its contributions.

CHAPTER 2

PRELIMINARY AND RELATED WORK

2.1 Preliminary

2.1.1 Internet of Things

The Internet of Things (IoT) refers to the interconnected network of physical objects which are equipped with sensors, software, and other technologies for connecting and exchanging data with other devices and systems over the Internet [61].

IoT systems have gained immense popularity due to their ability to bridge the gap between the physical and digital worlds, enabling a multitude of innovative applications across various sectors such as healthcare, transportation, manufacturing, smart homes, and smart cities [82]. These applications range from monitoring and control (such as smart thermostats or industrial process control) to complex information processing (like health tracking devices, autonomous vehicles, or drone systems).

At its core, an IoT system consists of three main components: IoT devices, the network, and the cloud or server. As illustrated in the Figure 2.1, the external layer consists of IoT devices that connect to the cloud/server via a network. The details of each component will be discussed in the following sections.

- **IoT Devices:** IoT devices, or "things", are the physical devices embedded with sensors, actuators, and processors. These devices can sense, interact with, and process data from the physical world. They collect, process, and exchange data via wireless or wired connections. The devices can range from simple sensors

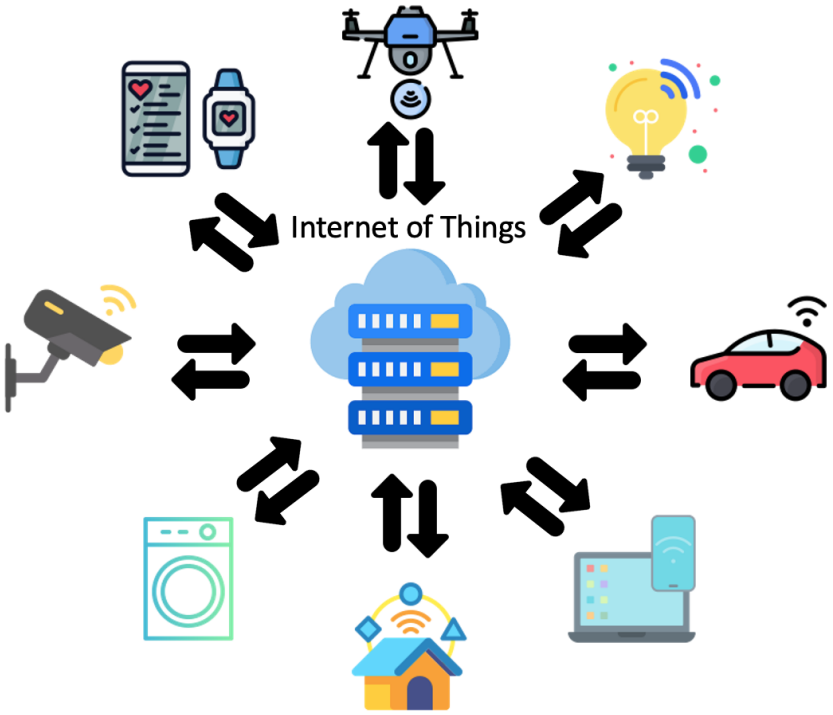

Figure 2.1: Internet of Things

like temperature sensors, to complex devices like drones or autonomous vehicles. Many IoT devices also have processing capabilities enabling them to perform computations on the data they collect before transmitting it.

- **Network:** The network component in an IoT system connects IoT devices to the cloud or server. It facilitates the exchange of data amongst devices and between devices and the cloud. This can involve various communication protocols such as Wi-Fi, Bluetooth, cellular, Zigbee, LoRaWAN etc., depending on the specific requirements of the IoT system such as range, power, bandwidth etc.

- **Cloud or Server:** The cloud component provides the required computing resources for processing and storing the data generated by IoT devices. It can host various services and applications for data analysis, visualization, and other functionalities. The cloud can also be responsible for managing the IoT devices, including tasks like firmware updates, fault detection, and handling communication between devices.

In addition to these core components, an IoT system often involves middleware, a software layer that sits between the hardware (the devices and the network) and the application layer (hosted on the cloud or server). The middleware is responsible for device

management, data management, and provides application support functions like event processing and service coordination.

From a systemic perspective, an IoT network can be defined as a graph $G = (V, E)$, where V is the set of nodes representing IoT devices, and E is the set of edges representing the communication links between the devices. The data generated by the IoT devices and communicated over these edges can be modeled as a time-series, given the temporal nature of the IoT data.

The large-scale, heterogeneous nature of IoT systems, alongside real-time data exchange, imposes several challenges including privacy, security, and data management [64]. IoT devices continuously generate a vast amount of data, often sensitive in nature, raising serious privacy concerns [27]. This data, if not properly protected, could be exploited by malicious entities. Similarly, security is of prime importance due to the potentially severe implications of successful cyber-attacks on IoT systems.

Moreover, in the context of integrating IoT systems with crowdsourcing, multi-agent systems, and federated learning, these challenges become more pronounced. These integrations may increase the complexity of the data interactions, making it critical to ensure privacy-preserving data exchanges and secure computations. As IoT technology continues to evolve and become more integrated with such systems, it becomes essential to consider these challenges and develop effective solutions for a safe, efficient, and reliable IoT ecosystem.

2.1.2 Blockchain

Blockchain technology, colloquially known as a distributed ledger technology (DLT), has emerged as an innovative solution to build trust in decentralized systems, effectively eliminating the need for a centralized authority [77]. This technology has attracted massive interest from various industries and academic disciplines due to its distinguishing features: decentralization, transparency, immutability, and robust security [1].

A blockchain is essentially a distributed, decentralized database, referred to as a ledger. It contains a chronological series of data blocks, each block being a collection of transactions or data records. The blocks are linked to each other using cryptographic principles, thus forming a chain of blocks. This interlinking ensures the immutability of the records; any change in a block would require changing all subsequent blocks, a task that is computationally infeasible.

An expanded formal definition can be illustrated as follows:

Definition 1. Blockchain: A blockchain BC is an ordered sequence of blocks $BC = B_0,B_1,...,B_n$, where B_0 is the genesis block, and each block B_i is defined by a tuple $B_i = (H_i,H_{i-1},T_i,T_{stamp})$. Here, H_i is the current block's cryptographic hash, H_{i-1} is the cryptographic hash of the previous block, T_i represents the transactions included in the current block, and T_{stamp} denotes the timestamp at which the block was created.

The transactions in a blockchain are validated and agreed upon by participants (referred to as nodes) in the blockchain network. This process is achieved through consensus algorithms. Blockchain can process a variety of transactions. In contexts like multi-agent systems, federated learning, and IoT, a transaction might refer to diverse activities such as the registration of an agent, sharing of advice, rating of an agent, or the dissemination of model updates. An example of a transaction is presented below:

Definition 2. Transaction: A transaction tx within our context is defined by the tuple $tx = (tx_{type},tx_{data},tx_{time},tx_{sign})$. Here, tx_{type} denotes the type of transaction, which could be "register", "advise", or "rate". tx_{data} signifies the data pertinent to the transaction, including information about the agent, the advice shared, or the rating. tx_{time} indicates the timestamp when the transaction was created. tx_{sign} refers to the digital signature of the agent initiating the transaction, confirming the transaction's authenticity.

Several key aspects of blockchain contribute to its broad application across numerous domains. Consensus algorithms, for example, are essential to validate transactions and maintain the ledger's consistent state across the network [6]. There are various types of consensus algorithms including Proof of Work (PoW)[57], Proof of Stake (PoS)[41], and others, each with their own advantages and disadvantages.

Cryptographic techniques secure the data, ensuring confidentiality, integrity, and non-repudiation [26]. The combination of public key cryptography for signing transactions and hash functions for linking blocks provides the basis for the security of blockchain systems.

Smart contracts represent another integral part of blockchain technology. They are self-executing contracts with the agreement's terms directly written into the code. These contracts facilitate the automation of various processes in a secure, transparent, and deterministic way [94]. They can be programmed to execute specific actions when certain conditions are met, enabling the creation of decentralized applications (DApps) on the blockchain.

Furthermore, blockchain's inherent immutability ensures that once a smart contract is deployed on the blockchain, it cannot be altered, enhancing trust in the system. This

immutability is a result of the blockchain's design where altering a block would require a recalculation of every subsequent block's hash, which is computationally impractical.

Finally, the principle of decentralization in blockchain enhances resilience against failures and attacks. As each participant maintains a copy of the entire blockchain, even if a part of the network goes down, the system as a whole continues to function. Moreover, the lack of a central authority makes the system robust against single-point failures or attacks. This decentralization is particularly valuable in environments with high security and reliability requirements, such as IoT networks, multi-agent systems, and federated learning scenarios.

2.1.3 Differential Privacy

Differential privacy (DP) is a theoretical construct and a powerful methodological tool that seeks to maximize the utility derived from statistical data analysis while protecting individuals' privacy in the dataset [17]. DP achieves this delicate balance by introducing controlled randomness to the data analysis procedure, preventing precise inference about any single individual's information.

The key idea behind differential privacy is that the addition or removal of a single database entry should not significantly affect the outcome of any statistical analysis. This is achieved by carefully adding noise to the function's output. The noise is typically drawn from a Laplace or Gaussian distribution.

Formally, the DP condition is defined as follows:

Definition 3 (Differential Privacy)**.** A randomized mechanism $\mathscr{M}$ provides (ϵ, δ)-differential privacy if for any two neighbouring datasets D and D' (i.e., D and D' differ in at most one record), the following inequality holds for all events S in the output range of $\mathscr{M}$:

$$\Pr[\mathscr{M}(D) \in Y] \leq e^{\epsilon} \Pr\left[\mathscr{M}\left(D'\right) \in Y\right] + \delta$$

In this definition, $\Pr[\cdot]$ denotes probability, S represents any possible output of the algorithm $\mathscr{M}$, and the probability space is over the coin flips of the randomized mechanism $\mathscr{M}$. Here, ϵ is a non-negative parameter controlling the amount of privacy (a smaller ϵ yields more privacy), and δ is a parameter that provides a bound on the chance that the privacy guarantee of ϵ is violated. When $\delta = 0$, the mechanism $\mathscr{M}$ is termed as ϵ-differentially private.

In practice, the choice of the privacy parameters ϵ and δ involves a trade-off between data privacy and data utility. Smaller values of these parameters result in stronger

privacy protection but introduce more noise, thereby potentially reducing the usability of the data [89]. Conversely, larger parameter values decrease privacy protection but make the data more usable.

An essential feature of differential privacy is its robustness to post-processing, which means that no matter how the output of a differentially private mechanism is further processed, the resulting output remains differentially private. This guarantees privacy protection irrespective of future data processing or analysis. Furthermore, differential privacy provides a composability property, which means that a sequence of differentially private computations will itself be differentially private. This is critical in scenarios where multiple queries or computations are performed on the same dataset.

Today, differential privacy finds extensive use in numerous sectors, including technology, healthcare, and research, owing to its strong privacy guarantees and versatility. Major technology companies like Apple [13] and Google [18] have adopted differentially private mechanisms to protect user data while drawing statistical inferences. Differential privacy thus serves as an essential tool for preserving privacy in the age of data proliferation.

2.2 Related Work

In recent years, the Internet of Things (IoT) has grown in importance due to its critical role in interconnecting a vast array of smart devices, facilitating the seamless exchange of data. As a result, it plays an essential part in many areas such as smart cities, healthcare, transportation, and industrial automation. However, this surge in interconnectivity also magnifies privacy concerns, as these IoT systems often handle sensitive information. Therefore, ensuring privacy within the IoT domain has become a top priority in both academia and industry.

Research efforts addressing privacy challenges in IoT have adopted various strategies, including crowdsourcing, multi-agent systems, and federated learning. Crowdsourcing, which leverages the collective intelligence of a large group of individuals, offers a promising avenue for IoT applications, though the increase in data sharing surfaces additional privacy risks. Similarly, multi-agent systems, which employ multiple interacting agents to solve complex tasks, have their own set of privacy issues, particularly concerning data sharing among agents. Federated learning, a machine learning approach where the learning process is distributed across many devices, also offers significant benefits to IoT applications. Still, it is not without privacy concerns, especially relating to the potential

leakage of sensitive data during the learning process.

This section aims to provide a comprehensive overview of current research exploring privacy issues in IoT in the context of crowdsourcing, multi-agent systems, and federated learning. Through an in-depth discussion of these studies, we seek to shed light on the current privacy challenges and solutions within the realm of IoT, thereby contributing to the continuous effort to enhance the privacy and security of IoT systems. Additionally, we have conducted comparative analysis between selected research studies and our proposed chapters.

2.2.1 Privacy Issues when Combining Crowdsourcing with IoT

Table 2.1: Summary of research on privacy issues when combining crowdsourcing with IoT

Author(s)	**Main Contribution**	**Key Techniques**
Ren et al. [66]	Reinforcement learning-based crowdsourcing	Trust evaluation
Yu et al. [90]	Crowdsourcing privacy protection	Multiauthority ciphertext-policy
Zhang et al. [91]	Privacy protection using BGV	Double-projection layers
Sharma et al. [69]	Edge-crowd integration	Mini-edge servers, entropy modelling
Ma et al. [51]	Data privacy in crowdsourcing	Hash function, Merkle tree
Gan et al. [22]	Multihop routing	Task requester's privacy
Seliem et al. [68]	Survey on IoT privacy issues	No solutions provided
Ang et al. [5]	Trustworthiness, privacy	Need for a trusted third party

When merging crowdsourcing with IoT networks, numerous studies have already presented privacy-preserving approaches. For instance, Ren et al. [66] described an unique reinforcement learning-based intelligent crowdsourcing approach with privacy protection. In their paper, they employ a trust evaluation mechanism to prevent co-cheating and to oppose the participant's privacy exposure preference.

To strengthen privacy protection in a data-sharing environment, Yu et al. [90] suggested a crowdsourcing privacy protection technique based on multiauthority ciphertext-policy attribute-based encryption. The core aim of their article is to present an independent key component distribution technique and employ an encrypted way to preserve the privacy of participants.

Zhang et al. [91] proposed a method for protecting the privacy of participants using the BGV encryption algorithm. In their paper, they propose replacing the conventional deep computation model's hidden layers with double-projection layers in order to solve the privacy issues in their newly designed model, which could project the raw input into two distinct subspaces in the hidden layers in order to learn interacted features of big data.

Sharma et al. [69] described a strategy that focuses on edge-crowd integration to preserve trust and privacy regulations in IoT. The suggested technique employs crowdsourcing as mini-edge servers and entropy modelling to establish entity-to-entity trust.

Ma et al. [51] discussed the use of a hash function and the Merkle tree for privacy protection. They present exhaustive evidence of employing the hash function, which could enable the protection of data privacy in crowdsourcing.

Gan et al. [22] designed a multihop routing incentive mechanism that can preserve task requester's privacy. However, our proposed method aims to protect the worker's privacy, not the requester's. Moreover, we are aiming for a smart contract rather than an incentive mechanism.

Seliem et al. [68] provided a comprehensive survey about IoT privacy issues. They pointed out that privacy may be leaked out from either data or tasks when combining crowdsourcing with the IoT environment, but they did not provide any potential solutions to solve the privacy issues when combining crowdsourcing and IoT together.

Ang et al. [5] surveyed the trustworthiness, privacy and security of combining crowdsourcing with IoT. According to their research, most solutions need a trusted third party in their environment, which will cause some privacy and security issues that lead to the centralized problem.

2.2.2 Privacy Issues when Combining Multi-Agent Systems with IoT

Table 2.2: Summary of research on privacy issues when combining multi-agent systems with IoT

Author(s)	**Main Contribution**	**Key Techniques / Focus**
Liang et al. [45]	Intrusion detection system	MAS, blockchain, deep learning
Mezquita et al. [54]	Security in smart grids	MAS, blockchain in MicroGrid
Nguyen et al. [58]	Task offloading and block mining	Blockchain-based MEC system
Luo et al. [50]	Distributed electricity trading	MAS, blockchain for trading
Calvaresi et al. [9]	Trustworthiness in AI agents	Blockchain, explainable MAS
Kapitonov et al. [34]	Communication protocol for MAS	Ethereum blockchain, UAVs
Yang et al. [84]	Meme transmission prediction	Decentralized blockchain, MAS

In the realm of Internet of Things (IoT) applications, the integration of multi-agent systems is not a novel concept. However, our focus lies in soliciting manuscripts that present innovative approaches merging multi-agent systems with blockchain technology to address the pressing privacy concerns encountered within IoT environments. For in-

stance, the research conducted by Liang et al.[45] develops an intrusion detection system for IoT networks, employing a unique combination of a multi-agent system, blockchain, and deep learning through a hybrid placement strategy. The system effectively counters transport layer attacks, as evidenced by the NSL-KDD dataset. In contrast, our research elaborates upon this premise, eschewing deep learning algorithms in favor of an innovative amalgamation of blockchain and multi-agent systems. Our approach utilizes smart contracts to govern agent interactions, thereby enhancing the decision-making processes. Simulation results consistently indicate performance improvements over traditional multi-agent systems. Moreover, we address a wider range of challenges encompassing privacy, security, scalability, and efficiency within multi-agent IoT environments.

The work of Mezquita et al.[54] pertains to the enhancement of security within smart grids. Their research employs an integrated multi-agent system and blockchain technology to facilitate peer-to-peer electricity trading within a MicroGrid (MG) environment. The application of blockchain amplifies security, reduces transaction costs, enables micro-transactions, and authenticates data ownership. The inclusion of a multi-agent system optimizes energy costs and fosters profitability in local energy production. Despite their system being tailored specifically to smart grid scenarios, our research introduces a more adaptable framework, efficiently leveraging multi-agent systems and blockchain technologies across diverse IoT settings. Additionally, we take a step further by explicitly addressing privacy and security issues while introducing an innovative agent rating system. This addition enhances advice-sharing reliability, leading to more robust operations.

In their paper, Nguyen et al.[58] elucidate a novel cooperative task offloading and block mining (TOBM) scheme for a blockchain-based mobile edge computing (MEC) system. The focus is on optimizing various parameters such as offloading decisions, channel selections, transmit power allocations, and computational resource allocations to maximize system utility. The research introduces a Proof-of-Reputation consensus mechanism to mitigate latency problems. Although their framework outlines a solid approach for integrating MEC and blockchain technologies, our work possesses distinct advantages. We broaden the application of blockchain and multi-agent systems beyond MEC to cover various IoT applications. Moreover, we introduce a unique agent rating system and an innovative advice-sharing method which contribute to improved system efficiency. Furthermore, our comprehensive privacy and security analysis bolsters the robustness of our model across different environments, thus showcasing our proposed framework's wider applicability.

Luo et al.[50] discuss a distributed electricity trading system designed to facilitate peer-to-peer electricity sharing among prosumers, encapsulating both a multi-agent system and blockchain technology. Their model caters aptly to the niche of electricity trading. In contrast, our proposed framework broadens the application scope to a wider array of IoT applications. Moreover, our research offers a refined agent rating system and a unique advice-sharing mechanism to enhance the functionality of multi-agent systems, aspects not explicitly addressed in the aforementioned work. Additionally, our privacy and security analysis reinforces the robustness of our system across varying conditions, yielding a more versatile and efficient solution for diverse scenarios within the IoT context.

Calvaresi et al.[9] underscore the expanding entwinement of AI with human society, particularly with the advent of multi-agent systems (MAS) in critical areas such as healthcare and finance. Their paper acknowledges trust issues arising from a lack of explainability in AI agents and proposes a solution merging blockchain technology with explainability in MAS decision-making processes. Their aim is to create more transparent, secure, and thus trustworthy systems from a human user's perspective. Our research, while aligning with the goal of trustworthiness, takes a slightly different approach. We amalgamate blockchain with multi-agent systems to enhance security and privacy in IoT applications and build trust via an effective agent rating mechanism. Furthermore, our system exhibits superior performance in reducing average hits and steps, indicating more efficient decision-making, which might implicitly augment user trust. Therefore, while both studies aim to enhance trust and efficiency, ours specifically thrives in IoT environments, demonstrating practical superiority and applicability.

Kapitonov et al.[34] present a novel communication protocol that allows agents within a multi-agent system (MAS) to interact and make decisions for executing tasks in cyber-physical systems. They focus on autonomous agents such as robots or smart things that operate in unreliable and unknown environments. The authors employ decentralized Ethereum blockchain technology and smart contracts to orchestrate peer-to-peer network communication among agents and present an architecture for autonomous business activity based on this communication method. Finally, the paper describes the practical application of this methodology in an autonomous economic system with unmanned aerial vehicles (UAVs). Our research, in comparison, also focuses on enabling secure and efficient interactions in MAS via blockchain technology. However, our approach diverges as it caters specifically to IoT environments. Furthermore, our work integrates an effective agent rating mechanism to increase trust and accountability, which the cited

research does not explicitly address. Both works are valuable, but ours is tailored for IoT applications and demonstrates enhanced decision-making efficiency and trustworthiness.

Yang et al.[84] utilize decentralized blockchain theory and multi-agent system for meme transmission prediction and discovery. Their system outperforms traditional prediction methods, showcasing improved prediction and a novel system, Meme-chain, which adeptly handles meme discovery and information transactions. In contrast, our study employs similar technologies but with a focus on securing IoT networks. We concentrate on analyzing agent behavior for intrusion detection, thereby enhancing the effectiveness of the IoT network. While both studies share technological application, our unique contribution lies in the domain of IoT security.

2.2.3 Privacy Issues when Combining Federated Learning with IoT

Table 2.3: Summary of research on privacy issues when combining federated learning with IoT

Author(s)	**Main Contribution**	**Key Issues/Threats**
Geng et al. [24]	Risk of gradient leakage	Gradient leakage
Madni et al. [52]	Highlighted gradient leakage risk	Gradient leakage
Fang et al. [20]	Addressed gradient leakage	Gradient leakage
Kim et al. [37]	Privacy and security risks	Centralizing training data
Li et al. [43]	Potential risks of centralization	Centralizing training data
Various [20, 24, 32, 36, 48, 52, 70]	Gradient inversion attacks	Gradient inversion
Various [12, 29, 62, 75]	Poisoning attacks	Data poisoning

In federated learning and blockchain systems, privacy issues have been at the forefront of concerns. Geng et al. [24], Madni et al. [52], and Fang et al. [20] highlighted the risk of gradient leakage, a substantial concern in federated learning where model updates are shared across devices. Simultaneously, Kim et al. [37] and Li et al. [43] delved into potential privacy and security risks associated with centralizing training data in federated learning, proposing that blockchain can provide decentralization to mitigate these risks.

However, while the above research provides valuable insights, several gaps and limitations persist:

- Limited Mechanisms for Privacy Preservation: Most of the existing work centers around identifying threats without proposing effective, scalable, and real-time solutions to preserve privacy during the training process.

- Scalability Concerns: With an ever-increasing number of devices partaking in federated learning, the proposed mechanisms must be evaluated under larger-scale scenarios, a dimension often overlooked.
- Generalization across Federated Networks: The techniques and protocols discussed have been predominantly tested in specific environments. Their efficacy across diverse federated learning setups remains under-investigated.

Specific attack threats have also been pinpointed. A significant tranche of the research has homed in on gradient inversion attacks [20, 24, 32, 36, 48, 52, 70] and poisoning attacks [12, 29, 62, 75]. Both attack vectors present profound privacy challenges, threatening the integrity of model training and the potential exposure of sensitive information.

In light of these limitations, this thesis introduces novel mechanisms for real-time privacy preservation in federated learning, ensuring data protection without compromising the quality of the trained model. Furthermore, we propose a scalable blockchain-based solution for federated networks, evaluated across various environments, confirming its efficacy and generalization capabilities. Through rigorous evaluations against gradient inversion and poisoning attacks, our techniques consistently demonstrate enhanced resilience, marking a pivotal advancement in the domain of federated learning and blockchain systems.

CHAPTER 3

DIFFERENTIALLY PRIVATE CROWDSOURCING WITH THE PUBLIC AND PRIVATE BLOCKCHAIN

3.1 Introduction

With the proliferation of network-connected smart sensing devices, the Internet of Things (IoT) has increased communication range, storage capacity, and computational power, which makes the development of large-scale data-based applications promising [47]. Crowdsourcing is a prevalent IoT application that could utilise sensor-equipped mobile devices to collect and exchange data. Combining crowdsourcing and the Internet of Things could make data collection more convenient and effective [79]. In addition, as a natural progression of the IoT concept, IoE consists of four essential components: people, things, data, and processes [55]. The increasing number of participants in an IoE scenario not only creates greater chances for the growth of crowdsourcing, but also increases the possibility of being fooled by hostile attackers or untrustworthy participants[35]. When merging crowdsourcing with an IoE environment, privacy and trustworthiness of IoT devices and data are two major obstacles that must be addressed.

Combining IoT with crowdsourcing will present numerous security and privacy problems, particularly privacy dangers posed by data and data integrity. In certain sensing jobs, the data are initially detected by the sensing IoT devices, then transferred to the service provider, and then returned back to end users or stored in a storage system [73]. The sensed data may contain sensitive information about the Internet of Things

devices, such as their position and identification. Sending the data back to IoT devices or storing it with the service provider increases the danger of sensing data exposure [81]. This procedure may expose sensory information to an unconnected or malicious party. In addition, due to the large number of IoT devices engaging in the activity, the traditional identity procedure of crowdsourcing will not be effective. We must also examine the credibility of the participant and the data.

Various techniques, such as dummy locations [60] and differential privacy [83, 92], have been proposed to overcome the privacy problem in crowdsourcing systems. For instance, Sun et al. [72] developed a twofold disturbance localised differential privacy technique to disrupt workers' location data. To broaden the breadth of privacy protection, Wang et al. [76] presented a location obfuscation strategy that combines $epsilon$ differential privacy with $delta$ distortion privacy. To protect location privacy, Qian et al. [63] suggested a geo-indistinguishable technique based on differential privacy. Some sensing data have been transferred to a third party without considering the third party's trustworthiness, despite the fact that the majority of these articles have adopted ways to preserve users' location privacy.

Motivated by the aforementioned privacy-related difficulties, we are searching for a reputable third party to resolve these issues. As a reliable third party in crowdsourcing, blockchain technology may be a suitable technology. A user's registration on the blockchain network is anonymous due to certain characteristics of blockchain. A user will receive a pair of public and private keys upon registering for the blockchain network. These two keys are created randomly and are unique for each user.

While considering the vast number of IoE devices in the IoT scenario, blockchain possesses sufficient network capacity. The smart contract, which is analogous to a logic programme that runs on the blockchain network, might ensure that the pre-written logic is executed automatically. All members in blockchain networks must adhere to the logic of smart contracts, which also means that blockchain can serve as a trusted third party.

In addition, several specific blockchain characteristics could ensure data privacy when employing blockchain in a system for crowdsourcing. For instance, we may use the private blockchain to enhance the level of user privacy protection. Due to the properties of the smart contract, we may automate the process of protecting privacy by incorporating privacy-preserving mechanisms within the smart contract.

Due to these benefits of the blockchain network, we presented a blockchain-based method for differentially private crowdsourcing. The overview of our system is depicted in Figure 3.1. In this figure, there are two different types of blockchain networks: public

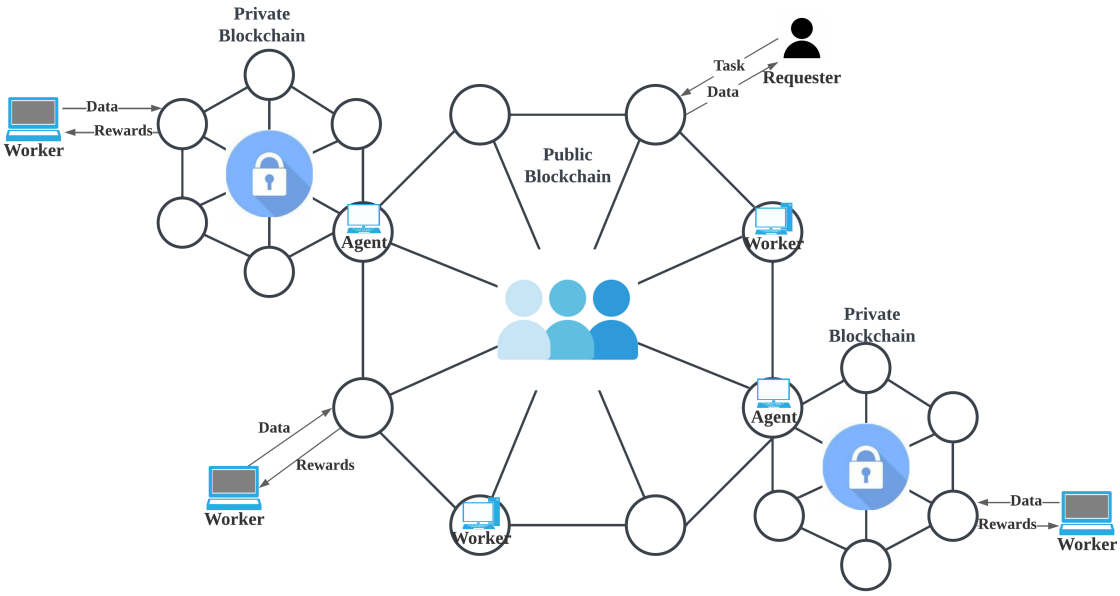

Figure 3.1: Framework of the proposed differential private crowdsourcing system

blockchain and private blockchain. The requester sends the task to the public blockchain and receives the data from it. If the worker chooses to complete the task on the private chain, the Agent assists in publishing the task to the private blockchain. Otherwise, the worker simply retrieves the task from the public chain, completes it, and then submits it back to the public blockchain network.

The main contributions of this chapter are summarized as follow:

- We proposed a blockchain-based differentially private crowdsourcing framework which is suitable for IoT environment. The framework brings significant trustworthy to all crowdsourcing participants .

- The framework could protect both the user's location privacy and the identity privacy of the user with different privacy-preserving levels via combing the public blockchain and private blockchain; the user could choose privacy-preserving levels according to their preference.

- We further systematically analyzed the throughput, latency and safety of the proposed system.

3.2 Crowdsourcing Preliminary

Crowdsourcing was initially described by Jeff Howe in 2006 [28]. It is a mix of the terms "crowdsourcing" and "outsourcing." Simply said, crowdsourcing is when a business or institution outsources its needs to an external network and pays for the outcomes [80]. There is also a literature review by Enrique [19] regarding the definition of crowdsourcing. Typically, crowdsourcing involves a large number of volunteers who are equipped with mobile sensors and travel to the mission site to collect sensory data. The classic system for crowdsourcing consists of three major components: requesters, servers, and workers. The requester will transmit the requirement to the server, which will then assign the task to the worker; the worker will receive payment upon completion of the task. The two types of job allocation in traditional crowdsourcing are worker selected tasks (WST) and server assigned tasks (SAT) [85].

In WST, the server will post the job and wait for the worker to select it based on his or her preferences. The benefit of WST is that the specific location of the workers will not be revealed, and the server will not know the job the worker selects. Because the worker's current location is unknown, the server is unable to predict the worker's destination. The clear disadvantage of this model is, however. The service has no control over the task allocation, and the worker will select the assignment according to their preferences. This approach will result in a low assignment success rate, which may provide a concern for efficiency.

In SAT, the worker must upload their current location to the server and wait for the server to select them. In this architecture, the server has complete control over job assignment. However, this paradigm has the disadvantage that both the worker's present location and job assignment information are disclosed to the server, which may pose privacy concerns.

Both types of traditional crowdsourcing have privacy issues within their respective systems. When combining crowdsourcing and IoT, efficiency and privacy issues must be considered.

3.3 Problem Definition and System Model

3.3.1 Probelm Definition

In this research, we present a blockchain-based crowdsourcing system that is differentially private and deployable in an IoT environment. The existing system of crowdsourc-

ing is unfit for future IoE networks. In the model that we developed, the identities of all devices were effectively resolved. All workers were able to complete the task and submit their responses and information to the blockchain network. In addition, workers could finish the assignment on either the public or private chain, depending on their personal inclination. However, since our technology will be deployed in IoE networks, the number of users will be substantial.

Moreover, the server in the blockchain system may be untrusted. The problem definition will show in the following. There should be a set of workers $\langle W_1, W_2, ..., W_n \rangle$, each of the workers is registered in our system and get their public key P_k and private key S_k. When the worker wants to finish the task allocated by the server, they should upload their sensitivity location-related data-set $\langle W_i, Task_n, E_i^{Task_n}, T_i^{Task_n} \rangle$ to the smart contract. The $E_i^{Task_n}$ and $T_i^{Task_n}$ means the energy cost and time consumption of the user to complete the $Task_n$.

However, when the user uploads their task data-set, all participants in the blockchain network can get their data set. This process will expose the user's identity and location information to the attacker. When attackers get this data-set, they could reckon the user's location and identity information according to the energy cost $E_i^{Task_n}$ and time consumption $T_i^{Task_n}$ in the data-set.

Therefore, a blockchain-based crowdsourcing system that provides a comprehensive privacy-preserving technique to secure worker privacy is required. This implies that not only must the user's location-related information be protected, but also the user's identity-related information. Unlike other relevant work, the majority of these projects create a third-party service to implement the privacy-preserving strategy. In the smart contract, we wish to implement the differential privacy approach, which will not transfer information to an untrusted third party.

3.3.2 System Model

To address the privacy issues described in the preceding section, we have constructed a novel architecture for a crowdsourcing system, details of which are presented below. Four parties are involved in our proposed framework:

- Requester: Initiates and dispatches tasks to the distributed ledger network. The main aim is to obtain ample location information data that align with their requirements. Via the blockchain system's smart contract, the requester obtains and compensates for qualifying data.

- Agent: The agent orchestrates the private blockchain for workers who prioritize a heightened privacy level. Once a worker opts for the private blockchain, the agent fetches the relevant task, placing it on the private chain awaiting task completion. The agent garners rewards when they facilitate this transfer to the private blockchain.

- Worker: As the pivotal task contributor, workers adhere to the requester's guidelines for task completion. Post-task, the data quality undergoes evaluation, with high-quality contributions earning rewards through smart contracts. Notably, all workers within the blockchain network remain anonymous.

Under the proposed system, the requester initializes their participation by registering on the blockchain network and subsequently releasing tasks. Workers then have an important decision to make regarding their privacy level:

- Regular Privacy Protection: Should the worker be content with standard privacy safeguards, they can directly upload their solutions, energy consumption, and time metrics to the public blockchain. Here, data protection is achieved using noise mechanisms like Dirichlet and Laplace noise.

- Enhanced Privacy Protection: For those seeking superior privacy measures, the private blockchain route awaits. In this approach, only the hash of the task's solution will be stored on the public blockchain, ensuring maximum privacy. An agent mediates the task data management, downloading the task and awaiting the worker's confidential submission on the private chain.

In the public chain, the use of Dirichlet and Laplace noise offers a differential privacy guarantee with a privacy budget ϵ. For the private chain route, only the hash of the solution is recorded on the public blockchain. This method offers a substantially stronger layer of privacy protection as the direct linkage between task data and its solution is obfuscated by the hash. Specifically, for tasks completed on the private chain, we can say the effective privacy level is significantly heightened compared to the public chain due to the absence of direct data-task association and the cryptographic strength of hashing techniques.

Upon the conclusion of the worker's task, the quality of data undergoes an assessment, and high-quality contributions are duly rewarded.

3.3.3 Adversary Model

Since all the data saved in the block is accessible to all blockchain users, our attacker has complete access to the data. Data-mining-based attack tactics are regarded as the primary harmful activity. Attackers. The attackers might be any blockchain network participant.

- Participants in the public blockchain: The number of network users will expand significantly in IoE networks. Due to the nature of blockchain networks, anyone is able to join, hence there must also be attackers among the users. In our planned blockchain network, however, we have included the permissioned blockchain, which requires all players to be authorised prior to registration. Moreover, in a public chain, sensitive data such as energy cost and time consumption may result in the worker's location privacy and identity privacy being encrypted using a mechanism that protects confidentiality. Therefore, it is challenging for an attacker to determine the true identities and whereabouts of the workers.

- Workers: A worker in blockchain networks may potentially be an attacker. In our proposed blockchain network, however, all users are required to make a cryptocurrency deposit, which will be forfeited if they engage in malevolent behaviour. In addition, if the user wishes to join the private blockchain, they must disclose their true identify to the private chain's membership control protocol.

- Agents: It is also conceivable for the agent to be malicious. They may exhibit harmful behaviour in blockchain networks. As with the worker, though, all users must deposit some cryptocurrency as a margin. If their nefarious behaviours are uncovered, the agent-influenced margin will be sent to the user.

3.4 Proposed System

In this part, a private crowdsourcing system with distinction will be presented. The proposed solution could tackle some significant privacy disclosure issues in the crowdsourcing system, including the location and identity of workers. Additionally, the system might automatically protect the location and identity privacy of workers.

3.4.1 Overview

Combining the blockchain with crowdsourcing has advantages over the original crowdsourcing approach, including decentralisation and anonymity. It could finish the entire crowdsourcing procedure without the assistance of a third party [42]. In a crowdsourcing system, blockchain functions as a third party, but not as the sole third party. On the blockchain, all workers and requesters are anonymous, so it provides primary security for the user's identity privacy. In addition, because blockchain has smart contract components, when a worker completes a task and passes a quality test, he or she will receive compensation immediately.

Figure 3.2 provides an illustrative representation of our proposed blockchain-integrated differentially private crowdsourcing system. The details of this figure are shown in the following:

- Registration Phase: At the system's outset, the worker, requester, and agent undergo a registration process on the public blockchain. This ensures their verified participation within the ecosystem.
- Task Propagation: Post-registration, the requester transmits the task information to the public blockchain. This task broadcast ensures visibility to all registered workers.
- Worker Decision: Workers, upon perceiving a task of interest, must make an informed decision about the blockchain they wish to operate on.
    1. If they opt for the public blockchain, their uploaded sensory data will be guarded by our differential privacy protection methodology. Successful completion fetches them rewards directly from the public blockchain.
    2. For those leaning towards the private blockchain for augmented privacy, an intermediate step is introduced. A designated agent, upon verification, bridges the task information from the public to the private blockchain. As the worker operates and uploads data to the private blockchain, only the data's hash gets reflected on the public blockchain, ensuring a heightened layer of data privacy.

The details of our proposed privacy-preserving method will be explained in the following parts.

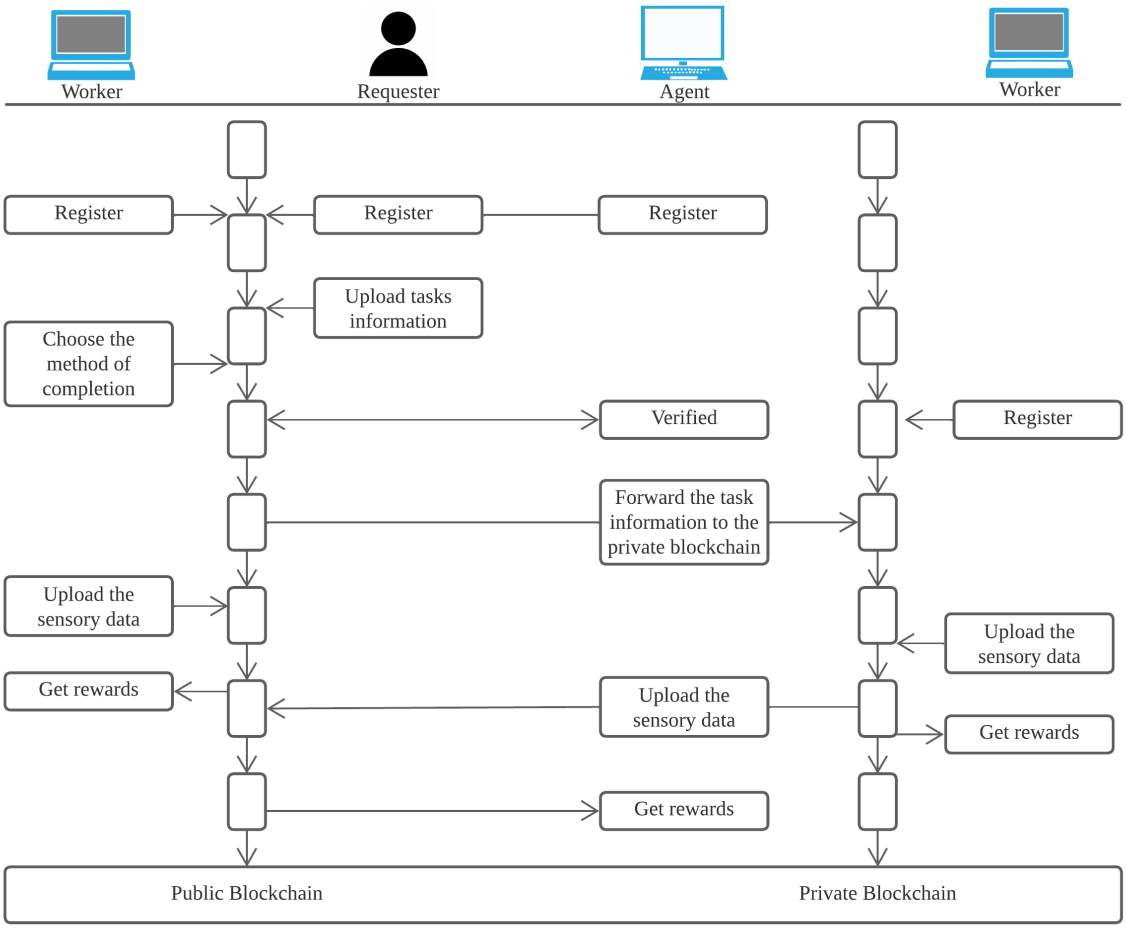

Figure 3.2: Overview of the executive process in the proposed system

- Register (Public Blockchain): All works, requesters, and agents must register in order to participate in the blockchain-based crowdsourcing system. Each registration receives a pair of keys consisting of a public key and a private key, and their identification is recorded in the user pool, which includes their username and the organisation to which they belong.
- Task Announcement: Once the requester has obtained their public and private keys, they are able to post task specifications on the blockchain-based crowdsourcing platform. They must provide the task's requirements and the incentives for each task.
- Smart Contract Creation in Public Blockchain: To ensure fair trade, the system would generate a smart contract that the traditional crowdsourcing system does

not possess, which could execute automatically based on the previously established protocol. All individuals who wish to join this system, both requesters and workers, must first deposit a margin of cryptocurrency. After the task is successfully published, workers can choose whether to do it on the public or private chain based on their desired level of privacy protection. If the worker decides to complete the task on the public chain, the end user(EU) will upload the answer, energy cost, and time consumption to the blockchain where the energy cost and time usage have already been stored.

- Load Tasks to the Private Blockchain: If the workers want to finish the task on the private chain, the leader will retrieve the task-related information from the public blockchain and post it to the private blockchain network. The agent is responsible for ensuring task information uniformity on the two blockchain networks (the public chain and their private chain).

- Upload Sensory Data: Workers upload sensor data to a distributed ledger. In a private blockchain, the worker's energy cost and time consumption will be stored using the same technique as in a public blockchain. In addition, the user might enter the answer in the private answer section, preventing other organisation participants from viewing the final response. The participant from other organisations could only view the hash value of the secret chain information, as well as the already-encrypted energy cost and time consumption. After confirming the uploaded data's quality. The qualified statistics are approved and recorded, and the associated workers are compensated. If the sensory data is deficient, the worker forfeits their security deposit.

- Payment: If the uploaded data is qualified, the smart contract automatically executes the payment process.

3.4.2 Implementation of the proposed system

3.4.2.1 Register

A user is not required to register on the blockchain with his or her true identify. The user who wishes to register in the blockchain must provide a username and organisation name, i.e., an anonymous name and the name of the organisation to which he or she belongs. After successful registration, the user will receive a pair of keys provided by the organization's CA.

Algorithm 1 Register in the Public Blockchain

Require: U_{type},U_{id}, *Org*
Ensure: *RegisterSucess*, P_k,S_k,*jwt*
1: *RegisterSucess* = *False*;
2: **if** $U_{id} \in U_{pool}$ **then**
3: **return** Change U_{id}
4: **end if**
5: $U_{type} \in$ *Worker, Requester*;
6: P_k,$S_k \leftarrow$ *keyGenerator()*;
7: $U_{id} \leftarrow P_k$,S_k ;
8: $Pool_u \leftarrow U_{pool} \cup ID_{ui}$;
9: *RegisterSuccess* = *True*;
10: **return** *RegisterSucess, jwt*

Algorithm 1 provides the details of the registration process. The *RegisterSuccess* is an indicator that reflects whether the user is registered successfully or not. When the user starts registering, the blockchain network will judge if the U_{id} has already been contained in or not. If the U_{id} has already existed in the U_{pool} it will return the message which will mention the user to change another U_{id} that means the registered progress will be false (Step 2 to 4). U_{type} indicates the type of the registered users, which contains two types (worker or requester). The *keyGenerator()* will generate a pair of keys and give them to the user according to X509 certification (Steps 6 to 7). Step 8 means if the user ID is not in the user ID pool $Pool_u$ it will be added inside, and the *RegisterSuccess* indicator would change to *True* in Step 9. Finally, it will return the *RegisterSuccess* and a *jwt* token, which will be used in the next progress. Moreover, the jwt token is generated according to the user ID, and its X509 certification results.

3.4.2.2 Task Announcement

After successfully registering, the requester could advertise the task on the public blockchain. The task information should include the task requirement, task ID, the task's reward, the due date, and the task's status. Moreover, the requester must possess the *jwt token* in order to publish the job on the public blockchain, ensuring that the task information is accurate and trustworthy.

In Algorithm 2, the indicator *TaskSuccess* indicate *Task* is successfully announced or not. Whenever a requester wants to announce a task, the blockchain network would check the *jwt token* of the requester first. If the *jwt token* is not effective, it will respond with the *jwt token* expired message (Step 2 to Step 4). In Step 5, some task-related

Algorithm 2 Task Announcement

Require: *jwt token*, *T, t, ID_t, R, S*
Ensure: *TaskSuccess, Task*
1: *TaskSucess* = False;
2: **if** *jwt token ineligibility* **then**
3: **return** *jwt token* expired
4: **end if**
5: *Task* ← *T, t, ID_t, R, S*;
6: *TaskSuccess* = *True*;
7: **return** *TaskSuccess, Task*

information *T, t, ID_t, R, S*, which means task requirement, task due time, task ID, rewards and task status, respectively, would give to *Task*. Then the *TaskSuccess* will be changed to *True* and the *TaskSuccess* and task information *Task* would be return (Step 6 to Step 7).

3.4.2.3 Smart Contract Creation in Public Blockchain

In the old crowdsourcing approach, several problems, such as the payment issue, could not be effectively resolved. However, with a blockchain-based system for crowdsourcing, this issue could be effectively resolved. In a blockchain-based crowdsourcing system, a special component known as a smart contract may assure fair trading. This smart contract would execute automatically based on the previously established protocol.

In addition, all users, both requestors and workers, who wish to announce a task or join a task must first deposit cryptocurrency as a margin. When the task is announced, it will be sent immediately to the blockchain through smart contract. The IoE background indicates that it is an Internet of Everything network. Therefore, all devices that have already registered on the blockchain might view the task specifications and submit their responses. As described in the previous section, the worker could choose the amount of privacy protection, allowing them to execute the transaction on the public or private chain, depending on their option. Algorithm 3 and Algorithm 4 respectively illustrate the algorithm for workers deciding whether to finish on the public or private chain.

3.4.2.4 Block Creation Algorithm in Public Chain

After the smart contract has been successfully generated, the system-registered worker could view the assignment. All workers will have access to the task specifications. According to our design model, when workers wish to complete a task, they must decide

whether to complete it on the public or private chain, based on their privacy-protection preferences. When customers want to complete on the public blockchain, the system will first validate their jwt token. Workers will be able to submit a response to the smart contract if the jwt token is valid.

In algorithm 3, the worker's chosen indicator c is used to indicate whether the worker chooses to finish the task on the public chain or private chain. After workers choose to finish the task on public blockchain, they need to send the response result set $\langle W_i, Task_n, E_i^{Task_n}, T_i^{Task_n} \rangle$ to SC (Step 1 to 2). SC need to wait for enough response (Step 3). In Step 4, when SC already get enough response result set, it will generates the Laplace noise *Lap()* and Dirichlet noise *Dir()* according to workers' Energy cost E_i^{Taskn} and time consumption $T_i^{Task_n}$. Then SC will add the noise to each worker's result set and store it in the public blockchain (Step 5 to 6). Finally, the system will give a successful message.

Algorithm 3 Block Creation Algorithm in Public Chain

Require: Workers result response Res, Worker's chosen indicator c
Ensure: The response set $\langle W_i, Task_n, DP(E_i^{Task_n}), DP(T_i^{Task_n}) \rangle$

1: Workers see the $Task_n$ and choose to finish the task on the public blockchain.
2: Workers finish the task and send the result $\langle W_i, Task_n, E_i^{Task_n}, T_i^{Task_n} \rangle$ set to SC.
3: SC received enough workers response Res.
4: SC generates the noise $Lap_j(\frac{\Delta Q}{\epsilon})$ and Dirichlet noise $Dir()$ according to workers Energy cost E_i^{Taskn} and time consumption $T_i^{Task_n}$.
5: SC add noise $Lap_j(\frac{\Delta Q}{\epsilon})$, $Dir()$ to $E_t^{i/T}$ and $T_i^{Task_n}$ in the result set to achieve $DP(E_i^{Task_n})$ and $DP(T_i^{Task_n})$.
6: SC stores result set $\langle W_i, Task_n, DP(E_i^{Task_n}), DP(T_i^{Task_n}) \rangle$ on blockchain.
7: **return** Result set $\langle W_i, Task_n, DP(E_i^{Task_n}), DP(T_i^{Task_n}) \rangle$

3.4.2.5 Load Tasks to the Private Blockchain

Because some workers would choose a high level of privacy protection, once they have been selected to complete the assignment, they might choose whether to complete it on a private blockchain. When a private job requires processing on our blockchain network, a private chain is constructed to gather replies and validate transactions. In other words, the private blockchain would be generated dynamically based on the demand for private work. Moreover, the private blockchain employs the PBFT consensus to defend against the Byzantine failure attack. The method of establishing a private blockchain is depicted

in Algorithm 4. The Merkle tree will be described in depth during the sensory data upload session.

Algorithm 4 Load Tasks to the Private Blockchain

Require: *PrivateChain*, Worker's chosen indicator c
Ensure: Successfully information, jwt Token
1: *PrivateChain* = False;
2: $PrivateChain \leftarrow c$;
3: **if** *PrivateChain* = True **then**
4: Load tasks to private blockchain.
5: Merkle tree establish.
6: **return** *PrivateChain*
7: **end if**

3.4.2.6 Upload Sensory Data

After completing the task, workers will attempt to submit the sensory data. In this session, there are two possibilities. The worker selected to establish the findings on the public chain comes first. The second step is for the chosen worker to record the results on a private blockchain. Consequently, we shall describe these two scenarios as follows:

a) Finish on public blockchain: If the worker fails to complete the task before the deadline, the smart contract will automatically end the operation, and the task will fail. If the worker completes the task before the deadline, after establishing the results, the miners will check the data's quality to determine whether or not it is qualified. If the data meets the requester's specifications, the workers will receive the reward, and the hashed findings will be added to the public blockchain.

b) Finish on private blockchain: If the worker elects to publish the results on a private blockchain, a new private blockchain is created. When the worker completes the task before the deadline, the agent will first validate the answer; if the answer is acceptable, the answer of the worker will be hashed in a Merkel tree and just the hash will be stored in the public blockchain.

The main process of this session would be present in Algorithm 5.

3.4.2.7 Payment

When a task is complete, all public and private blockchains will be upgraded proportionally. The payment procedure would be carried out via the smart contract. Due to the

Algorithm 5 Upload Sensor data

Require: *PrivateChain*, Answer of workers *Ans*, agent answer of data quality *DQ*
Ensure: *PublishSuccess*

PublishSuccess = *False*;
PublishSuccess ← *DQ*;
if *PrivateChain* = *True* **then**
  if PublishSuccess = False **then**
    return *Publish falis*
  end if
  return PublishSuccess
else
  if VerifiedData = False **then**
    return *Publish fails*
  else
    Workers finish the task and send the result $\langle W_i, Task_n, E_i^{Task_n}, T_i^{Task_n}, Ans\rangle$ set to SC.
    SC received enough workers response Res.
    SC generates the noise $Lap_j(\frac{\Delta Q}{\epsilon})$ and Dirichlet noise $Dir()$ according to workers Energy cost E_i^{Taskn} and time consumption $T_i^{Task_n}$.
    SC generate result set $\langle W_i, Task_n, DP(E_i^{Task_n}), DP(T_i^{Task_n}), Ans\rangle$.
    SC upload *hash* to public blockchain.
    return *PublishSuccess*
  end if
end if

fact that the worker's account information is not tied to any personal information in the blockchain network, the smart contract will pay workers in bitcoin via their public keys without revealing their true identity.

3.5 Privacy and Security Analysis

3.5.1 Privacy analysis

In this work, we investigate a few common privacy issues in the conventional crowdsourcing system and examine how our system could defend against these attacks.

- The server knows the worker's current location when a worker submits an interest in the task. Typically, in a conventional crowdsourcing system, the worker must provide their current location in order to receive more acceptable projects. However, our solution is distinct in that the server assigns the task and the worker need

simply submit their chosen location. Therefore, the worker's location privacy will not be compromised. Since the entire system relies on the blockchain, all workers within the blockchain are anonymous. According on the worker's energy cost and time consumption, the assignment is assigned.

- The server knows the worker's further location when they are chosen to finish tasks. In a conventional crowdsourcing system, when the worker decides to complete the task, he or she will travel to the precise area to gather the data. However, in our proposed system, all workers are anonymous in the blockchain system; even if an attacker knows the worker's task, they do not know the true identity of the worker with which it is associated. Moreover, when workers are selected to complete the assignment, they may choose whether to complete it on a public blockchain or a private blockchain. If the worker decides to complete the assignment on a private blockchain, the Merkel tree will protect the worker's location privacy, resulting in a better level of privacy protection. The hash will conceal all worker-related data.

- The server knows the worker's previous location from the payment information. In a conventional crowdsourcing system, all workers are required to provide their personal information, such as their real name and bank account number, in order to be paid after the assignment is complete. With contrast, in the solution we offer, the entire payment procedure will be conducted automatically in accordance with the smart contract. The worker will not be required to provide their real identifying details in order to be compensated. They will automatically receive the bitcoin upon completion of the assignments. Therefore, it is difficult to match the assignment with the appropriate workers. Therefore, our system might retain the worker's previous location.

Based on these evaluations, we assert that our suggested system could maintain the location information of workers regardless of their current, future, or previous location. Our proposed system could prevent the disclosure of location privacy in conventional crowdsourcing systems.

3.5.2 Security analysis

In a conventional crowdsourcing system, the majority of security issues will occur during the payment process. To date, the most effective solution to these payment process security issues has been to find a trustworthy third party. Using a third party, however,

will still result in complications. For instance, even if the worker completes the task on time and the data passes the data quality test, it is difficult to guarantee that the third party will not underpay or pay the worker late.

However, as previously said, our suggested system is a combination of blockchain technology and crowdsourcing; all regulations are automatically executed by smart contracts. In our proposed system, there is no third party. When the smart contract's conditions are met, the payment task will execute automatically, and the worker will receive the cryptocurrency immediately.

Malicious workers also pose a threat to the security of traditional crowdsourcing platforms. Some malevolent workers will accept the assignment but will not complete it or will provide unqualified data. This means that a worker who receives the identical task but submits valid data on time will not be eligible for rewards. In a conventional crowdsourcing system, a reputation system is typically implemented to address these issues. The task will be assigned to the worker in the system with the best reputation history. On the other hand, it will create difficulties for new workers who do not yet have a solid reputation.

However, our proposed solution could alleviate this issue; the server will assign tasks based on the energy cost and time consumption of the end user. Even if you are a new worker with a poor reputation, you will be given the job. As a solution to the problem of workers not submitting qualified data, our proposed system requires customers to deposit cryptocurrency as a margin. The consensus procedure will ensure that the smart contract operates efficiently. If the worker gave invalid data, the smart contract would automatically send the margin to the requester. The same applies to the requester; if the requester is deemed malevolent, the deposit is forfeited.

Another potential security issue in a typical crowdsourcing system is that malevolent workers could download and reuse the system's sensitive data. However, all information in the blockchain network is encrypted, and all user identities are encrypted by X509 certification, so the attacker will not be able to determine the user's true identity.

3.6 Results and Analysis

The software configuration comprised Hyperledger Fabric Version 2.X and Hyperledger Caliper 0.4.1. The hardware configuration included a MacOS 10.15.7 operating system, a 2.2GHz CPU, an Intel Core i7, and 32 GB of 2400MHz LPDDR4 memory. The deterministic programming for the blockchain network was developed in Go, running on Visual

Studio Code.

We may obtain matrices such as send rate, maximum latency, minimum latency, average latency, and throughput from the Hyperledger Caliper. Comparing the original system without our privacy-preserving approach to our differential private crowdsourcing method, we utilised three primary matrices: average delay, throughput, and send rate. The primary variable we analyse is the number of 50, 100, 200, 400, 500, and 800 individual blocks.

3.6.1 Latency

When seen in Figure 3.3, as the block size rose, the average latency of both the existing system and our new system increased. The corresponding unit of latency is s. Before 200 blocks, however, the latency of these two techniques is essentially identical. From 200 block increments to 400 block increments, these two systems have a nearly identical upward pattern. Our proposed method has a greater rise trend than the previous system after 400 blocks.

The average latency of our proposed method is slightly more than that of the original system since we have employed a mechanism that protects privacy, which requires more time to calculate. In addition, we may discover that our proposed technique performs better with smaller block sizes. Even though it demonstrates a bigger rise tendency after 400 blocks, it is still acceptable because the worker's anonymity will be preserved. However, merely the line graph was unable to demonstrate the distinction clearly, so we drew a box plot to provide more information.

From Figure 3.4, it is virtually identical to what we found in the line graph. In this diagram, the blue box shows the system without a technique for protecting privacy, whereas the green box displays our proposed method. We utilised not only the average latency, but also the maximum and minimum latency values. It would be preferable to display the latency's entire shift trend. Both of these delay measures exhibit an upward trend from 50 to 800 per block. However, it will display certain outliers that have a considerable impact on the mean latency. The reason why there are outliers in the average delay is because the hardware will affect the blockchain network. Figure 3.4 reveals that the median average latency before 200 blocks is roughly identical.

Despite the fact that in Figure 3.3, the average latency of our proposed method is greater than that of the original system, when all the maximum latency, minimum latency, and average latency are taken into account, the latency of our proposed method performs better than that of the original system. In other words, the latency performance

of our proposed solution is more consistent than that of the original system. In addition, we discovered that, when the block size increases, our proposed technique would have a superior latency performance than the original system when all maximum latency, minimum latency, and average latency are taken into account. From Figures 3.3 and 3.4, we can deduce that the average latency of our suggested technique is slightly higher than that of the standard system, but is acceptable given that it offers a higher level of privacy protection.

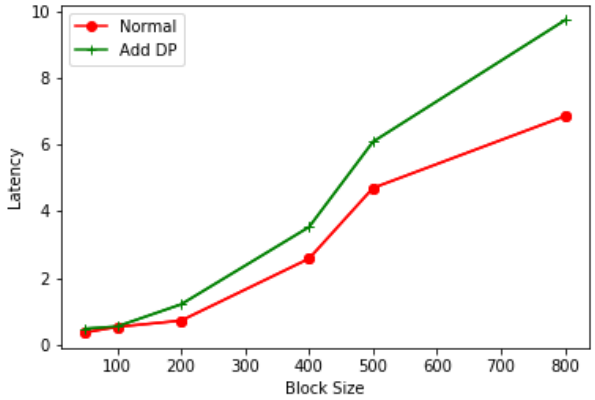

Figure 3.3: The latency comparison of our proposed method with the baseline

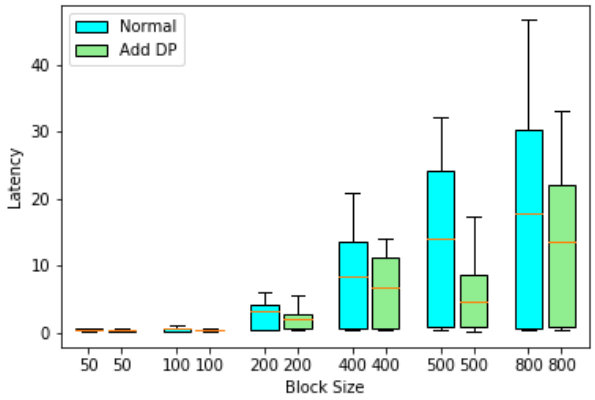

Figure 3.4: The box plot comparison of latency

3.6.2 Throughput

In Figure 3.5, we can see that both systems exhibit a quick decline from 100 to 400 per block. Interestingly, the throughput of our suggested method is always greater than that of a system without any privacy-protecting method. Taking into account the previous results for average latency, our suggested method will have somewhat higher average latency but a higher throughput than the system without a privacy-preserving mechanism.

The future IoE network will be more suited to the strategy we suggest. In future IoE networks, the latency will reduce even further, while the number of connected devices will skyrocket. At this time, the system's throughput will be of utmost importance. Consequently, our proposed approaches are appropriate for future IoE networks. Similar to the average delay section, a box plot is required to observe more information.

Box plot of the throughput for two systems is depicted in Figure 3.6. In this diagram, the blue box shows the system without a technique for protecting privacy, whereas the green box displays our proposed method. Identical to what we discovered in the line graph. Our proposed solution always yields a greater median than the original system.

In addition, the minimum throughput of our suggested technique is occasionally more than the maximum throughput of the original system, such as when the block size is 400, 500, or 800. Although Figure 3.5 and Figure 3.6 both show fluctuations in the curves, these are primarily due to the computer's environment. From this section, we can conclude that the performance of our suggested system's throughput is superior to that of the original system, making it more suitable for use in IoE scenarios.

3.6.3 Send Rate

Regarding transmit rate, we have configured both settings at 40 TPS, hence the send rate of the two systems should be close to 40 TPS. Figure 3.7 demonstrates that the send rate of both systems falls as the block size increases from 50 to 200. However, both systems are still approximately 40 TPS. The increase in block size has little effect on the send rate.

In addition, after 500 block sizes, the transmission rate of our proposed approach is almost identical to that of the original system. Our new solution performs better than the previous system prior to 200 blocks. Similar to the average latency and throughput section, a box plot has been generated for the subsequent data.

Similar to the previous box plot, the blue box in Figure 3.8 shows the system without

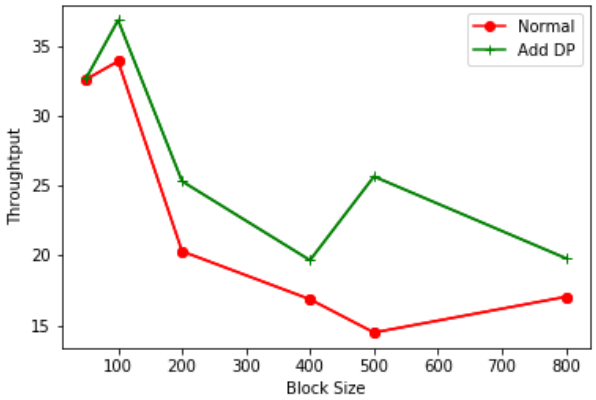

Figure 3.5: The throughput comparison of our proposed method with the baseline

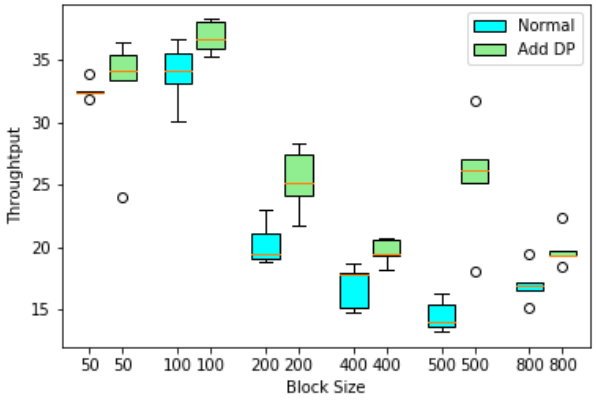

Figure 3.6: The box plot comparison of throughput

a privacy-preserving approach, while the green box represents our suggested method. Figure 3.8 demonstrates that the median send rate for the two systems is practically identical when the block size is less than 100. After 200 blocks, our suggested approach's send rate will fluctuate marginally more than the system without the privacy-preserving mechanism.

However, the lowest send rate of our proposed approach is still 36 TPS. From Figures 3.7 and 3.8, we can deduce that the block size does not greatly affect the send rate, indicating that when our approach is used in future planned IoE networks, it will continue to exhibit a good send rate performance.

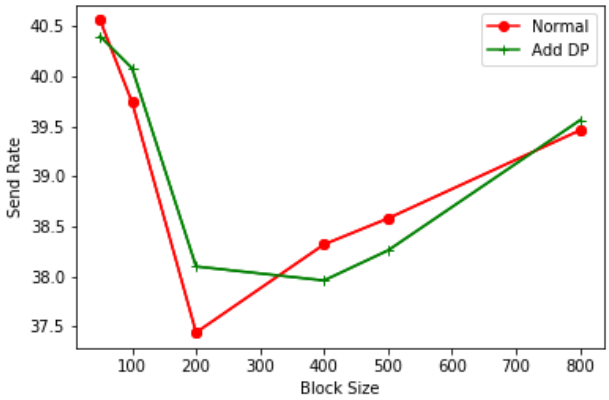

Figure 3.7: The send rate comparison of our proposed method with the baseline

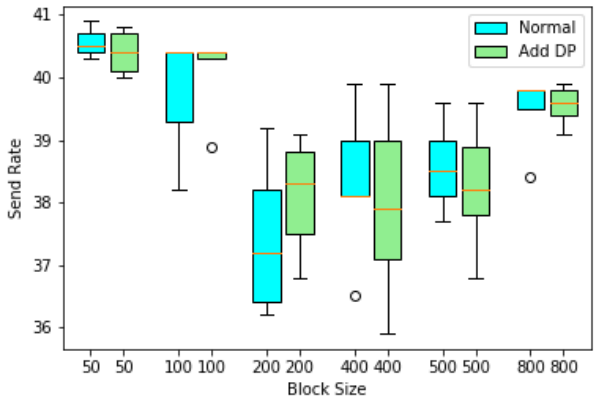

Figure 3.8: The box plot comparison of send rate

3.6.4 Crowdsourcing performance

Regarding the evaluation of the crowdsourcing system's performance, there are numerous metrics to consider. Cullina et al.[15] have separated crowdsourcing metrics into four categories: crowd membership, crowd platform, crowd incentive, and crowd interactions. We shall discuss these indicators individually in the sections that follow:

- Crowd participants level: Participant is the most essential component of the crowdsourcing system. At this level, the identity and quantity of participants are the two most important indicators. As the number of users increases on the platform we offer, only the communication time would be affected by the use of blockchain technology. No other performances will be affected.

Moreover, in order to join the blockchain network, all participants must first register and obtain both private and public keys; thus, the identity of participants issue would be eliminated.

- Crowd platform level: Reliability is one of the most critical metrics for crowdsourcing platforms. Because the blockchain network lacks a third party, all processes are executed automatically. The platform is far more reliable than conventional crowdsourcing platforms. In addition, both privacy and security might be maintained by our differential privacy strategy.
- Crowd incentivisation level: The reward is also a fundamental component of conventional crowdsourcing. In our platform design, the payment procedure will be carried out using a smart contract. It will not have the same unpaid issues that plague conventional crowdsourcing platforms.
- Crowd interactions level: According to our Algorithm 2, the assignment will be publicised to the entire blockchain network, hence the success rate of allocating the task will be greater than with a conventional crowdsourcing approach.

3.6.5 Analysis Conclusion

In conclusion, our proposed method may have a good throughput performance and maintain the same send rate performance, albeit with a slight increase in latency, although the block size will get larger. However, the problem of latency will be effectively resolved in future IoE networks. Considering the overall findings, the delay issue may be acceptable if the worker's anonymity is protected.

3.7 Summary

When merging crowdsourcing and IoT, the privacy and dependability of IoT devices and data must be taken into account. Using a combination of public and private blockchains and a differential privacy mechanism, our suggested technique, a blockchain-based crowdsourcing system, may successfully address the identity and credibility challenges. In addition, future IoE environments could benefit from the scalability performance of our proposed technique. This chapter offers differentially private blockchain-based crowdsourcing for the Internet of Things (IoT) network. We began by providing some background information and introducing crowdsourcing, blockchain, and differential

privacy. After detailing our models, we highlighted obstacles associated with the deployment of the blockchain-based crowdsourcing system in IoT networks. Below is an evaluation of the privacy and security of our system, as well as our suggested differential private crowdsourcing. Finally, we examined the performance analysis of our system and determined that it is suitable for future IoE networks. We believe that this discussion will generate interest and promote more research into the blockchain-based crowdsourcing technique that will be utilised in future IoE networks.

CHAPTER 4

BLOCKCHAIN EMPOWERED MULTI-AGENT SYSTEMS: ADVANCING IOT SECURITY AND TRANSACTION EFFICIENCY

4.1 Introduction

In our increasingly digital world, cutting-edge technologies like the Internet of Things (IoT), multi-agent systems, and blockchain are spearheading various aspects of contemporary life [77]. These domains bring unique capabilities to the table: the widespread reach and interconnection of IoT, the autonomy and adaptability inherent in multi-agent systems, and the secure, traceable nature of blockchain technology. As technological boundaries are incessantly pushed, the significance of exploring the synergistic interactions between these areas cannot be overstated [56].

The IoT revolution has redefined our interactions with the immediate physical environment. By creating a network of interlinked devices that can communicate and share data, IoT has enabled unparalleled levels of connectivity and automation [78]. From enhancing the functionality of residential spaces to facilitating industrial automation, IoT is now integral to a multitude of cross-industry applications. However, the flip side of increased interconnectivity is the daunting task of managing dispersed, complex data while ensuring robust security and privacy measures [65].

Concurrently, the field of artificial intelligence has witnessed impressive strides in the development of multi-agent systems [25]. Comprising numerous autonomous entities

or agents capable of environmental perception, information processing, and independent decision-making, these systems are engineered to achieve specified objectives. The extent of an agent's capabilities can vary from executing simple tasks to tackling intricate problems, depending on the degree of sophistication.

Despite the enhanced efficiency and adaptability that multi-agent systems offer, particularly in complex and distributed environments, they are not without challenges [39]. A notable concern is the management of information or advice exchange among agents. Given the independent operation of agents, which may often have distinct goals, it is critical to ensure the quality, authenticity, and effectiveness of shared advice.

Blockchain technology, primarily recognized for underpinning cryptocurrencies, fills this gap. Its potential, however, transcends digital currencies [14]. It is a decentralized, distributed digital ledger technology that securely documents transactions across several computers, essentially making recorded data impervious to tampering. This makes blockchain an ideal mechanism for improving traceability and securing information exchange in a distributed system.

The amalgamation of multi-agent systems, blockchain technology, and IoT is an exciting frontier, capable of managing the intricacies of distributed data processing and decision-making within IoT systems [40]. Secure and transparent information exchange in an IoT system via blockchain can substantially enhance system robustness and reliability. Concurrently, the autonomous decision-making capability of multi-agent systems can optimize system operations based on real-time data garnered from IoT devices [44].

This Chapter presents an innovative methodology that synthesizes multi-agent systems with blockchain technology in the context of IoT, aiming to enhance the traceability of advice shared among agents within a multi-agent system via a public blockchain network. This integration fosters an environment conducive to transparent, secure, and efficient information exchange, thereby augmenting the overall performance and reliability of the IoT system.

In this pursuit, we put forth a suite of algorithms that orchestrate various facets of the advice exchange process within the blockchain network, including agent registration, advice request publication, advice sharing, and agent rating. In this system, each agent (potentially an IoT device or a virtual agent) must initially register with the blockchain network, authenticated via JSON Web Tokens (JWTs). Subsequently, an agent seeking advice publishes a request, which is broadcast across the network. Responsive agents offer advice, and the smart contract processes and uploads this advice onto the public

chain.

To stimulate the sharing of quality advice, we propose a reward-based rating mechanism. Upon advice utilization, the requesting agent transfers a reward to the advising agent via the smart contract, which consequently updates the advising agent's rating based on the received reward. The transparency and immutability intrinsic to blockchain ensure that these ratings are fair, unbiased, and tamper-resistant.

Our approach amalgamates the strengths of IoT, multi-agent systems, and blockchain technology, thereby heralding a new era of secure, transparent, and efficient IoT systems. By illuminating this potent synthesis, we aspire to contribute to the genesis of innovative solutions that fully exploit these technological domains.

Our contributions can be encapsulated as follows:

- This study offers an exhaustive examination of the integration of multi-agent systems, blockchain technology, and IoT. We scrutinize the unique advantages of these technologies and their synergistic potential in managing intricate, distributed data processing and decision-making challenges within IoT systems.
- Our research unveils a collection of novel algorithms dedicated to agent registration, advice request publication, advice sharing, and agent rating within a blockchain-anchored multi-agent system. These algorithms fortify the traceability, quality, and efficacy of advice sharing among agents in IoT landscapes.
- We implement a transparent, reward-based rating mechanism within the blockchain network to promote the exchange of high-quality advice among agents. By fostering an environment that rewards effective advice, we contribute to building a more robust multi-agent system. The secure and transparent nature of blockchain ensures these rewards are fair, unbiased, and tamper-resistant, bolstering the overall credibility and efficacy of the system.

4.2 Multi-agent Preliminary

4.2.1 Multi-agent System

A Multi-Agent System (MAS) is a network of agents that interact and communicate with each other, each pursuing a shared or individual objective [33]. In this context, an agent refers to an entity that can perceive its environment, reason to understand its perception, and take actions accordingly. A formal definition of an agent can be given as follows:

Definition 4. Agent: An agent α is characterized by a tuple $\alpha = (S_\alpha, A_\alpha, P_\alpha, Z_\alpha, \tau_\alpha)$, where:

- S_α delineates the set of all potential states of the agent.
- A_α symbolizes the set of all actions the agent can execute.
- $P_\alpha : S_\alpha \times A_\alpha \to \Delta(S_\alpha)$ serves as the state transition function, where $\Delta(\cdot)$ signifies a set of all probability distributions over S_α.
- Z_α represents the set of all possible observations the agent can make about its environment. An observation can be viewed as a partial or noisy reflection of the actual state the agent is in. Depending on the agent's sensors and the complexity of the environment, the agent might not always observe the exact state. Thus, Z_α encompasses all the interpretations or perceptions the agent can have about its surroundings.
- $\tau_\alpha : S_\alpha \to Z_\alpha$ is the observation function mapping each state to an observation. It signifies how the real state of the agent translates to what the agent observes or perceives.

Definition 5. Multi-Agent System: A Multi-Agent System (MAS) is represented as a tuple $MAS = (\mathscr{A}, \mathscr{E}, \mathscr{R})$, where:

- $\mathscr{A} = \{\alpha_1, \alpha_2, ..., \alpha_n\}$ defines a finite set of agents.
- $\mathscr{E}$ characterizes the environment within which agents operate and interact.
- $\mathscr{R} = \{r_1, r_2, ..., r_n\}$ denotes the set of reward values, where each $r_i \in \mathbb{R}_{\text{reward}}$ represents the reward obtained by agent α_i for a certain action in a particular state. Here, $\mathbb{R}_{\text{reward}}$ symbolizes the set of real numbers, quantifying the rewards obtained by agents for their actions.
- The global state or 'stage' of the MAS, denoted as S_{global}, is an aggregate of the states of each individual agent, i.e., $S_{\text{global}} = S_{\alpha_1} \times S_{\alpha_2} \times ... \times S_{\alpha_n}$.

Within the multi-agent system, agents can communicate and share advice with each other to augment their performance or broaden their understanding of the environment. This information exchange can be formalized as an advice-sharing function.

Definition 6. Advice Sharing: In a multi-agent system, advice sharing refers to the process wherein agent α_i conveys a dynamic piece of information or knowledge, denoted as $Advice_{\alpha_i}$, to another agent α_j at a particular state $State_n$ of the system. As the system evolves, transitioning from one state to another (e.g., from $State_n$ to $State_{n+1}$), the advice might change based on new observations, actions, or outcomes. This advice can encompass insights about the optimal action, environmental perceptions, or any other relevant knowledge that might aid α_j in its decision-making or in comprehending the environment better. Mathematically, this advice-sharing process can be described by the function:

$$f_{adv} : \alpha_i, \alpha_j, State_n \rightarrow Advice_{\alpha_i} \tag{4.1}$$

where both α_i and α_j are elements of the set of agents $\mathscr{A}$, and $State_n$ is from the set of all feasible states (or stages as previously mentioned) within the system.

The advice-sharing mechanism promotes collaboration among agents. When combined with blockchain technology, it ensures the traceability and security of the disseminated advice, thus establishing a dependable and resilient multi-agent system. The following sections delve deeper into how the integration of blockchain technology augments the advice-sharing procedure and overall efficacy of the multi-agent system.

4.2.2 Multi-agent Q-Learning

Q-learning is a model-free reinforcement learning technique that instructs agents to learn an optimal policy via their interactions with the environment [**?**]. In the multi-agent system (MAS) context, this strategy is referred to as Multi-Agent Q-Learning (MAQL), where multiple autonomous agents independently apply Q-learning algorithms to maximize their individual rewards.

Definition 7. Q-Learning: In Q-Learning, each agent α maintains a Q-table $Q^\alpha : S \times A \rightarrow \mathbb{R}$, where S is the set of states and A is the set of actions. The Q-value of a state-action pair (s,a), denoted by $Q^\alpha(s,a)$, is an estimate of the expected future reward when agent α executes action a in state s and subsequently follows the optimal policy. The Q-value is updated iteratively through the Bellman equation:

$$Q^\alpha_{\text{new}}(s,a) \leftarrow (1-\lambda)Q^\alpha(s,a) + \lambda[r + \gamma \max_{a'} Q^\alpha(s',a')], \tag{4.2}$$

where λ is the learning rate, r is the reward received after executing action a in state s, γ is the discount factor, s' is the succeeding state, and the max operation is executed over all actions a' in state s'.

Definition 8. Multi-Agent Q-Learning (MAQL): In MAQL, each agent $\alpha \in \mathscr{A}$ independently learns its Q-table $Q^{\alpha} : S \times A \rightarrow \mathbb{R}$ and updates the Q-values according to its individual rewards. This procedure can be formalized as:

$$Q^{\alpha}_{\text{new}}(s,a) \leftarrow (1-\lambda)Q^{\alpha}(s,a) + \lambda[r_{\alpha} + \gamma \max_{a'} Q^{\alpha}(s',a')], \tag{4.3}$$

where r_{α} is the reward for agent α. It's crucial to note that, from the perspective of an individual agent, the environment in the multi-agent context is non-stationary, as it is also influenced by the actions of other agents.

Q-learning and its multi-agent extension have broad applicability, particularly in dynamic and competitive environments. Integrated with blockchain technology, it becomes a potent tool for enhancing the autonomous and decentralized decision-making process within the MAS.

4.3 Problem Definition and System Model

4.3.1 Problem Definition

Considering the criticality of trust and traceability in multi-agent systems, especially within the Internet of Things (IoT) context, our primary objective is to devise and implement a system that not only prompts agents to share high-quality advice but also guarantees the traceability and integrity of shared advice. Additionally, we aim to assess the agents based on their contributions to the system, encompassing the quality of their advice and their level of engagement.

We tackle this problem by integrating a blockchain network into the multi-agent system to capitalize on its transparency, immutability, and decentralization attributes. The multi-agent system under consideration consists of a set of agents $\mathscr{A} = \alpha_1, \alpha_2, ..., \alpha_n$, where each agent α_i is capable of sharing advice, receiving advice, and carrying out actions based on the received advice.

Formally, our problem is articulated in three distinct parts:

Agent Registration: Every agent α_i is required to register within the system to partake in advice sharing. This registration procedure, fortified and verified by the blockchain network, ensures the genuineness of each agent's participation, maintaining the trustworthiness of the system.

Advice Sharing and Block Creation: When an agent α_i encounters a situation that demands counsel, it dispatches a request to the system. Subsequently, the system

broadcasts this request, inviting other agents to provide their advice. Each shared piece of advice undergoes rigorous logging and endorsement by the blockchain network, ensuring the traceability and authenticity of each recommendation.

Rating Mechanism: After an agent α_i acts upon the advice from its peers, the system embarks on a dual evaluation:

- **Outcome Evaluation:** This can be mathematically formulated as $O : A_{\alpha_i} \times S_{\alpha_i} \rightarrow \mathbb{R}_{\text{reward}}$, where A_{α_i} represents the action space and S_{α_i} represents the state space of agent α_i. The outcome evaluation function, $O(a,s)$, quantifies the tangible rewards, r, that agent α_i accumulates, following the implementation of the advice through action a in state s.

$$O(a,s) = r$$

- **Quality of Advice:** We can denote this as $Q : \mathbb{R}_{\text{reward}} \times A_{\alpha_i} \rightarrow \mathbb{R}_{\text{quality}}$, where $\mathbb{R}_{\text{quality}}$ represents the set of real numbers quantifying advice quality. The quality evaluation function, $Q(r,a)$, derives from the outcome, signifying how beneficial, accurate, and relevant the advice was in assisting agent α_i to achieve better rewards.

The challenge lies in devising a system that seamlessly integrates these three components while ensuring efficient and secure agent registration, traceable advice sharing, and a fair rating mechanism. The system needs to be scalable to accommodate a growing number of agents and transactions, while preserving the integrity and authenticity of shared advice, even in the presence of adversarial attempts. The performance of the system will be assessed based on criteria such as advice traceability, the quality of shared advice, and the fairness of the rating mechanism.

4.3.2 System Model

This section presents the system model that merges multi-agent systems and blockchain to ensure the traceability of advice exchanged by agents within the multi-agent system. We detail three critical components: agents, smart contracts, and the blockchain network.

- Agents: Agents are autonomous entities interacting within the multi-agent system. They possess individual knowledge and expertise, which can be leveraged to offer advice to other agents when requested. Each agent has a unique identifier, referred to as $Agent_{id}$, and is associated with a specific stage in the decision-making process,

referred to as $Stage$. The agents collectively facilitate informed decision-making by sharing and receiving advice.

- Smart Contracts: Smart contracts are autonomous contracts with the terms of the agreement directly written into code stored within the blockchain network. They act as mediators between agents and the blockchain, enabling the execution of predefined actions based on certain triggers or conditions. In our system, smart contracts play a pivotal role in enabling and validating advice sharing among agents. They ensure the integrity and transparency of the advice sharing process by enforcing predefined rules and verifying the eligibility of participating agents.

- Blockchain Network: The blockchain network is the underlying infrastructure of our system, providing a decentralized and unalterable ledger for recording advice shared among agents. It comprises a distributed network of nodes, each maintaining a copy of the blockchain. The blockchain network ensures traceability and transparency of the advice by securely storing the advice sets on the public chain. Each advice set comprises the requester's identifier ($Requester_{id}$), the advising agent's identifier ($Agent_{id}$), the current stage ($Stage$), and the advice given ($Advice$). By utilizing blockchain technology, the system guarantees tamper-proof and auditable records of the advice, allowing for the evaluation and rating of agent suggestions.

- Advice Sharing Algorithm: To illustrate the operation of our system, we present the Advice Sharing Algorithm in Blockchain (Algorithm 8). This algorithm outlines the steps involved in the exchange of advice among agents and its recording on the blockchain network. Initially, agents recognize the need for advice and consult their knowledge to offer suitable suggestions. They then forward the advice sets to the smart contract (SC), which acts as the intermediary between the agents and the blockchain network. The SC verifies the eligibility of the advising agent and uploads the advice sets to the public chain. The requesting agents can then retrieve the advice sets and proceed to the next stage of their decision-making process.

In summary, our system model combines multi-agent systems, smart contracts, and blockchain technology to ensure the traceability and evaluation of advice within the multi-agent system. By capitalizing on the transparency and immutability of blockchain, we safeguard the integrity and accountability of the advice-sharing process, ultimately enhancing the decision-making capabilities of agents.

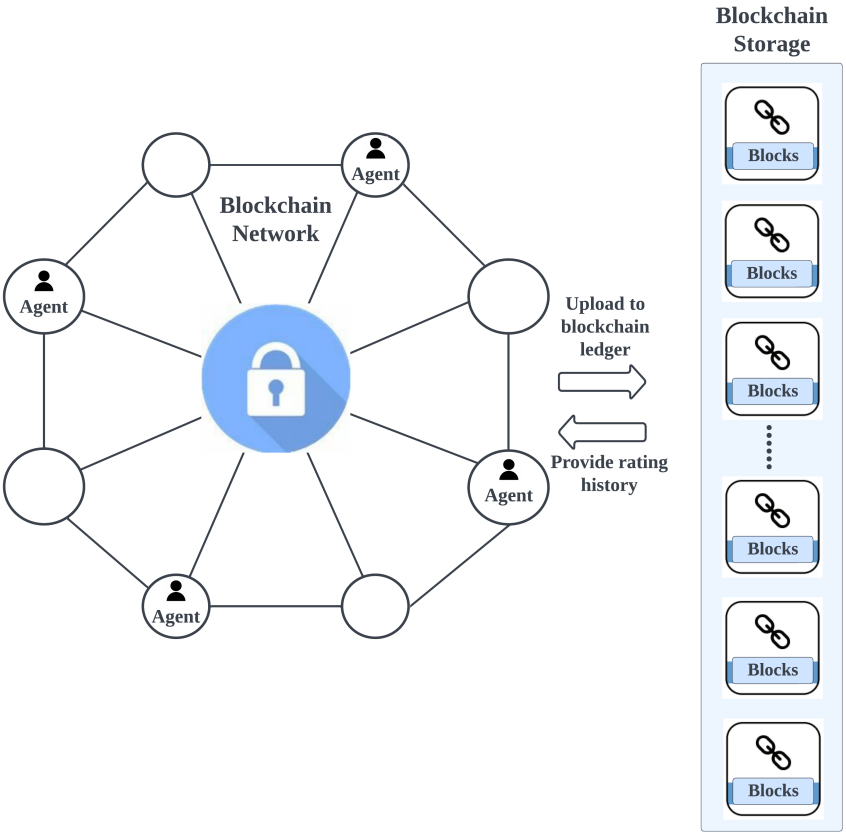

Figure 4.1: Overview of proposed system

4.4 Proposed System

4.4.1 Overview

The proposed system marries multi-agent systems and blockchain technology to construct a secure and efficient framework for collaborative decision-making and information exchange. The system ensures data confidentiality, integrity, and traceability while preserving privacy. Figure 4.1 presents a comprehensive visual of the proposed system. Multiple entities are enrolled within the blockchain network, wherein they contribute their collective expertise by uploading advice. In turn, the blockchain network furnishes a comprehensive record of their rating history. The system employs multiple algorithms to craft a robust and efficient blockchain-based multi-agent system. Further information on the algorithms will be provided in Section 4.4.2.

4.4.2 Implementation of Our Designed System

4.4.2.1 Agent Register

The Agent Registration procedure, defined in Algorithm 6, allows for the inclusion of an *Agent* and its corresponding organization (*Org*) into the blockchain network. This process is vital for enabling the *Agent* to offer advice within the multi-agent system, participate in the rating mechanism, and ensuring the traceability of the advice shared by agents through the blockchain.

Algorithm 6 Agent Register

Require: *Agent, Org*
Ensure: *RegisterSucess, jwt*
1: *RegisterSucess = False*;
2: *Check Org*;
3: **if** $Agent_{id} \in U_{pool}$ **then**
4: **return** $Agent_{id}$ already existed.
5: **end if**
6: P_k,$S_k \leftarrow$ *keyGenerator()*;
7: $jwt \leftarrow P_k$, S_k;
8: $Agent_{id} \leftarrow jwt$;
9: $Pool_{agent} \leftarrow ID_{pool} \cup ID_{ui}$;
10: *RegisterSuccess = True*;
11: **return** *RegisterSucess, jwt*

Initially, the *RegisterSuccess* flag is set to false, indicating that the registration process is yet to be successful. The *Org* is then examined. If the *Agent* ID ($Agent_{id}$) already exists in the pool of unique agent IDs ($Upool$), the algorithm returns an error message notifying that the $Agentid$ is already in use. This feature serves to prevent duplicate registrations and ensure the uniqueness of each agent in the system.

If no duplication is found, the *keyGenerator* function produces a pair of keys: a public key (P_k) and a private key (S_k). These keys underpin the cryptographic measures that secure the registration and subsequent transactions of the *Agent*. The generated keys are then attributed to a JSON Web Token (jwt), a digital token utilized to authenticate the agent's identity.

Next, the $Agent_{id}$ is assigned to the *jwt*, effectively associating the agent's identity with the authentication token. The $Agent_{id}$ is then added to the agent pool ($Pool_{agent}$), which is updated to include the new agent ID.

Upon successful registration, the *RegisterSuccess* flag is set to true. The algorithm then returns the *RegisterSuccess* flag and *jwt*, denoting the successful registration of

the agent into the network with appropriate security measures in place. This sequence of operations allows agents to securely register on the blockchain network, laying the foundation for their participation in advice sharing and rating within the multi-agent system.

4.4.2.2 Smart Contract Creation in Blockchain

In the proposed system that merges blockchain and multi-agent systems, smart contracts serve a critical role in enabling secure and transparent interactions among agents. Smart contracts allow the implementation of predefined actions and uphold agreed rules within the blockchain network.

In the system, agents within the multi-agent system leverage smart contracts to establish a shared protocol governing their interactions. The smart contract acts as a conduit, facilitating the exchange of information, advice, and other pertinent data among agents.

Algorithm 7 and Algorithm 8 illustrate the decision-making processes for agents when they determine the execution of their tasks. These algorithms afford agents the flexibility to make decisions based on their specific needs and preferences.

The fusion of blockchain and multi-agent systems via smart contracts offers several advantages. It heightens the security, transparency, and traceability of agent interactions. The immutability and consensus mechanisms offered by blockchain ensure the integrity of shared information and guard against unauthorized alterations or tampering.

By harnessing smart contracts within the blockchain, the proposed system establishes a robust and efficient framework for collaboration and decision-making within multi-agent systems. The usage of smart contracts promotes trust, transparency, and autonomy among agents, ultimately improving the overall performance and effectiveness of the system.

4.4.2.3 Block Creation Algorithm in Blockchain

The Block Creation Algorithm, detailed in Algorithm 7, is formulated to manage situations where an agent within the multi-agent system needs advice from other agents. This algorithm ensures the secure conveyance and recording of advice requests on the public blockchain network, employing smart contracts (SC).

Initially, an agent encounters a scenario that requires advice from others. The agent then forms a request set $\langle Requester_{id}, Agent, Stage \rangle$, where $Requester_{id}$ is the ID

Algorithm 7 Block Creation Algorithm in Public Chain

Require: Agent meet the situation that needs advice from other agents
Ensure: The request set $\langle Requester_{id}, Agent, Stage \rangle$

PublishSuccess == *False*
Agent meets the situation that needs advice from other agents.
Agent send the request advice set $\langle Requester_{id}, Agent, Stage \rangle$ to SC.
if *Agent jwt token ineligibility* **then**
    return *jwt token* expired
end if
SC publish the set $\langle Requester_{id}, Agent, Stage \rangle$ on the public chain.
PublishSuccess == *True*
return Response set $\langle Requester_{id}, Agent_{id}, Stage, Advice \rangle$

of the agent seeking advice, $Agent$ signifies the advising agent, and $Stage$ denotes the current state or context of the needed advice.

At this juncture, the *PublishSuccess* flag is initially set to False, signifying that the advice request has not been successfully broadcasted on the public blockchain network.

Subsequently, the requesting agent sends the request set to the SC. If the agent's JSON Web Token (jwt) is found to be ineligible (e.g., it's expired), the algorithm stops and returns an error message indicating the *jwt token* has expired. This step ensures that only authenticated agents can request advice, thereby protecting the integrity of the advice-sharing process.

Once the jwt token is verified as valid, the SC publishes the request set $\langle Requester_{id}, Agent, Stage \rangle$ on the public chain, making the advice request available to all other agents. After a successful publication, the *PublishSuccess* flag is set to True, indicating a successful completion of the advice request broadcast.

The algorithm concludes by returning the response set $\langle Requester_{id}, Agent_{id}, Stage, Advice \rangle$. This operation marks the completion of the block creation process, effectively enabling a secure, transparent, and traceable request for advice within the multi-agent system on the public blockchain network.

4.4.2.4 Advice Upload to Blockchain

The Advise Sharing Algorithm, as detailed in Algorithm 8, is designed to handle the exchange of advice between agents within the multi-agent system on the public blockchain network. This process involves a requesting agent asking for advice, other agents providing advice based on their knowledge, and the smart contract (SC) processing and uploading the advice to the public chain.

Algorithm 8 Advise sharing Algorithm in Public Chain

Require: Agent request set $\langle Requester_{id}, Agent, Stage \rangle$

Ensure: The response set $\langle Requester_{id}, Agent_{id}, Stage, Advice \rangle$

1: Other agents realise there are some agents who need advice
2: Other agents check their knowledge and return the *advice*
3: Other agent give advice and send the advice set $\langle Requester_{id}, Agent_{id}, Stage, Advice \rangle$ to SC.
4: **if** *Agent jwt token ineligibility* **then**
5: **return** *jwt token* expired
6: **end if**
7: SC upload the set on the public chain.
8: Request agent download the set and move on next stage.
9: **return** Response set $\langle Requester_{id}, Agent_{id}, Stage, Advice \rangle$

At the beginning of the algorithm, a requesting agent sends a request set consisting of $\langle Requester_{id}, Agent, Stage \rangle$, where the $Requester_{id}$ is the ID of the agent asking for advice, $Agent$ is the advising agent, and $Stage$ signifies the current stage or context of the advice needed.

Upon realizing that an agent needs advice, the other agents within the system use their knowledge to generate appropriate advice. They then create an advice set in the form $\langle Requester_{id}, Agent_{id}, Stage, Advice \rangle$, where $Advice$ is the information or suggestion they are providing.

The advising agents submit their advice set to the Smart Contract (SC). However, if an agent's JSON Web Token (jwt) is found to be ineligible (e.g., expired), the process stops and returns an error message stating that the *jwt token* has expired. This step helps to maintain the integrity and authenticity of the advice being provided and safeguards the security of the multi-agent system.

Assuming that the jwt tokens are valid, the SC uploads the advice set onto the public chain. This action makes the advice accessible to the requester and promotes transparency and traceability.

Subsequently, the requesting agent downloads the advice set from the public chain, enabling it to use the advice for its next action.

Lastly, the algorithm returns the response set $\langle Requester_{id}, Agent_{id}, Stage, Advice \rangle$, signaling the successful completion of the advice-sharing process. This sequence of operations fosters a collaborative learning environment in which agents can exchange advice securely and efficiently, thus enhancing the overall system performance.

4.4.2.5 Rating

The Rating Algorithm, outlined in Algorithm 9, is established to quantify and monitor the performance of agents based on the advice they provide within the multi-agent system on the public blockchain network. The process involves the requesting agent evaluating the usefulness of the received advice and awarding rewards accordingly. The algorithm then updates the rating of the advising agent.

Algorithm 9 Rating algorithm

Require: The response set $\langle Requester_{id}, Agent_{id}, Stage, Advice \rangle$
Ensure: The rating set $\langle Agent_{id}, Times, Reward, Rating \rangle$
1: Agent who needs advice has downloaded the set $\langle Requester_{id}, Agent_{id}, Stage, Advice \rangle$
2: Agent use the *advice* and send the getting reward to SC
3: **if** *Agent jwt token ineligibility* **then**
4: **return** *jwt token* expired
5: **end if**
6: SC update the rate of the agent according to the rewards $\langle Agent_{id}, Times, Reward, Rating \rangle$
7: **return** The rating set $\langle Agent_{id}, Times, Reward, Rating \rangle$

The algorithm starts with the requirement of the response set $\langle Requester_{id}, Agent_{id}, Stage, Advice \rangle$. This set represents the advice given by the $Agent_{id}$ to the $Requester_{id}$ in a particular $Stage$.

The agent in need of advice has already downloaded this set. After utilizing the advice in the respective context ($Stage$), the agent then sends a reward, reflecting the perceived quality of the advice, to the Smart Contract (SC).

The process is halted if the agent's JWT token is found to be ineligible, such as in cases where it has expired, ensuring that only valid and authenticated agents are able to award rewards.

Assuming the JWT token is valid, the SC then updates the rating of the advising agent based on the received reward. This involves creating or updating the rating set $\langle Agent_{id}, Times, Reward, Rating \rangle$, where $Agent_{id}$ is the ID of the agent who provided advice, $Times$ indicates the number of times this agent has given advice, $Reward$ represents the total rewards received, and $Rating$ is the updated rating based on these rewards.

Finally, the algorithm returns the rating set. This allows for continuous monitoring and evaluation of each agent's performance within the system, promoting a merit-based advice-sharing environment. The transparency and traceability of the public blockchain

network support this process, ensuring fairness and reliability in the performance appraisal of agents.

4.5 Privacy and Security Analysis

4.5.1 Privacy Analysis

Privacy preservation is a crucial aspect of the proposed system, given its application in a multi-agent system environment where each agent might possess unique private information. This section analyzes how our model preserves privacy while facilitating advice sharing, traceability, and agent evaluation.

Our privacy protection strategy fundamentally relies on blockchain technology and the use of a cryptographic scheme during agent registration and transaction recording. Privacy analysis primarily focuses on two elements: agent anonymity and advice confidentiality.

- Anonymity of Agents: In our model, each agent is represented by a unique identifier, the $Agent_{id}$, which is generated from the JWT token at the time of registration. This ID is used throughout the system to denote the agent's activities, such as advice request, advice sharing, and rating updates. This design ensures that the agent's real identity is concealed during interactions, thereby providing a level of anonymity.

  It's important to note that while $Agent_{id}$ is visible within the blockchain network, it doesn't lead back to the actual identity of the agent outside the system due to the one-way nature of the cryptographic function used in the JWT generation. Therefore, the blockchain network ensures non-linkability between the $Agent_{id}$ and the real-world identity of the agent, providing strong anonymity protection.

- Confidentiality of Advice: The advice provided by agents is critical information that requires protection from unauthorized access. In our model, the advice content is encapsulated within a transaction and recorded onto the blockchain. Here, the blockchain's immutability property ensures that once the advice is written onto the blockchain, it cannot be altered or deleted, thus maintaining the integrity of the advice.

  Furthermore, we utilize cryptographic techniques to protect the confidentiality of advice content. Specifically, advice shared by an agent is encrypted using the

public key of the advice requester before it is recorded on the blockchain. Only the intended requester, who holds the corresponding private key, can decrypt and access the advice content. This process not only preserves the confidentiality of the advice but also binds the advice to its intended requester, further enhancing the privacy of the system.

In summary, our system effectively guarantees privacy by ensuring the anonymity of the agents and the confidentiality of the advice content. The integration of blockchain technology and cryptographic techniques offers a solid foundation for privacy protection in the proposed multi-agent system. In the next section, we will discuss the security analysis of our proposed system.

4.5.2 Security Analysis

In this section, we focus on the security aspects of the proposed system. Given the potential adversarial threats in a multi-agent system, maintaining a high level of security is crucial. The security properties we evaluate in our model include authenticity, integrity, and non-repudiation.

- Authenticity: Authenticity, in this context, refers to the verification of an agent's identity and the assurance that an advice-sharing transaction is indeed initiated by the claimed agent. Our system ensures authenticity through a secure registration process and the use of JWT tokens.

  When an agent registers, it generates a pair of public and private keys and a JWT token. The JWT token, containing the public key, is recorded in the blockchain network and used to authenticate the agent in all subsequent transactions. This cryptographic technique guarantees that an impersonating agent cannot forge a transaction without the rightful agent's private key.

- Integrity: The integrity of the system pertains to the assurance that the advice shared in the system remains unchanged during storage and transmission. The inherent immutability feature of blockchain technology guarantees the integrity of the system.

  Once an advice-sharing transaction is validated and recorded into a block, it becomes part of the blockchain, making it tamper-resistant. This immutability ensures that the advice, once written onto the blockchain, cannot be modified or

deleted by any agent or third-party entity, thereby preserving the integrity of the advice.

- Non-Repudiation: Non-repudiation ensures that a participating agent cannot deny its actions retrospectively. In our system, the non-repudiation property is achieved by employing digital signature technology in the transaction process.

  When an agent initiates a transaction, such as advice sharing, it signs the transaction with its private key. The signature, along with the transaction content, is recorded onto the blockchain. Since the private key is only known to the signing agent, this mechanism provides robust evidence that the transaction was indeed initiated by the claimed agent, offering non-repudiation assurance.

In conclusion, the proposed system incorporates necessary security measures to tackle potential security threats. The integration of blockchain technology with secure cryptographic methods offers robust protection against common security threats, thereby ensuring the safe operation of the multi-agent system in an adversarial environment.

4.6 Results and Analysis

In this section, we meticulously analyzed the impact of two pivotal variables, namely the map size and the number of agents, on average hits and steps, thereby facilitating a comparative evaluation between our innovative approach and traditional multi-agent systems. Two distinct map sizes were considered: 800*800 and 1200*800. Moreover, we experimented with varying agent numbers, specifically employing 4, 5, 6, and 8 agents, to discern the influence exerted by different quantities of agents.

It's imperative to note that the chosen number of agents is intricately tied to the map size. An overly expansive map, when coupled with a sparse agent population, could detrimentally impact learning efficacy by hindering thorough exploration and exploitation of the environment. Conversely, in smaller maps, an excessive number of agents might lead to overcrowdedness, resulting in potential conflicts, resource contention, and communication bottlenecks, thereby possibly diminishing the overall system performance and learning effectiveness. Consequently, an optimal balance is sought to ensure that the agent population is commensurate with the map size, ensuring effective learning and operation. Hence, we strategically chose to work with 4, 5, 6, and 8 agents to perform a thorough comparison between our method and traditional multi-agent systems, carefully ensuring that the selected agent quantities are computationally efficient and

demonstrative of the distinctive advantages of our approach amid varying environmental complexities. Our selection aims to astutely balance efficient resource utilization and potent demonstration of our method,Äôs scalability and applicability across different scenarios.

In traditional multi-agent systems, certain agents occasionally provide misleading advice when approached by others. Our model, however, utilizes blockchain technology to identify and record such inappropriate advice, thereby deterring agents from dispensing false information. Figures 4.2 through 4.17 illustrate this: the yellow line portrays the performance of a conventional multi-agent system, while the blue line represents the performance of our proposed model.

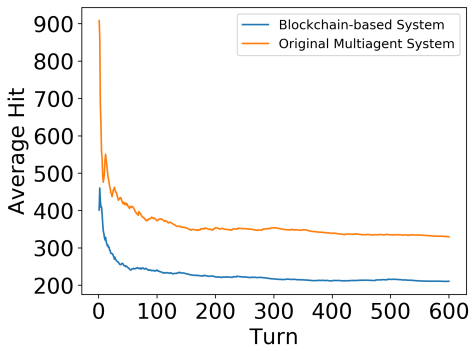

Figure 4.2: Average hits of setting with 800*800 map size and 4 agents

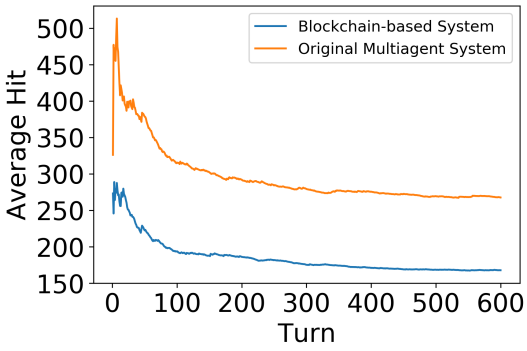

Figure 4.3: Average hits of setting with 800*800 map size and 5 agents

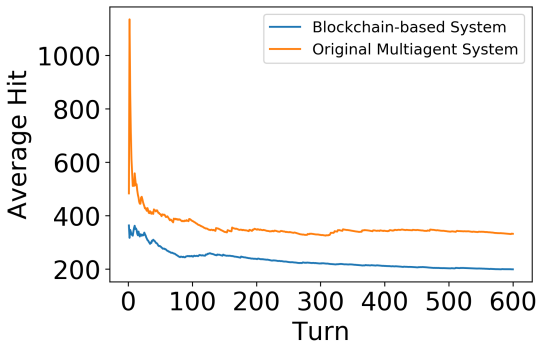

Figure 4.4: Average hits of setting with 800*800 map size and 6 agents

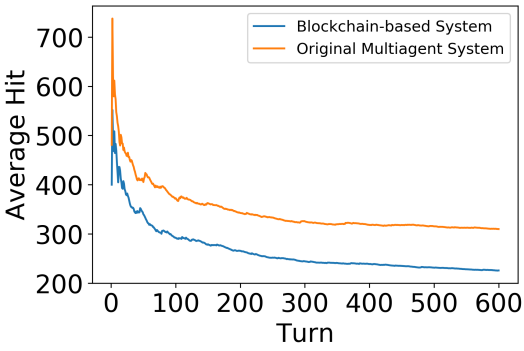

Figure 4.5: Average hits of setting with 800*800 map size and 8 agents

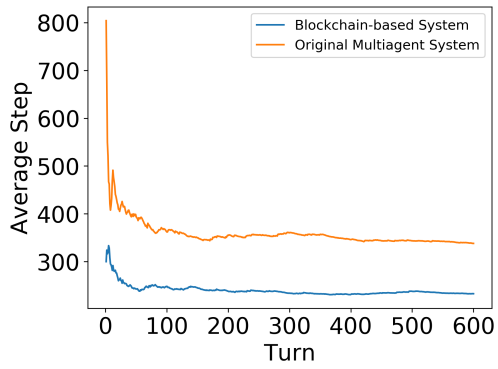

Figure 4.6: Average steps of setting with 800*800 map size and 4 agents

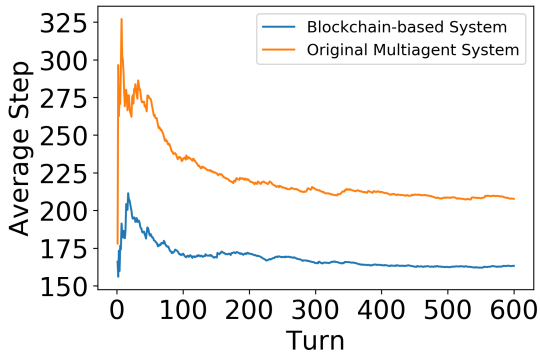

Figure 4.7: Average steps of setting with 800*800 map size and 5 agents

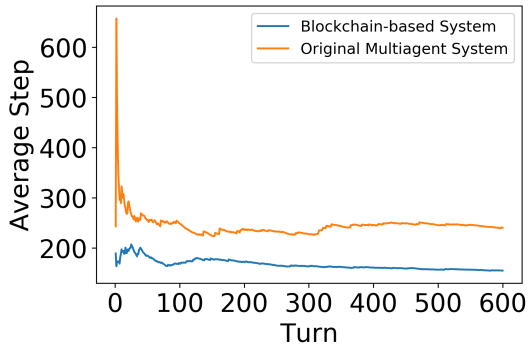

Figure 4.8: Average steps of setting with 800*800 map size and 6 agents

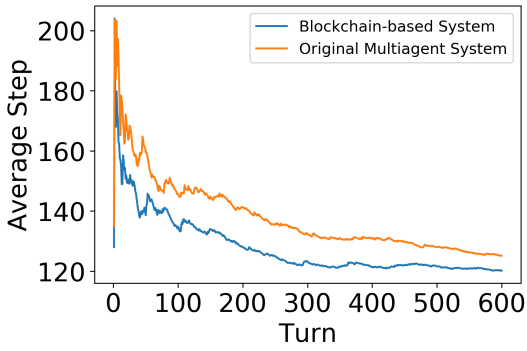

Figure 4.9: Average steps of setting with 800*800 map size and 8 agents

4.6.0.1 Setting of 800*800 map size

- Analysis of Average Hits: Following, we delve into an examination of the average hits under the chosen experimental settings. As illustrated by Figures 4.2 through 4.5, our suggested strategy consistently surpasses the performance of the conventional multi-agent system. Particularly noteworthy is that after 600 cycles, our model tends to yield approximately 200, or even fewer, average hits per round.

  Figures 4.2, 4.3, 4.4, and 4.5 highlight a greater degree of fluctuation in the conventional system when juxtaposed with our proposed model. This volatility is most strikingly displayed in Figure 4.3, where average hits of the original system exhibit significant variations. This irregularity primarily stems from instances when specific agents in the conventional system distribute faulty advice, thereby leading other agents to undertake incorrect actions, resulting in irregularities in

average hits.

Moreover, Figure 4.4 demonstrates that the conventional system shows a rising trend in average hits after 300 cycles. Conversely, our model, after merely 100 cycles, allows the average hits across different agent numbers to converge slowly.

- Analysis of Average Steps: Our focus then shifts towards the examination of average steps within these parameters. Intriguingly, the results bear a close resemblance to those of the average hits. According to Figures 4.6 to 4.9, our innovative approach generated average steps of 232.9, 163.2, 154.9, and 120.2, whereas the traditional multi-agent system produced 338.1, 207.7, 240.2, and 125.2, respectively.

  Of particular note in Figure 4.7, the results of the conventional system display significant oscillations from 0 to 100 cycles, which can be attributed to poor advice from certain agents. This variation is particularly pronounced in comparison to other findings. Figures 4.6, 4.7, and 4.8 reveal an increasing trend in the volatility of the conventional system. In stark contrast, our strategy illustrates a steady convergence pattern after 300 cycles in Figures 4.6 to 4.9.

  An additional observation from Figure 4.9 shows that post 600 cycles, both systems yield similar results of 120.2 and 125.2. This can be primarily attributed to the smaller map size (800x800) for eight agents, hence, the average steps do not markedly differ. However, the trend still reinforces that our proposed solution outperforms the conventional multi-agent system in terms of average hits.

4.6.0.2 Setting of 1200*800 map size

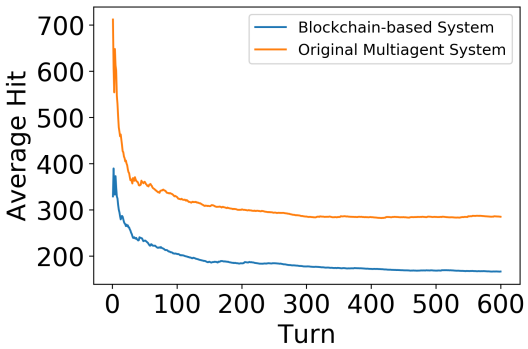

Figure 4.10: Average hits of setting with 1200*800 map size and 4 agents

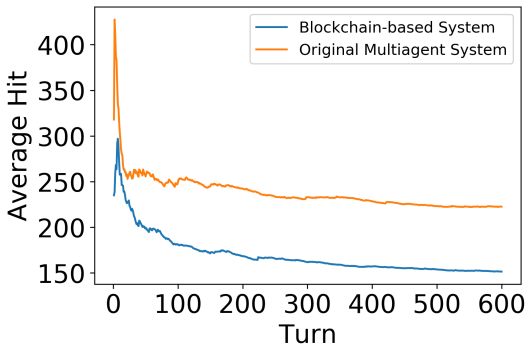

Figure 4.11: Average hits of setting with 1200*800 map size and 5 agents

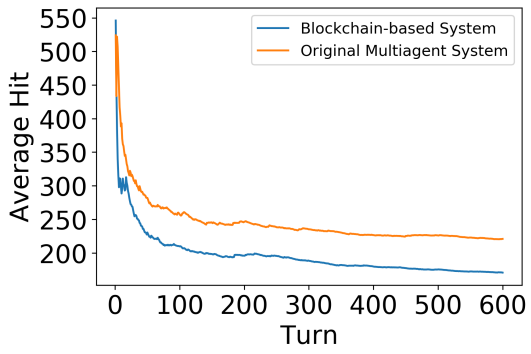

Figure 4.12: Average hits of setting with 1200*800 map size and 6 agents

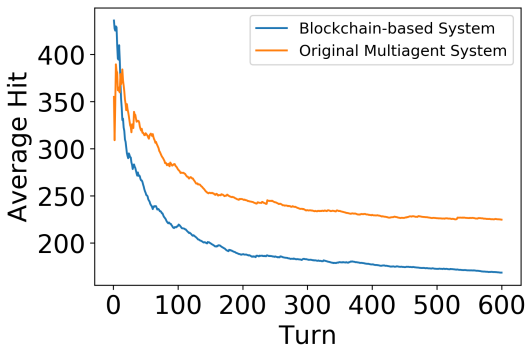

Figure 4.13: Average hits of setting with 1200*800 map size and 8 agents

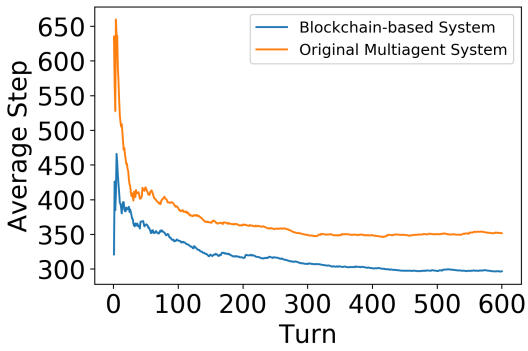

Figure 4.14: Average steps of setting with 1200*800 map size and 4 agents

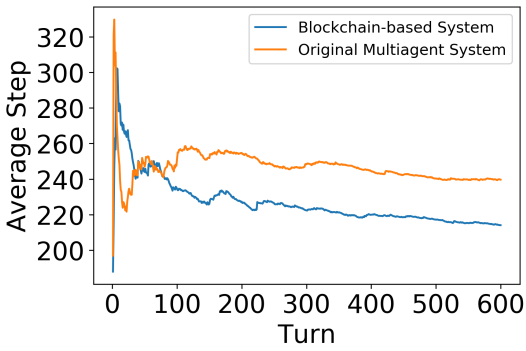

Figure 4.15: Average steps of setting with 1200*800 map size and 5 agents

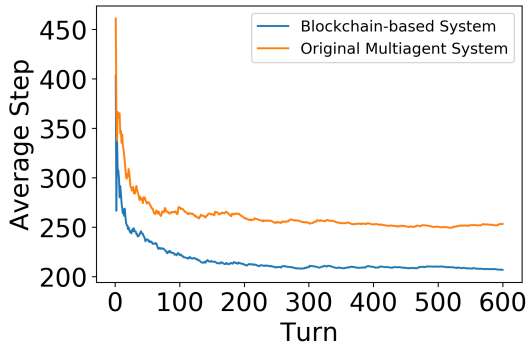

Figure 4.16: Average steps of setting with 1200*800 map size and 6 agents

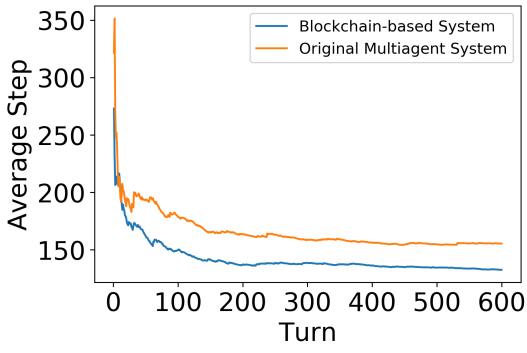

Figure 4.17: Average steps of setting with 1200*800 map size and 8 agents

- Analysis of Average Hits: Probing into the average hits within a setting of 1200x800 map size, we uncover patterns that mirror those observed in the previous 800x800 map size setting. Figures 4.10 through 4.13 elucidate how our recommended strategy continues to outdo the conventional multi-agent system. Notably, after 600 cycles, our method gravitates towards approximately 160 average hits, whereas the original system is skewed towards 240 average hits.

  Figure 4.11 reveals substantial fluctuations in the original system's average hits from 50 to 100 cycles, with no discernible convergence. In stark contrast, our approach demonstrates a consistent trend of convergence across Figures 4.10 to 4.13. Particularly in Figure 4.13, our strategy displays a convergence trend from 0 to 600 cycles, while the conventional system reveals a growing trend between 20 to 30 cycles.

  When compared with the 800x800 map size, it is evident that erroneous advice exerts a lesser influence on the average hits. As the map enlarges, agents are exposed to a more diverse range of scenarios, thus mitigating the negative impact of incorrect advice. This is mirrored in the lower average hits for the 1200x800 setting, regardless of whether it's our proposed approach or the conventional system.

- Analysis of Average Steps: Regarding average steps for the 1200x800 map size, our method maintains its superior performance over the original system. As portrayed by Figures 4.14 to 4.17, following 600 cycles, the average steps of our approach settle at 284.9, 214.1, 206.8, and 132.5, as compared to 351.5, 239.7, 253.3, and 155.3 in the traditional system. Remarkably, our approach exhibits a decreasing trend in average steps across varied agent numbers post 600 cycles, a trend conspicuously absent in the original system.

  Significantly, Figure 4.15 reveals the conventional system charting a fluctuating and ascending trend from 20 to 200 cycles. In contrast, our method maintains a steady convergence trend from 0 to 600 cycles. The volatility in the original system's average steps can primarily be ascribed to faulty advice disseminated by several agents during this stage, leading to the lack of convergence post 100 cycles.

  Comparing the average steps for the 1200x800 setting with the 800x800 setting, it is observable that after 600 cycles, the results are larger for both our proposed approach and the conventional system. This can be traced back to the expanded map size, which necessitates more steps for all agents to complete their tasks.

4.6.0.3 Summary of Analysis

Our empirical evaluations provide compelling evidence of our proposed approach's superiority over the traditional multi-agent system across a multitude of performance measures. In the context of an 800x800 map size, our methodology consistently exhibited strong performance, particularly after 600 cycles, during which it regularly returned fewer average hits and steps compared to the traditional system. In addition, our method displayed a notable reduction in volatility and exhibited a convergence trend, underscoring its resilience against poor advice and errors induced by agents.

When examined under the more expansive 1200x800 map size framework, our system sustained its leading performance, recording lower average hits and steps than the conventional system after 600 cycles. Despite the growth in map size, which introduced a broader variety of scenarios and diluted the effects of faulty advice, our method maintained a lower average hit count. This outcome emphasizes the robustness of our multi-agent system's advice sharing protocol.

Comparative analysis of results across different map settings unveils that as the map size escalates, the steps required for agents to accomplish their tasks also increase, regardless of the system employed. However, our proposed approach exhibits a significantly slower growth rate of required steps than the conventional system. This observation reinforces the efficiency and adaptability of our methodology across diverse scenarios.

In conclusion, our experimental findings offer compelling validation of our proposed approach's efficacy in meeting its goals - minimising the count of hits and steps, safeguarding robustness against poor advice, and demonstrating efficient adaptability in variable scenarios. As such, it establishes a promising foundation for future investigations and advancements in the field of blockchain-integrated multi-agent systems.

4.7 Summary

In this chapter, we have unveiled an innovative model that successfully integrates multi-agent systems and blockchain technology, nested within the broad context of the Internet of Things. Our proposed solution provides a resilient and effective mechanism for secure, autonomous interactions among disparate agents within an IoT setting. The empirical data gathered emphatically indicate the significant efficiency and adaptability improvements offered by our model over conventional systems. These improvements encompass reductions in hits and steps, robustness against inadequate advice, and

commendable scalability in handling extensive scenarios. These outcomes serve as an affirmation of the practicality and efficacy of our system.

CHAPTER 5

PUBLIC AND PRIVATE BLOCKCHAIN INFUSION: A NOVEL APPROACH TO FEDERATED LEARNING

5.1 Introduction

The realization of the Internet of Everything (IoE) is steadily progressing due to the rapid expansion of the Internet of Things (IoT) . A crucial aspect of the IoE is the interconnectedness of all entities within the network. Although this simplifies life for individuals, it also results in numerous devices connecting to the internet and generating massive amounts of data.

Federated learning (FL) has emerged as a promising technique for training machine learning models using decentralized data sources, such as those found in IoT devices [59, 67, 73]. FL enables collaborative model training without requiring data sharing or centralization, addressing critical concerns regarding data privacy and security [7]. However, the communication and computational overheads associated with FL become a significant hindrance as the number of IoT devices expands, particularly when handling sensitive data.

Blockchain technology, a distributed and secure system for data storage and management, has been employed across various industries due to its openness, immutability, and security. Integrating FL with blockchain technology has been proposed as a solution to these challenges, as it provides a decentralized and secure platform for data sharing, verification, and payment [37]. The fusion of FL and blockchain allows for a more

efficient and secure method to train machine learning models on decentralized data sources. Blockchain technology can also offer a more scalable alternative for managing and distributing models developed with FL, particularly in IoT scenarios with numerous resource-constrained devices.

Prior research has explored the integration of FL and blockchain, proposing different approaches, such as utilizing smart contracts to govern the FL process on a blockchain [71], or employing cryptographic techniques to ensure FL's security and anonymity [70]. Lu et al. [49] presented the first blockchain-powered secure data sharing architecture for distributed parties. However, the fusion of public and private chains in federated learning has yet to be thoroughly examined.

Designing an efficient and secure method for managing models on a blockchain is a primary challenge when merging FL with blockchain technology. This requires addressing various factors, including consensus processes, transaction speed, storage capacity, and privacy concerns. Furthermore, managing the models and ensuring their security and privacy differ depending on the use of public or private blockchains.

We propose a groundbreaking method that incorporates FL with both public and private blockchains to address these challenges. Specifically, we recommend employing private blockchains to allow smaller entities, such as corporations, to train models internally using FL before sharing the results on a public blockchain. The trained models will be stored on the public blockchain, where other organizations can use them for additional training and evaluation. This method not only lessens communication and computational demands on devices but also ensures data security and privacy, as each organization maintains control over its private blockchain. Figure 5.1 offers an overview of our proposed system. In this figure, two types of blockchain networks are depicted: the public blockchain and the private blockchain. The requester first sends the global model to the public blockchain. Subsequently, the agent uploads this global model to the private blockchain, allowing the slower client to perform model aggregation. Once the epoch matches the predetermined epoch setting in the private blockchain, the agent transfers the model from the private chain to the public blockchain for further model aggregation.

Our contributions are in followings:

- We present a novel proposition for a federated learning-based blockchain network that enables the establishment of a private chain dedicated to the model aggregation process within federated learning. Our framework harnesses the advantageous

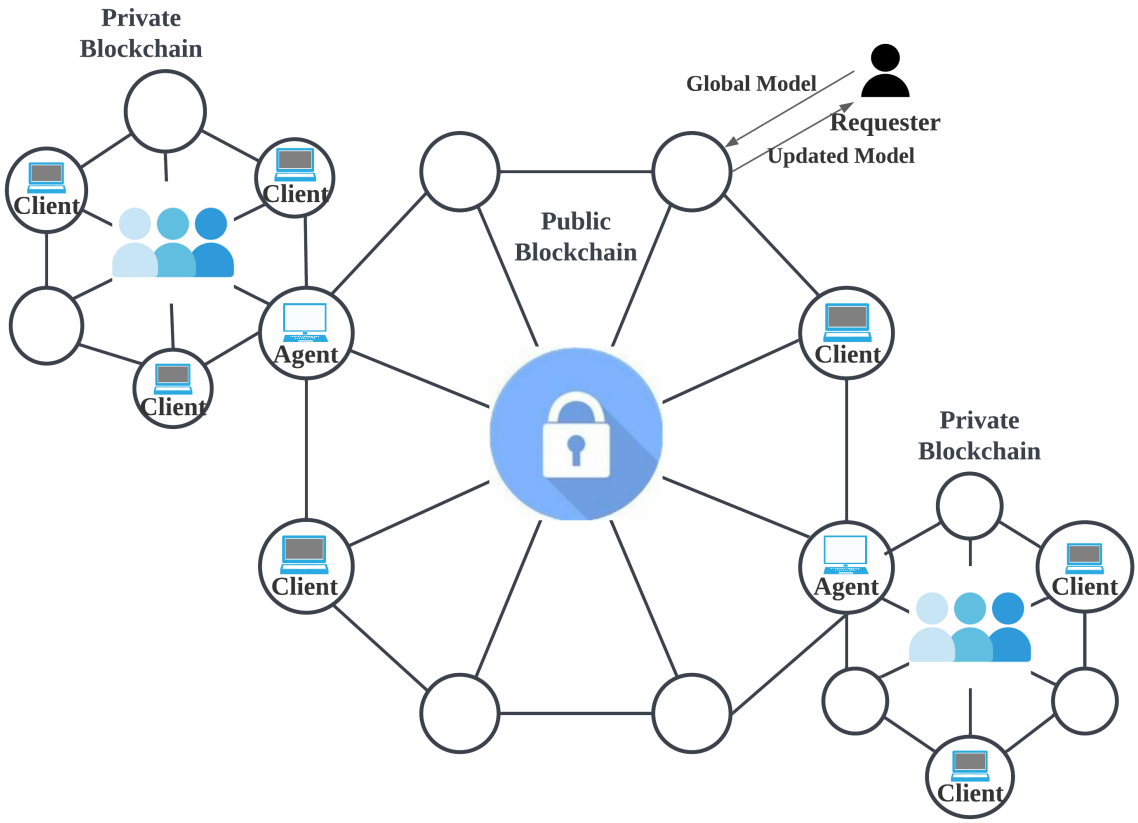

Figure 5.1: Overview of our proposed system

fusion of public blockchain and private chain technologies, thereby augmenting the performance of the federated learning training process.

- Our proposed system demonstrates enhanced accuracy and convergence rates compared to traditional federated learning.
- We further evaluate the performance of our proposed system compared to the conventional blockchain network by examining latency, throughput, and send rate metrics.

5.2 Federated Learning Preliminary

Federated learning is a decentralized machine learning technique designed to tackle the problem of private data leakage that may arise during collaborative model training across multiple devices. Consider N clients with respective datasets D_1, D_2, ..., D_N.

Traditional machine learning would require collecting these datasets at a central server for model training, potentially exposing sensitive information and resulting in privacy breaches. In contrast, federated learning enables local model training, reducing the need for data sharing.

Federated learning typically consists of three stages:

- The central server selects participating clients for the current training round and shares the present global model, denoted as $Model_g$, with them.
- Each participating client, C_i, independently computes a local model, denoted as $Model_p$, using their local dataset D_i, containing n_i samples. The local model is then transmitted back to the central server.
- The central server aggregates local models from each participating client and generates a new global model, denoted as $Model_g$. If the accuracy of the new global model meets a predefined threshold, the training process stops; otherwise, the process continues for the next training round.

The local model learning process can be described as minimizing the local loss function, as shown in Equation (5.1).

$$F_i(w) = \frac{1}{|D_i|} \sum_{j \in D_i} f_j\left(w, x_j, y_j\right) \tag{5.1}$$

where f_i represents the loss function of the $i-th$ client, x and y are sample indices, $f_j(w, x_j, y_j)$ is the loss function on data sample (x_j, y_j) with parameter vector w, and $|D_i|$ is the size of data samples in D_i.

To synchronize the learning across all clients and achieve a global model, the global loss function is defined as the weighted sum of all local loss functions. This global loss function can be expressed as shown in Equation (2):

$$F(w) = \sum_{i}^{n} p_i F_i(w) \tag{5.2}$$

where n is the total number of clients, and p_i is the fraction of the total data that client i possesses.

The learning stages persist until the loss function converges or reaches the maximum number of iterations or training time allowed. To assess the effectiveness of federated

learning algorithms, we consider the algorithm to have achieved ϵ-accuracy loss if Equation (3) holds:

$$|V_{Fed} - V_{Sum}| < \varepsilon \tag{5.3}$$

where V_{Fed} and V_{Sum} represent the accuracy of federated learning algorithms and conventional distributed learning methods, respectively.

5.3 Problem Definition and System Model

5.3.1 Problem Definition

In traditional federated learning, as depicted in Figure 5.2, if a fast device and a slow device have similar data distribution, the central server can transfer a high-quality feature extractor from the fast device to the slow device. This transfer mechanism accelerates the training process for the slower device and promotes quicker convergence of the global model.

However, challenges arise in the traditional federated learning model (Figure 5.2). Specifically, communication bottlenecks occur when the central server is waiting for slower devices to complete their training. This waiting period can significantly delay the aggregation and update of the global model, especially when there is a broad disparity in computational power among the devices. Some researchers have turned to game theory to optimize this waiting dilemma. Still, in this chapter, we present a novel approach grounded in blockchain technology. By allowing devices with limited computational power, or entities managing a large fleet of devices, to form private chains, we can streamline the training process.

In the proposed blockchain-based federated learning system, a set of clients, denoted as $< C_1, C_2, ..., C_n >$, participate. Each client, upon registration, receives a pair of cryptographic keys: a public key P_k and a private key S_k. After obtaining the global model, denoted as $Model_g$, and executing the federated learning process, clients update the blockchain with a new dataset $< B_{ID}, Weights, Gradient, Model, ID >$. Here, B_{ID} denotes the block ID, with weights, gradient, and model being essential parameters for the global model update. The term ID corresponds to the client's unique identifier.

The computation bottleneck in federated learning arises when devices with varying computational capacities participate. For devices with limited processing power, executing the learning algorithm can be time-intensive. Communication bottlenecks, on the

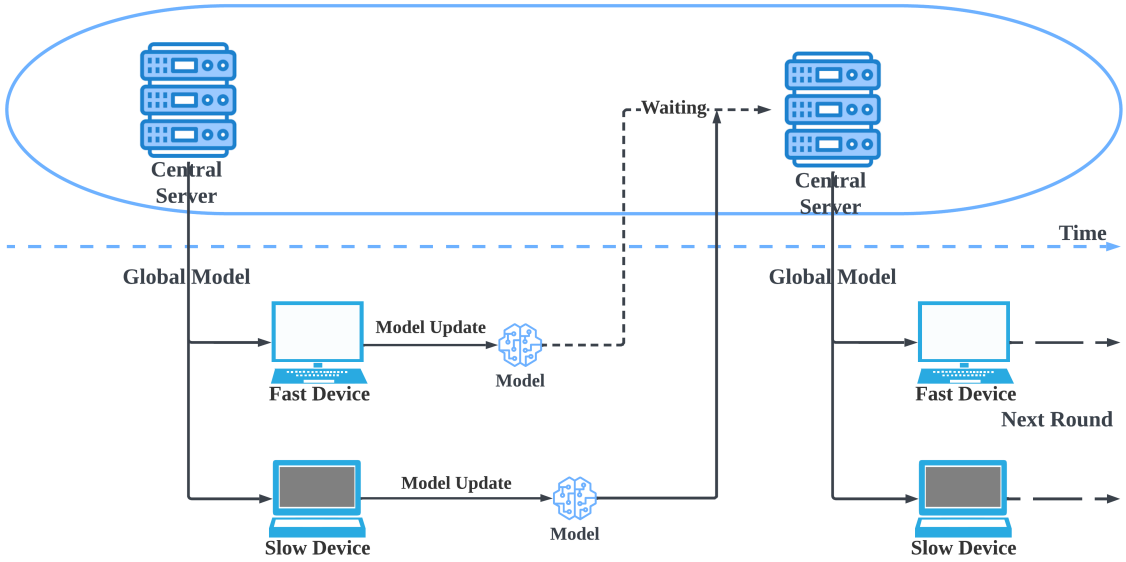

Figure 5.2: Traditional Federated Learning

other hand, occur due to the latency inherent in transmitting updates, especially in a decentralized system like blockchain.

In addressing these challenges, we introduce an IoT-compatible federated learning system underpinned by blockchain technology. Traditional methods, which merge federated learning with blockchain, often rely solely on a singular type of blockchain. When deploying federated learning within the Internet of Everything (IoE) framework, it becomes imperative to amalgamate both public and private chains. This dual-chain setup not only combats computational and communication challenges but also facilitates efficient device ID management. While the public chain serves as a universal repository for model uploads and downloads, the private chains cater to specific entities or companies desiring a segregated space for model sharing.

5.3.2 System Model

To address the computing challenges described earlier, we have developed a new architecture for a crowdsourcing system, which involves three parties in our proposed framework.

- Agent: The agent is a crucial component in our proposed system. First, the agent needs to register in the blockchain network and undergo identity verification. The agent can then upload the global model $Model_g$, which is used for subsequent fed-

erated learning training processes. After waiting for the entire system to complete their federated learning training, the agent can download the trained model and publish the new model for training in the blockchain network.

- Client: The client, which also represents IoT devices in our system, is essential in future IoE scenarios as numerous IoT devices will join the blockchain networks. Like the agent, each client needs to register in the blockchain first. Additionally, during the registration process, they must specify the organization or company they belong to. Clients can download the global model and train the model locally. After completing the training process, they will publish the dataset $< B_{ID}, Weights, Gradient, Model, ID >$ on the blockchain network to help update the global model.

- Company: There are two types of companies: those containing a few clients and those containing many clients. Companies with only a few clients do not need to establish a private chain. They can naturally run the federated learning training process according to the settings in the smart contract. For companies with many clients, as mentioned earlier, they may bring significant computing challenges that slow the federated learning training process. In our designed system, companies with many clients can establish their private chain for a more efficient federated learning training process.

The agent will register and upload the global model to the public chain first. The federated learning epoch must be set in the smart contract. A blockchain network will be initialized first. After the entire blockchain network is initialized, the federated learning training process will start. Different types of companies will download the global model to train the model with their dataset. Companies with only a few clients will naturally run the training process. However, for companies with many clients, the blockchain network will establish a private chain for them. The federated learning training process epoch setting on the private chain is determined by the epoch setting on the public chain. After completing the private chain's epoch setting, the company with many devices will upload their private model $Model_p$ to the public blockchain to finish the model updating process. A more detailed description of this entire process will be provided in Section 5.4.

5.3.3 Adversary Model

In our designed system, we take into account two potential threat categories:

- Insider Threats: Within our proposed system, clients are deemed semi-trusted during the training process. Given this assumption, clients may be honest but curious about parameter updates and could potentially deduce sensitive information from blockchain transactions. Specifically, although the original data is not directly shared, inquisitive clients can still extract training data from gradients and approximate the raw data, particularly when the architecture and parameters are not completely secured. Moreover, malicious clients can exploit and learn data structures, such as image pixels derived from global model updates, without the consent of other clients and MEC servers. However, in our blockchain network, each time a client accesses data, the action is recorded in the blockchain network, facilitating the detection of data leaks.

- External Threats: Clients may confront adversaries who try to introduce backdoors into the model by manipulating data features or incorporating an incorrect subset of data into the original dataset, aiming to alter local clients' training objectives. Attackers may also compromise client devices and tamper with local model parameters during the learning process, leading to errors in the global model update. Furthermore, adversaries can exploit wireless communication channels during FL training to access clients' personal information. For example, sensitive user information, such as age and preferences, might be extracted from parameter update packages. External eavesdroppers can also gain unauthorized access to clients, potentially taking control of the model update aggregation process. However, our proposed system utilizes a permissioned blockchain, which requires clients to pass an identity check before participating in the network. This significantly raises the cost of external threats and helps prevent attacks.

5.4 Proposed system

5.4.1 Overview

In our proposed federated learning system with public and private blockchain, we address the massive computing challenges in future IoE scenarios. This approach can tackle traditional federated learning problems as mentioned in the problem definition section. The smart contract of the blockchain network automates model updating and aggregation while preserving the privacy of participants since all clients on the blockchain network are anonymous. Figure 5.3 depicts the procedure of our proposed federated learning

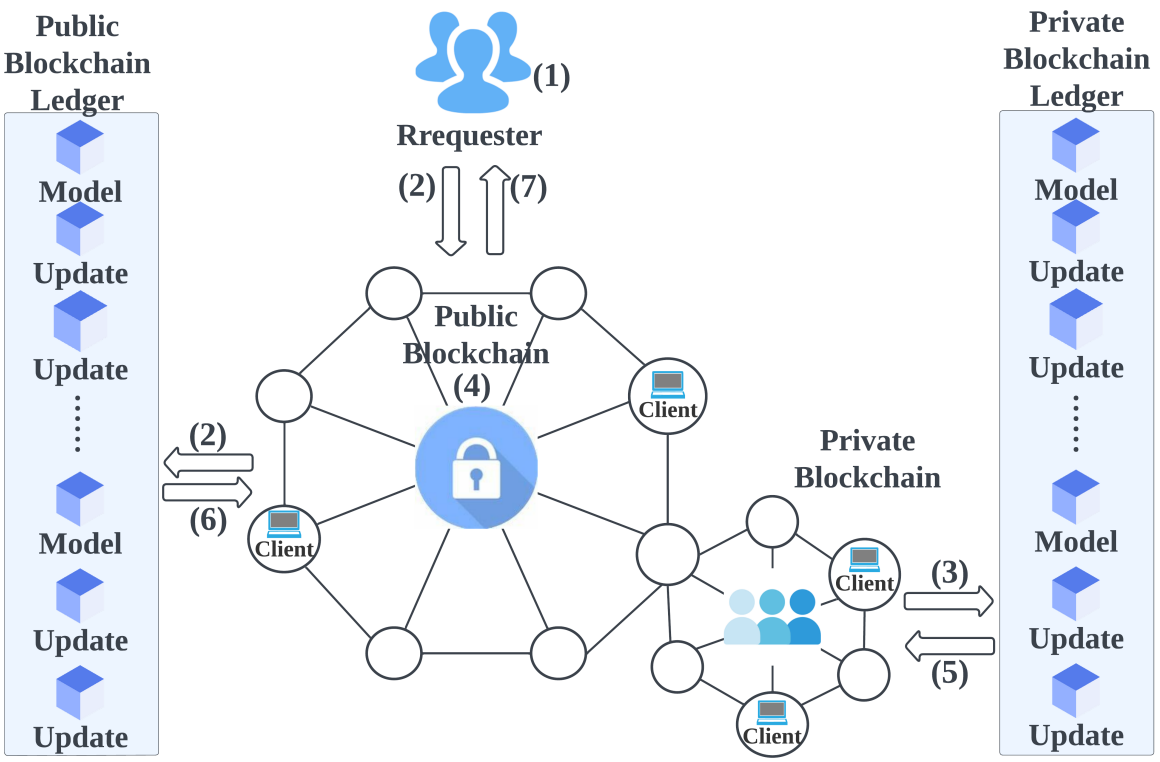

Figure 5.3: Process of proposed system. (1) Register. (2) Global model upload. (3) Upload the model to the private chain. (4) Model updating setup. (5) Model aggregation in the private chain and send it to the public chain. (6) Model aggregation in the public chain. (7) Global model update and the requester request the model.

system with public and private blockchain. More details of the implementation will show in Section 5.4.2.

Initially, the agent, client, and company register on the public blockchain. Companies register as separate organizations within our proposed system, with each organization consisting of multiple clients. When registering on the blockchain network, clients must specify their affiliated organization. In contrast, agents' registration does not involve indicating an organization, as they are responsible for uploading, rather than updating, the global model. Subsequently, the agent uploads the global model to the public chain. Once the global model is uploaded, companies establish their private chains to improve the federated learning training process. The federated learning training epoch is set within the smart contract. When the private chain's training process reaches the specified epoch in the smart contract, model aggregation commences in the private chain. The private model, $Model_p$, is then sent to the public chain for further aggregation. Once

the smart contract detects the completion of the model aggregation process, it updates the final global model, $Model_f$, which the agent can then obtain. The following sections provide a detailed explanation of our proposed method.

- Register: All agents, clients, and companies must register to participate in our proposed system. Upon registration, the company will receive an organization name in the blockchain network. Each client and agent will receive a pair of keys, and their identification is recorded in the user pool, which includes their username and the organization or company they are affiliated with.
- Global model upload: Once the agent successfully registers in the public blockchain, they can publish the global model $Model_g$ to the public chain. However, the agent's identity must be verified before uploading the global model.
- Private model upload: When the smart contract recognizes that the global model has been successfully uploaded and there are some organizations with numerous clients that may impede the federated learning training process, a private chain will be established. The agent in that company can upload the Global model to the private chain to address the computing problem.
- Model updating setup: To ensure the federated learning training process runs successfully on both the public and private chains, the epoch of the federated learning training process on both chains must be set. The epoch setting on the public chain should be a multiple of the epoch setting on the private chain. This is done to ensure that when the model aggregation is finished on the private chain, it can be sent to the public chain for model aggregation immediately. The model aggregation on the public chain will not need to wait for the private chain.
- Model aggregation in private chain and public chain: When the smart contract realizes the federated learning training process on the private chain has reached the set epoch, it will instruct the agent in the organization to send the private model $Model_p$ to the public chain and wait for further model aggregation on the public chain. At this time, the agent's identity check will also be required. Moreover, when the federated learning training process on the public chain also reaches the epoch setting, the whole model aggregation process will start on the public chain.
- Global model update: After the entire training process is finished on the public chain, the smart contract will update the final global model according to the model on the public chain and wait for the next model upload.

5.4.2 Implementation of Our Designed System

5.4.2.1 Register

In our proposed blockchain-based registration process, users are not obligated to reveal their true identities. Instead, they supply a pseudonym and their affiliated organization's name during registration. Upon successful registration, the organization's Certificate Authority (CA) issues a pair of cryptographic keys to the user. Algorithm 10 outlines the registration process, with the *RegisterSuccess* indicator signifying a successful user registration.

Algorithm 10 Client Register

Require: U_{name}, *Org*
Ensure: *RegisterSucess, jwt*
1: *RegisterSucess = False*;
2: *Check Org*;
3: **if** $U_{name} \in U_{pool}$ **then**
4: **return** U_{name} already existed.
5: **end if**
6: P_k,$S_k \leftarrow$ *keyGenerator()*;
7: $jwt \leftarrow P_k$, S_k;
8: $U_{name} \leftarrow jwt$;
9: $Pool_u \leftarrow U_{pool} \cup ID_{ui}$;
10: *RegisterSuccess = True*;
11: **return** *RegisterSucess, jwt*

During registration, the blockchain network verifies if the user's pseudonym, $Uname$, is already included in the $Upool$, and the associated organization is checked in step 2. If $Uname$ exists in the $Upool$, the user is prompted to choose a different pseudonym, and the registration process is considered unsuccessful (Steps 3-4). A pair of keys is generated by the *keyGenerator()* and provided to the user through a *JSON Web Token (JWT)* following X509 certification (Steps 6-8). If the user ID is not in the user ID pool, $Pool_u$, it is added in step 10, and the *RegisterSuccess* indicator is set to true. The process concludes by returning the *RegisterSuccess* status and a *JWT*, which is used in later steps. Notably, the *JWT* is generated based on the user ID and corresponding X509 certification results.

5.4.2.2 Global Model Upload

Algorithm 11 Global Model Upload

Require: *jwt token*, $Model_g$
Ensure: *UploadPublic, Model*
1: *UploadPublic* = False;
2: **if** *jwt token ineligibility* **then**
3: **return** *jwt token* expired
4: **end if**
5: $Model \leftarrow Model_g$;
6: *UploadPublic* = *True*;
7: **return** *UploadPublic, Model*

After successful registration, users can continue to upload the global model to the public chain. As depicted in Algorithm 11, the *UploadPublic* indicator represents successful model uploads to the public chain. When a user tries to upload the global model, the blockchain network evaluates the *JWT* for validity. If found invalid, an expiration message is returned (Steps 2-4). In step 5, the global model $Model_g$ is uploaded to *Model* on the public chain. Subsequently, *UploadPublic* is set to true, and both *UploadPublic* and *Model* are returned (Steps 6-7).

5.4.2.3 Private Chain Establish

Following the successful upload of the global model to the public chain, it is essential to upload the model to the private chain as well, incorporating an authorization process.

Algorithm 12 Upload to the Private Chain

Require: *jwt token*, $Model_g$
Ensure: *UploadPrivate,* $Model_p$
1: *UploadPrivate* = False;
2: **if** *jwt token ineligibility* **then**
3: **return** *jwt token* expired
4: **end if**
5: $Model_p \leftarrow Model_g$;
6: *UploadPrivate* = *True*;
7: **return** *UploadPrivate, Model*

In Algorithm 12, upon successful upload of the global model to the public chain, the model must be uploaded to the private chain. The *UploadPrivate* indicator signifies the successful upload of $Model_g$ to the private chain. Similar to Algorithm 11, user authentication is required when uploading $Model_g$ to the private chain (Steps 2-4). In

Step 5, the global model $Model_g$ is uploaded to $Model_p$. Subsequently, *UploadPrivate* is set to true and returned with the model.

5.4.2.4 Model Updating Setup

Once the global model has been successfully uploaded to both the public and private chains, the model updating setup must be finalized, as depicted in Algorithm 13. The system sets $Public_{epoch}$ and $Private_{epoch}$ according to the specified *epoch* in Step 2. Importantly, the $Public_{epoch}$ must be a multiple of $Private_{epoch}$ to ensure simultaneous model aggregation on both the public and private chains. If this condition is met, the *UpdatingSetup* indicator is set to true (Steps 3-5), and both *UpdatingSetup* and *epoch* are returned.

Algorithm 13 Model Updating Setup

Require: *epoch*
Ensure: *UpdatingSetup,* $Public_{epoch}$, $Private_{epoch}$
1: *UpdatingSetup* = False;
2: Set $Private_{epoch}$ and $Public_{epoch}$
3: **if** $Public_{epoch}$ is multiple of $Private_{epoch}$ **then**
4: *UpdatingSetup = True*
5: **end if**
6: **return** *UpdatingSetup, epoch*

5.4.2.5 Model Aggregation in Private Chain

With the epoch successfully configured, federated learning proceeds on the private chain according to the set epoch. Upon reaching the specified $Private_{epoch}$, the smart contract (SC) initiates the model aggregation process on the private chain, detailed in Algorithm 14.

In Algorithm 14, the *UpdatingSetup* indicator is required initially. As federated learning progresses on the private chain, and private node P_n reaches the $Private_{epoch}$ setting, the SC aggregates the model on the private chain $Model_p$ (Steps 2-5). Additionally, the *JWT token* is verified when P_n uploads $Model_p$ (Steps 6-8). The SC then sends the model to the public chain for model aggregation (Step 10). The private chain aggregation indicator, $Aggregation_p$, is set to true, and both $Aggregation_p$ and the model are returned.

Algorithm 14 Model Aggregation in Private Chain

Require: *jwt token*, $Private_{epoch}$, $Model_p$, $Aggregation_p$
Ensure: *UpdatingSetup*
1: $Aggregation_p$ = False;
2: SC realise node reach $Private_{epoch}$;
3: **if** P_n reach $Private_{epoch}$ **then**
4: SC aggregate $Model_p$;
5: $Model \leftarrow Model_p$;
6: **if** *jwt token ineligibility* **then**
7: **return** *jwt token* expired
8: **end if**
9: **end if**
10: SC send *Model* to *PublicChain*.
11: $Aggregation_p$ = True;
12: **return** $Aggregation_p$, *Model*

5.4.2.6 Model Aggregation in Public Chain

In parallel to the private chain model aggregation process, a similar process occurs on the public chain, as outlined in Algorithm 15. At the onset of Algorithm 15, the public chain aggregation indicator ($Aggregation_g$) is set to false. When the global node G_n reaches the previously set *epoch*, the SC commences model aggregation (Steps 2-4). A *JWT token* must be validated to upload the model (Steps 5-7). Ultimately, the indicator $Aggregation_g$ is set to true and returned alongside $Model_g$.

Algorithm 15 Model Aggregation in Public Chain

Require: *Model, jwt token,* $Aggregation_g$
Ensure: *epoch*, $Aggregation_p$
1: $Aggregation_g$ = False;
2: **if** G_n reach *epoch* **then**
3: SC aggregate *Model*.
4: $Model_g \leftarrow Model$;
5: **if** *jwt token ineligibility* **then**
6: **return** *jwt token* expired
7: **end if**
8: **end if**
9: $Aggregation_g$ = True;
10: **return** $Aggregation_g$, $Model_g$

5.4.2.7 Global Model Update

Following model aggregation on the public chain, a global model update process is initiated, wherein the $Model_f$ is updated using $Model_p$ and $Model_g$. Algorithm 16 outlines the primary process for the global model update. Initially, the *UpdateSuccess* indicator is set to *False*. When the smart contract (SC) identifies that both $Aggregation_p$ and $Aggregation_g$ indicators are *True*, it signifies the completion of model aggregation on both private and public chains. Consequently, the SC updates the final global model $Model_f$ based on $Model_p$ and $Model_g$ (Steps 2-4). The *UpdateSuccess* indicator is then set to *True*, indicating the conclusion of the global model update process. The final global model $Model_f$ and the *UpdateSuccess* indicator are returned.

Algorithm 16 Global Model Update

Require: *SC,* $Model_f$
Ensure: $Aggregation_g$, $Aggregation_p$, *Model, UpdateSuccess*
1: *UpdateSuccess* = False;
2: **if** $Aggregation_g$ **and** $Aggregation_p$ == True **then**
3: SC update the final global model $Model_f$.
4: $Model_f \leftarrow Model_p$,$Model_g$;
5: *UpdateSuccess* = True;
6: **end if**
7: **return** *UpdateSuccess*, $Model_f$

5.5 Privacy and Security Analysis

5.5.1 Privacy Analysis

A paramount concern in federated learning revolves around the protection of client data during the model's training phase [80]. In the conventional paradigm of federated learning, data is accumulated and amalgamated on a singular centralized server, inadvertently elevating the potential for privacy infringements [88]. The synergistic employment of both public and private chains can serve as a potent antidote to these privacy threats.

Public chains, celebrated for their transparency and incorruptibility, emerge as the ideal candidates for documenting and authenticating transactions within a decentralized milieu. Yet, their application in federated learning is not devoid of challenges. Inherent in the very design of public chains is the fact that all data is open to inspection by every participant in the network. This includes all data accumulated during the model training

process. The unintended consequence is that any inadvertent revelation of sensitive information might snowball into severe privacy transgressions, with ensuing legal and ethical repercussions.

Private chains, juxtaposed against public chains, introduce a heightened echelon of data privacy and dominion. Their architecture is such that they restrict network entry solely to accredited entities, creating a fortress around sensitive data and permitting access only to those who bear the requisite authorization. When inducted into federated learning, private chains play a pivotal role in curtailing the risk of data exposure and bolstering the sanctity of client data. A unique facet of private chains is their adaptability. They can be meticulously molded to resonate with explicit privacy stipulations, encompassing techniques like advanced encryption and stringent access controls to fortify data.

However, the incorporation of private chains is not without its hurdles. Navigating coordination intricacies among disparate participants becomes imperative to vouchsafe the precision and reliability of the model. A promising resolution to this conundrum lies in the embrace of a hybrid blockchain framework, adeptly weaving together the strengths of both public and private chains. This amalgamated architecture, as elucidated in our proposed methodology, adeptly surmounts these challenges.

5.5.2 Security Analysis

Besides privacy concerns, federated learning also presents security challenges related to data integrity and availability [87]. The centralized architecture of traditional federated learning, in which client data is gathered and aggregated on a centralized server, is susceptible to external attacks. Employing public and private chains in tandem can bolster the security of the federated learning process.

Although public chains offer transparency and decentralization, they may compromise security. Public chains are susceptible to attacks such as 51% attacks, where a single entity gains control of over 50% of the network's computational power, allowing them to manipulate chain transactions. Additionally, the deployment of smart contracts in public chains could expose the network to vulnerabilities like reentrancy attacks and denial-of-service attacks. These security risks can threaten the integrity and availability of data stored on the chain, which can have disastrous consequences in the context of federated learning.

On the other hand, private chains deliver improved security and network control. Private chains limit access to authorized participants, reducing the likelihood of external

attacks. Furthermore, private chains can be customized to satisfy particular security requirements, including implementing access controls and encryption to protect sensitive data.

By leveraging public and private chains simultaneously, we can exploit the benefits of both strategies while mitigating their respective risks. Specifically, we can use private chains to maintain clients' data privacy and enhance the security of the federated learning process. In addition, public chains can guarantee transaction transparency and immutability, as well as provide a decentralized network that is resilient to single-point-of-failure attacks.

It is crucial to acknowledge that incorporating public and private chains in federated learning introduces challenges associated with network management, such as interoperability between different chains and coordination of participants' efforts. These obstacles necessitate the development of novel frameworks and protocols to guarantee the secure and efficient functioning of the network.

5.6 Results and Analysis

In this section, we analyze our system from two aspects: federated learning performance and blockchain platform performance. The experiment settings are detailed in the corresponding sections below.

5.6.1 Federated Learning Performance

We assessed the performance of our proposed system using two datasets: MNIST and CIFAR10. The results of the train loss are as follows:

- MNIST: We undertook a comprehensive comparative analysis, evaluating our system in relation to the original federated learning approach and other federated learning systems integrated with a public blockchain network, utilizing the MNIST dataset.

  In particular, we conducted a meticulous investigation, examining two distinct epoch settings: 10 and 50 epochs. The training losses of the traditional federated learning approach, the other federated learning systems incorporating a public blockchain, and our proposed system are depicted in the subsequent figures.

  Both Figure 5.4 and Figure 5.5 present the training progress for an epoch setting of 10, considering variations in the number of users (50 and 100, respectively).

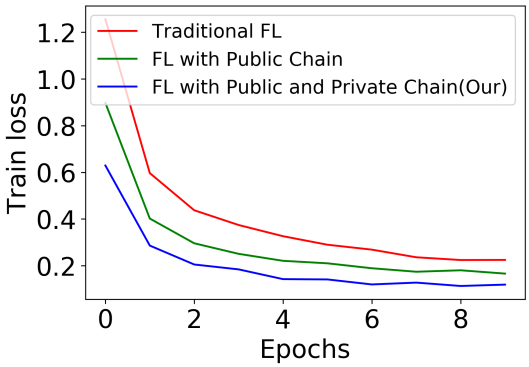

Figure 5.4: Setting with 10 epochs and 50 users under MNIST dataset

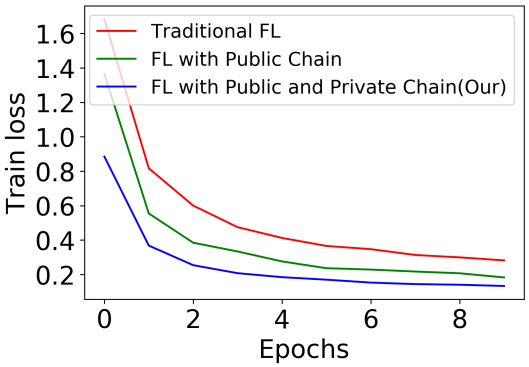

Figure 5.5: Setting with 10 epochs and 100 users under MNIST dataset

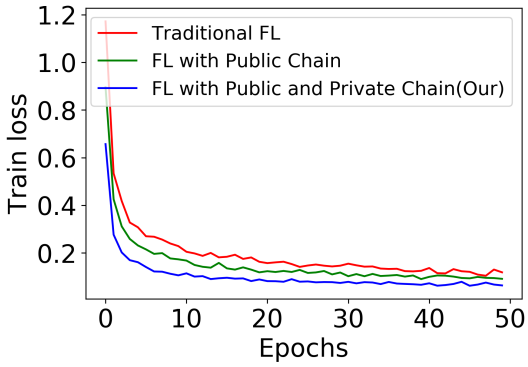

Figure 5.6: Setting with 50 epochs and 50 users under MNIST dataset

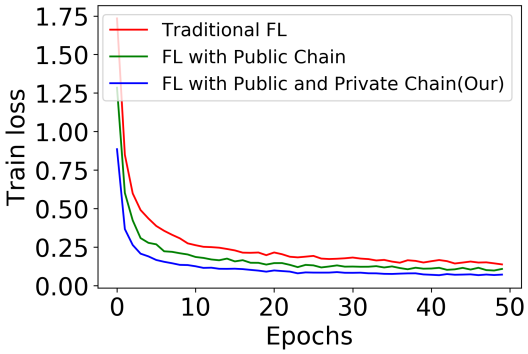

Figure 5.7: Setting with 50 epochs and 100 users under MNIST dataset

Additionally, Figure 5.4 focuses on the accuracy attained by the traditional federated learning approach, which reaches 96.94 after 10 epochs. Conversely, the integration of other federated learning systems with a public blockchain network achieves a higher accuracy of 97.89. Significantly, our proposed system outperforms both, attaining an accuracy of 98.39.

Furthermore, in Figure 5.5, the accuracy of the traditional federated learning system is 96.92, which is lower than that of the other federated learning systems combined with a public blockchain network (97.14). However, our proposed system achieves an accuracy of 97.97, surpassing both of these performances. Notably, our proposed system consistently demonstrates lower training loss than the traditional federated learning system and other federated learning systems combined with a public blockchain network at each epoch.

In Figure 5.6 and Figure 5.7, we present results that are consistent with the previous findings. We conducted experiments using an epoch setting of 50 to compare our system with other federated learning systems that incorporate a public blockchain, as well as the traditional federated learning approach.

When considering 50 epochs and 50 users, the accuracy of the traditional federated learning system is observed to be 98.14, while the accuracy of the other proposed system is 98.77. However, our proposed system achieves a higher accuracy of 99.04.

Under the setting of 100 users, our proposed system demonstrates an accuracy of 98.75, surpassing both the accuracy of the traditional system (98.12) and the other proposed system (98.36).

In summary, our proposed system demonstrates superior performance compared to other federated learning systems that solely integrate a public blockchain network, as well as the traditional federated learning approach, in MNIST dataset. Our system consistently achieves higher accuracy and lower training loss across various epochs and user settings. These findings underscore the effectiveness of our proposed system in enhancing the performance of federated learning tasks.

- CIFAR10: For the CIFAR10 dataset, we employed a similar experimental setup as with the MNIST dataset, utilizing two distinct epochs (20 and 50 epochs) and varying the number of users (50 and 100) for comparison.

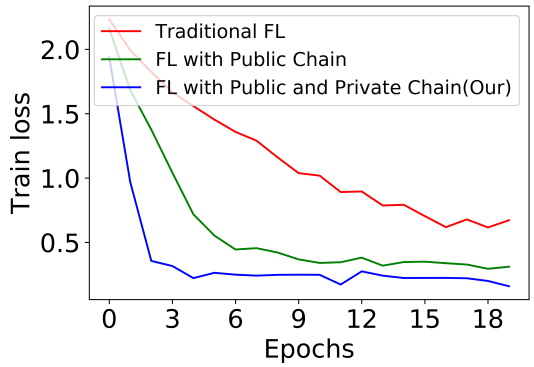

Figure 5.8: Setting with 20 epochs and 50 users under CIFAR-10 dataset

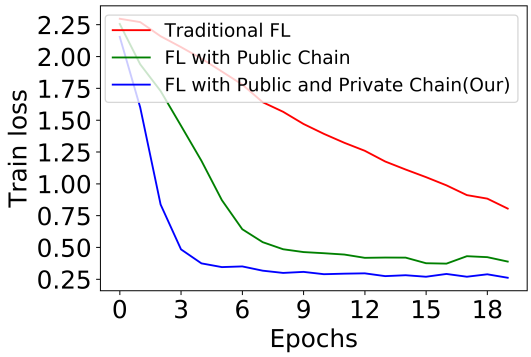

Figure 5.9: Setting with 20 epochs and 100 users under CIFAR-10 dataset

In the case of 20 epochs and 50 users, the traditional federated learning system achieved an accuracy of 51.47, while the other proposed system reached 51.19. However, our proposed system demonstrated superior performance with an accuracy

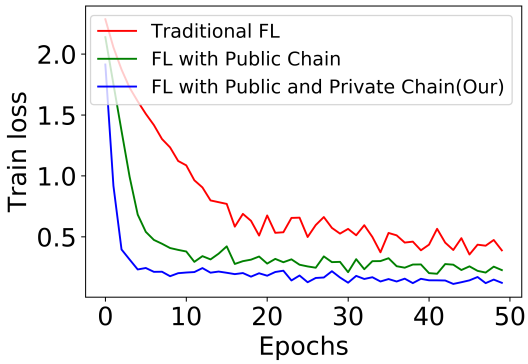

Figure 5.10: Setting with 50 epochs and 50 users under CIFAR-10 dataset

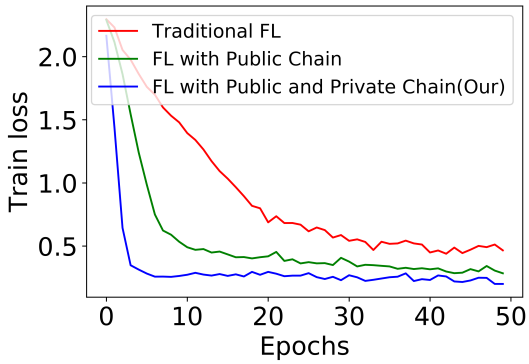

Figure 5.11: Setting with 50 epochs and 100 users under CIFAR-10 dataset

of 55.02. Figure 5.8 illustrates a faster convergence rate for our proposed system, as evidenced by significantly lower training loss after only 2 epochs. Similarly, Figure 5.9 presents a similar trend, where our proposed system demonstrates a faster convergence rate compared to the other two methods. Specifically, our system achieves an accuracy of 49.11, which is higher than the accuracy of the other federated learning system that incorporates a public blockchain (48.42), as well as the accuracy achieved by the traditional federated learning system (48.97).

Similarly to the findings in Figure 5.8 and Figure 5.9, our proposed system demonstrates a faster convergence rate in Figure 5.10 and Figure 5.11. In both cases, our proposed system achieves a low training loss after 5 epochs.

Under the setting of 50 epochs and 50 users, our proposed system achieves a higher accuracy of 53.51, surpassing the accuracy of the traditional federated learning system (46.24) and the other proposed federated learning system that incorporates a public blockchain (51.37).

Moreover, with 50 epochs and 100 users, our proposed system also attains a higher accuracy of 54.51 compared to the accuracy of the traditional federated learning system (51.17) and the other proposed system (52.19).

In summary, for the CIFAR-10 dataset, our proposed method exhibits higher accuracy compared to the traditional federated learning system and the other proposed system that integrates a public blockchain. Additionally, our proposed method demonstrates a faster convergence rate. This advantage becomes more pronounced as the number of epochs increases.

5.6.2 Blockchain Platform Performance

In the blockchain platform performance analysis, we assessed our system using three key metrics: latency, throughput, and send rate. These metrics are crucial in determining the performance of a blockchain platform. We compared our proposed system, which incorporates federated learning, with the original blockchain platform that does not, using varying numbers of individual blocks (50, 100, 200, 400, 500, and 800).

- Latency: As illustrated in Figure 5.12, the mean latency for both the traditional blockchain network and our proposed method rose as the block size increased. Up to 500 blocks, the latency performance of our proposed method closely paralleled that of the conventional blockchain network. Beyond 500 blocks, our proposed method demonstrated a more rapid latency growth rate, mainly due to the rising number of model updates transmitted through the blockchain network as the block size grew. Although the latency of our proposed method reached 18 seconds at 800 blocks in contrast to 15 seconds for the traditional blockchain network, it remains within tolerable limits. To better showcase the differences, we generated a box plot encompassing the average, maximum, and minimum latencies.

  The box plot in Figure 5.13 validates our observations from the line graph. Our proposed method, signified by green boxes, displayed a higher median latency compared to the traditional blockchain network, indicated by blue boxes. Notably, our proposed method showed superior performance at 500 blocks, achieving a lower maximum latency than the traditional blockchain system. In conclusion, our proposed method exhibited a slightly higher latency performance compared to the traditional blockchain network.

- Throughput: Figure 5.14 reveals a rapid decline in throughput for both systems from 100 to 800 blocks. Notably, our proposed system initially outperformed the original system. However, due to the model update and aggregation processes in federated learning, the transmission of additional data may have caused a delay in throughput. We constructed a box plot to provide more detail, similar to our latency analysis.

  The box plot in Figure 5.15 mirrors the findings of the line graph, with our proposed system achieving a higher median throughput at 100 blocks than the original blockchain network. However, for other block size settings, the original blockchain network demonstrated superior performance.

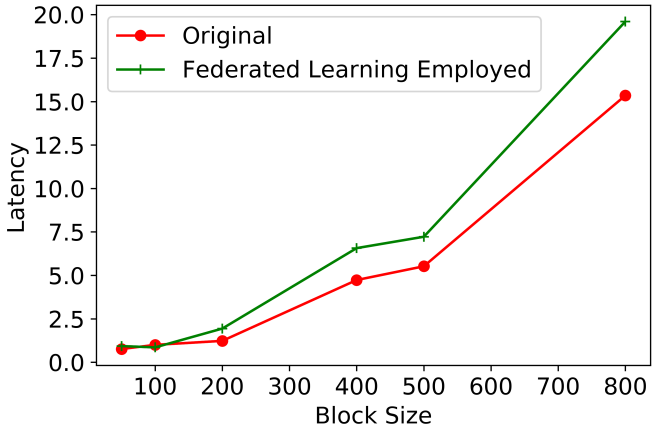

Figure 5.12: The latency comparison of our system with the original blockchain network

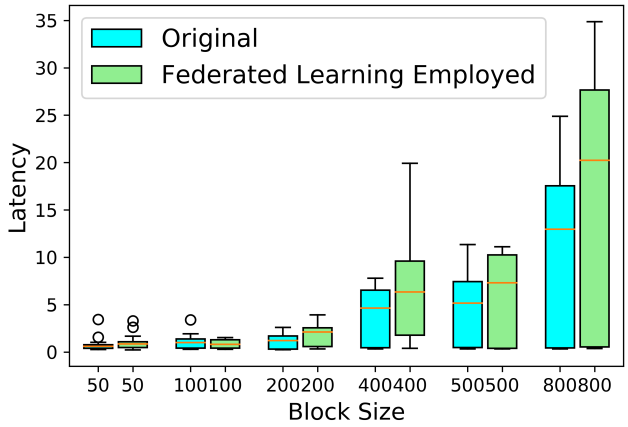

Figure 5.13: More details of latency comparison

- Send Rate: Both systems were configured with a send rate of 40 TPS, and the system with a send rate closer to this benchmark is considered better performing. As the send rate is critical to the federated learning training process, a low send rate could substantially affect the training speed. In Figure 5.16, our proposed system is closer to the 40 TPS send rate, particularly for 100, 200, and 500 blocks. We created a box plot in Figure 5.17 to provide more insight.

  The box plot in Figure 5.17 indicates that our proposed system often exhibits a considerably higher median send rate than the original system. Across all block settings, the lowest send rate for our system is 37 TPS, compared to 36 TPS for the original blockchain network. Based on figures 5.16 and 5.17, we can conclude that

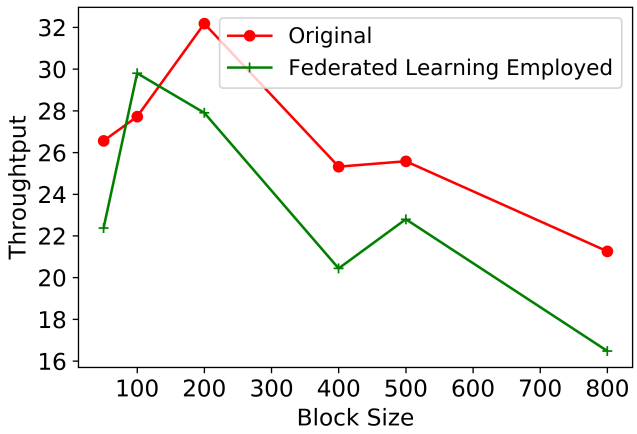

Figure 5.14: The throughput comparison of our system with the original blockchain network

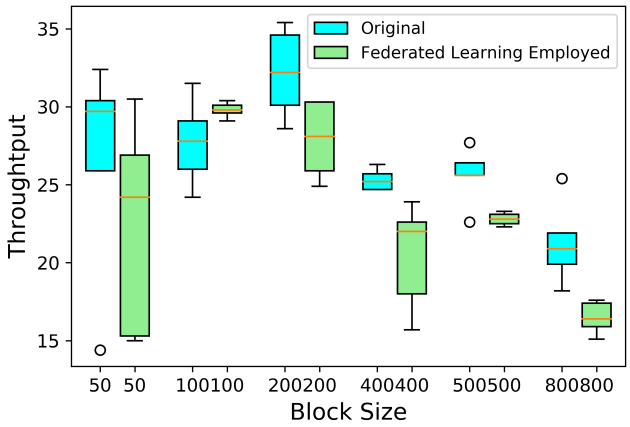

Figure 5.15: More details of throughput comparison

our proposed system outperforms the original blockchain network in terms of send rate.

5.6.3 Analysis Conclusion

In conclusion, our proposed system outperforms both the traditional federated learning approach and other federated learning systems that incorporate a public blockchain, across both the MNIST and CIFAR-10 datasets. Our system consistently demonstrates superior performance, exhibiting lower training loss, higher accuracy, and a faster convergence rate. These results highlight the efficacy and potential of our proposed

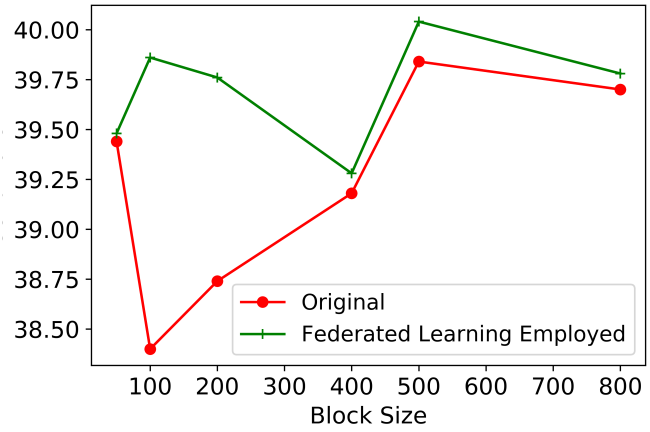

Figure 5.16: The send rate comparison of our proposed method with the original blockchain network

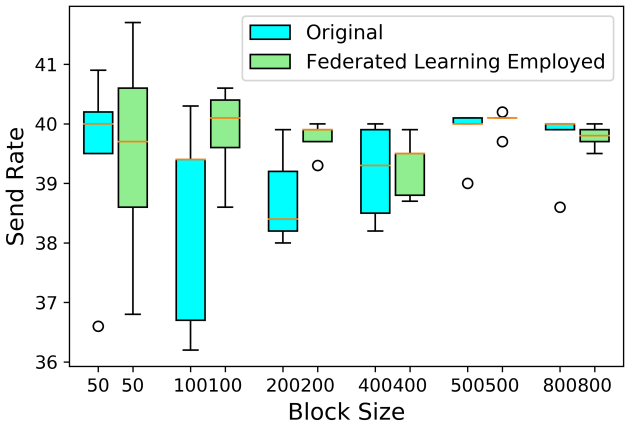

Figure 5.17: More details of send rate comparison

system in improving the performance of federated learning tasks in various datasets.As for blockchain platform analysis, our proposed system show a bit higher latency and throughput than the original blockchain network, but we get a higher send rate which is useful when deploying the federated learning in blockchain networks. As the block size increases, both the original system and our proposed system are likely to exhibit higher latency and lower throughput. The results show our proposed system is suitable for future IoE networks when employ the federated learning in blockchain networks.

5.7 Summary

In conclusion, this chapter introduces a novel approach to address the challenges of federated learning by integrating public and private chains. Through extensive experiments conducted on the MNIST and CIFAR-10 datasets, we have demonstrated that our proposed method achieves higher accuracy, faster convergence rates, and reduced training loss compared to both traditional federated learning approaches and other proposed federated learning systems that integrate blockchain networks. Furthermore, in comparison to the original blockchain network, our method exhibits slightly increased latency and throughput but a superior send rate, which is crucial for effectively integrating federated learning and blockchain. By adopting our approach, the computational burden on client devices is alleviated while ensuring the preservation of data privacy and security.

CHAPTER 6

BLOCKCHAIN-BASED GRADIENT INVERSION AND POISONING DEFENSE FOR FEDERATED LEARNING

6.1 Introduction

The rapid development of the Internet of Things (IoT) has ushered in a new era of interconnected devices and data generation, with countless applications spanning various domains, including healthcare, transportation, smart homes, and agriculture [78]. The exponential growth in data has necessitated the development of advanced machine learning techniques to analyze and derive actionable insights from this data. As traditional centralized machine learning models face challenges in terms of data privacy, communication overhead, and scalability, Federated Learning (FL) has emerged as a promising decentralized approach to address these issues [73].

Federated Learning enables the training of machine learning models using data from various devices while preserving user privacy by keeping the data on local devices and sharing only the model updates [59]. This approach has become increasingly popular in IoT scenarios, where numerous connected devices generate and process data at the edge. FL not only reduces the communication overhead associated with transmitting large volumes of data to a central server but also fosters collaborative learning among edge devices while ensuring that sensitive information remains secure [46].

However, the distributed nature of FL exposes it to various types of attacks that can compromise the learning process, model performance, and overall system security [87].

Among these, Gradient Inversion Attacks (GIA) and Poison Attacks (PA) are particularly detrimental. GIAs involve adversaries that manipulate the gradient updates in a way that sabotages the learning process, ultimately compromising the performance of the model [31]. These attacks are difficult to detect, as the adversaries can subtly alter the gradient updates, leading to a gradual degradation of model performance over time.

PAs, on the other hand, are orchestrated by adversaries injecting malicious data samples into the training process [74]. These data samples are carefully crafted to manipulate the model's behavior, causing it to produce incorrect or malicious outputs when presented with specific inputs [80]. The malicious data samples often appear to be legitimate, making it challenging to identify and prevent such attacks. The consequences of PAs can be severe, leading to a decline in the model's performance and integrity, and potentially causing significant harm if deployed in critical applications.

Given the importance of data security and privacy in IoT and FL, developing effective defense mechanisms against these types of attacks is crucial. Blockchain technology, with its inherent decentralized and tamper-resistant nature, has emerged as a promising solution to enhance the security of FL in IoT scenarios [3]. By leveraging the cryptographic features, consensus algorithms, and smart contracts provided by blockchain technology, we can ensure the integrity and authenticity of the gradient updates and data samples used in the FL process.

In response to these threats, we propose a novel framework that combines blockchain technology and differential privacy to defend against Gradient Inversion Attacks and Poison Attacks in Federated Learning. To protect against GIAs, our framework employs a two-pronged approach. First, it uses a public blockchain to secure the gradient information uploaded by clients through smart contracts and applies Differential Privacy Stochastic Gradient Descent (DPSGD) for enhanced protection. Second, it incorporates a private blockchain, where clients directly integrate DPSGD during the Federated Learning process to protect the gradients. To defend against PAs, our framework takes advantage of the traceability offered by blockchain technology to identify and exclude attackers from the learning process.

This chapter presents the design and implementation of the proposed defense framework, detailing its components and underlying mechanisms for defending against GIA and PA. The framework is designed to ensure the security and privacy of the data used in the FL process, improve the overall reliability and performance of the trained models, and provide robust protection against various attack scenarios.

Our contributions are in the followings:

- We develop a blockchain-based defense against gradient inversion and poisoning attacks, directly addressing critical security and privacy concerns in federated learning IoT applications. This innovative solution mitigates threats to global model integrity and effectiveness, warranting consideration for wide-scale adoption.
- Our approach uniquely combines public and private blockchain levels, ensuring robust gradient protection while maintaining model performance and utility. This balanced and comprehensive method offers a significant advantage over existing techniques, as it addresses a broader range of attack vectors without sacrificing model quality.
- Extensive experiments validate our method's effectiveness, revealing improved accuracy and stable training loss convergence in attack scenarios. These findings showcase the potential of our approach for enhancing security and privacy across various IoT applications, substantiating its relevance and value to the federated learning research community.

6.2 Preliminary

6.2.1 Gradient Inversion Attack in Federated Learning

Gradient Inversion Attack is a type of adversarial attack targeting the Federated Learning process, in which an attacker manipulates the gradient updates submitted by their device to intentionally invert the learning process [30]. The goal of a GIA is to degrade the global model's performance and compromise the overall system's security.

During the Federated Learning process, the clients compute the gradient updates (∇L) based on their local data and transmit these updates to the central server. In a GIA, the attacker modifies the gradient updates, leading to an inverted learning process. The attacker can perform this manipulation as follows:

$$\nabla L_{\mathrm{A}} = -\alpha \nabla L + \beta \nabla A \tag{6.1}$$

where ∇L_{A} represents the malicious gradient update submitted by the attacker, α and β are the attack parameters controlling the strength of the inversion, and ∇A is the additional adversarial gradient crafted by the attacker.

Defending against GIA requires the implementation of robust mechanisms to ensure the authenticity and integrity of the gradient updates shared among participating devices.

6.2.2 Poison Attack in Federated Learning

Poison Attack is another adversarial attack targeting Federated Learning systems [74], in which an attacker injects carefully crafted malicious data samples (x^*, y^*) into the training process. These poisoned samples can negatively impact the global model's performance and cause the model to produce incorrect predictions.

The attacker aims to maximize the loss function $L(x^*, y^*, w)$ with respect to the model parameters w, in order to influence the training process:

$$L(x^*, y^*, w) = L(x, y, w) + \lambda(x^*, y^*) \tag{6.2}$$

where λ is the attack parameter controlling the strength of the poisoning effect, and (x, y) represents the original training data.

Defending against PA necessitates the development of methods to detect and exclude attackers who contribute malicious data samples and to maintain the integrity of the training process.

6.3 Problem definition and system model

In this section, we define the problem that our proposed defense framework addresses and present the system model that underlies our approach. The problem definition focuses on defending against Gradient Inversion Attacks and Poison Attacks in Federated Learning systems, while the system model provides an overview of the components and interactions that constitute the proposed framework.

6.3.1 Problem Definition

The primary goal of our defensive framework is to safeguard Federated Learning systems within Internet of Things (IoT) contexts from Gradient Inversion Attacks (GIA) and Poison Attacks (PA). Specifically, we strive to:

- Hinder adversaries from manipulating gradient updates submitted by their devices in order to reverse the learning process, which would result in the deterioration of the global model's performance and jeopardize the overall system's security.
- Identify and exclude attackers that introduce malicious data samples into the training process, leading the global model to generate inaccurate predictions and adversely affecting its performance.

Numerous *clients* $< C_1, C_2, ..., C_n >$ exist, with each client initially registering in the blockchain network. Following the acquisition of the *model* from the blockchain network, they commence the federated learning training process to obtain the result set $\langle C_{ID}$, $epoch$, $Gradient$, $ID \rangle$. The C_{ID} represents the ID of the client, while the *epoch* and *Gradient* are essential components for updating the global model. The term ID refers to the block ID.

Nevertheless, as previously noted, some attackers within the blockchain network will attempt to employ GIA to extract private information from the *Gradient*. Additionally, there may be attackers who utilize poison attacks and upload the poison set $\langle C_{ID}$, $epoch$, $A(Gradient)$, $ID \rangle$.

To accomplish these objectives, we employ a combination of blockchain technology and differential privacy techniques to guarantee the integrity and authenticity of gradient updates and data samples utilized in the Federated Learning process.

6.3.2 System Model

Our proposed defensive framework is composed of multiple components and interactions that collectively offer robust protection against GIA and PA in Federated Learning systems. The system model includes the following elements:

- Clients: A collection of IoT devices participating in the Federated Learning process, generating and processing data locally. Each client calculates gradient updates based on its local data and shares the set $\langle C_{ID}$, $epoch$, $Gradient$, $ID \rangle$ with the blockchain network.

- Smart Contract: Within the context of blockchain networks, a smart contract is a program designed to independently execute pre-defined logic on the network. The present study proposes a smart contract-enabled system wherein a Client may employ a smart contract to train a local model with Differential Privacy Stochastic Gradient Descent (DPSGD), while also incorporating DPSGD when uploading sets of $\langle C_{ID}$, $epoch$, $Gradient$, $ID \rangle$. Furthermore, the smart contract can detect attackers attempting to engage in poison attacks.

- Public Blockchain: A decentralized and transparent ledger utilized to secure gradient information uploaded by clients via smart contracts. The public blockchain enforces integrity and authenticity checks on the gradient updates, ensuring their validity.

- Private Blockchain: A permissioned and secure ledger employed during the Federated Learning process to directly safeguard the gradients. The private blockchain offers additional privacy and security guarantees in comparison to the public blockchain.

- Differential Privacy Stochastic Gradient Descent (DPSGD): A privacy-preserving technique applied to gradient updates in order to protect them from GIA. DPSGD introduces calibrated noise to gradient updates, guaranteeing strong privacy while maintaining model performance.

The interplay of these components establishes the foundation of our defense framework, with the objective of delivering extensive protection against Gradient Inversion Attacks and Poison Attacks in Federated Learning systems. By incorporating blockchain technology and differential privacy methodologies, our framework guarantees the security and privacy of data and gradient updates involved in the learning process, ultimately enhancing the overall dependability and performance of the trained models.

6.4 Proposed System

6.4.1 Overview

The proposed privacy-preserving federated learning framework is designed to facilitate secure and efficient model updating and training while maintaining the confidentiality and privacy of sensitive data. The system employs both public and private blockchains to defend against malicious attacks and ensure secure storage and tracking of result sets. Multiple algorithms operate in concert to create a robust and efficient blockchain-based federated learning system. Figure 6.1 illustrates the comprehensive overview of our proposed system. Further details on the algorithm will be provided in Section 6.4.2.

6.4.2 Implementation of our designed system

6.4.2.1 Registration

Both *Client* and *Agent* must initially register in the blockchain network to participate in the public blockchain. Algorithm 17 outlines the Client Registration procedure, which enables the enrollment of *Client* and *Agent* into the network.

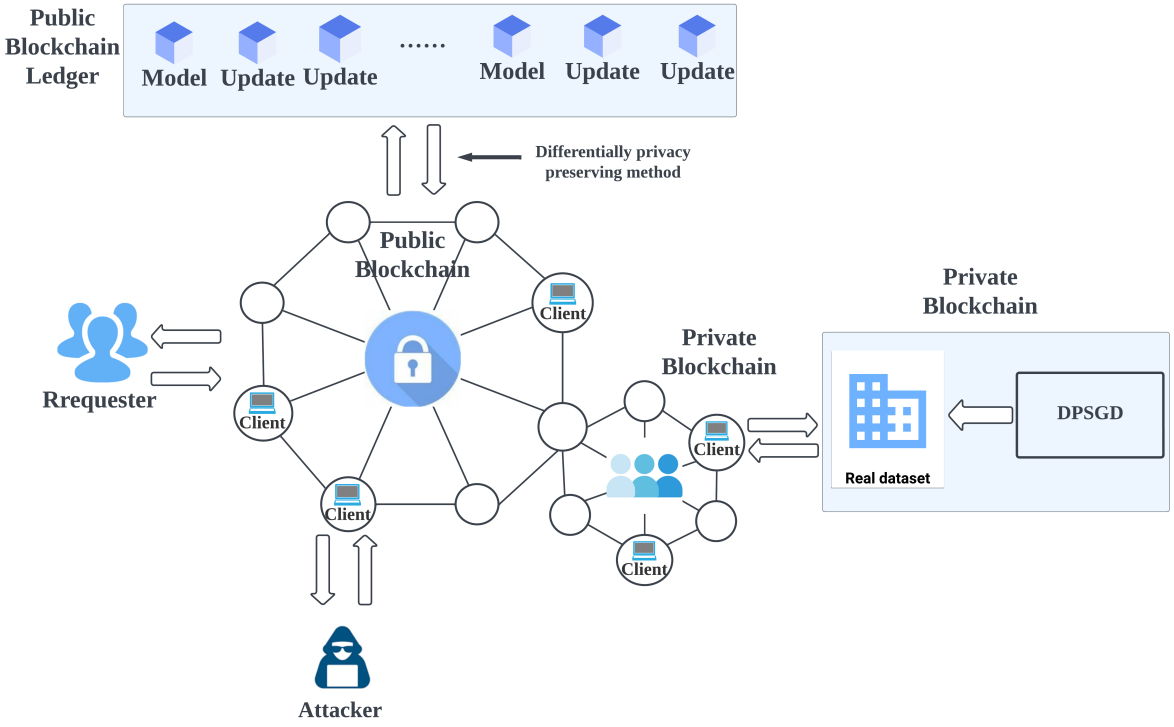

Figure 6.1: Overview of proposed system

Algorithm 17 Client Register

Require: C_{id}, *Agent*, *Org*
Ensure: *RegisterSucess*, *jwt*

RegisterSucess = *False*;
Check Org;
if $C_{id} \in C_{pool}$ **then**
  return C_{id} already existed.
end if
keyGenerator() generated a pair of key according to *Org*
$jwt \leftarrow P_k$, $S_k \leftarrow$*keyGenerator()*;
$C_{id} \leftarrow jwt \leftarrow SC$, $Agent \leftarrow jwt \leftarrow SC$;
$Pool \leftarrow C_{pool} \cup ID_{ui}$;
$Pool \leftarrow Agent \cup ID_{ui}$;
RegisterSuccess = *True*;
return *RegisterSucess*, *jwt*

Initially, the *RegisterSuccess* flag is set to false, and the *SC* verifies whether the *Org* exists. If the C_{id} is already present in the pool of client IDs (C_{pool}), the algorithm returns an error message. Otherwise, the *keyGenerator* produces a pair of public and private keys using the *Org*. The JSON Web Token (jwt), a digital signature employed to authenticate and verify the client's identity, is created by assigning the generated public and private keys to the Agent and C_{id}, respectively. The client ID and Agent are added to the pool of registered clients, and the *RegisterSuccess* flag is set to true. The algorithm then returns the *RegisterSuccess* flag and *jwt*. This procedure enables secure registration of clients into the network with proper authentication measures in place.

6.4.2.2 Global Model Upload

Following successful registration in the blockchain network, the *Agent* can upload the global model. Algorithm 18 delineates the Global Model Upload procedure, enabling the Agent to upload a global model to the network. The input parameters for the algorithm include the Agent's *jwt* token and the global model ($Model_g$) to be uploaded. The algorithm's output consists of a success flag *UploadModel* and the uploaded global model (*Model*).

Algorithm 18 Global Model Upload

Require: *Agent jwt token*, $Model$
Ensure: *UploadModel, Model*
1: *UploadModel* = False;
2: *Agent* send model to SC
3: **if** *jwt token ineligibility* **then**
4: **return** *Agent jwt token* expired
5: **end if**
6: *SC* upload the gloabl model *Model*
7: $Model_g \leftarrow Model$;
8: *UploadModel* = *True*;
9: **return** *UploadModel, Model*

Initially, the *UploadModel* flag is set to false, and the Agent submits the global model to the Smart Contract (SC) for upload. If the *jwt* token provided by the Agent is invalid, the algorithm returns an error message indicating that the Agent's *jwt* token has expired. If the *jwt* token is valid, the SC uploads the global model to the network. The global model is then stored as $Model_g$, and the *UploadModel* flag is set to true. Finally, the algorithm returns the *UploadModel* flag and the uploaded global model. This procedure

ensures the secure upload of global models to the network with proper authentication measures in place.

6.4.2.3 Model Updating Setup

After completing the global model setup, it is essential to establish the epoch settings for the federated learning training process. Algorithm 19 outlines the Model Updating Setup process, which determines the epoch and batch size parameters required for model training. The input parameters of the algorithm include the epoch and batch size, and the output consists of a success flag *UpdatingSetup* and the epoch parameter.

Algorithm 19 Model Updating Setup

Require: $epoch$, $batchsize$
Ensure: *UpdatingSetup*, $epoch$, $batchsize$
1: *UpdatingSetup* = False;
2: Set $epoch$ and $batchsize$ to SC
3: **if** $Public_{epoch}$ is multiple of $Private_{epoch}$ **then**
4: *UpdatingSetup* = *True*
5: **end if**
6: **return** *UpdatingSetup*, *epoch*

In step 1, the *UpdatingSetup* flag is set to false. The *epoch* and *batch size* parameters are then established in the Smart Contract (SC). If the Public epoch value is a multiple of the Private epoch value, the *UpdatingSetup* flag is set to true. This condition is verified to ensure that the private epoch value divides the public epoch value without a remainder, signifying that the private model training is synchronized with the global model updates. Lastly, the algorithm returns the UpdatingSetup flag and the epoch parameter. This process aids in ensuring the accuracy and consistency of model training by maintaining synchronization between the private and public *epoch* values.

6.4.2.4 Private Chain Establishment

Algorithm 20 describes the process of creating a private chain and uploading a model to it. The input parameters for the algorithm include the Agent's *jwt* token and the private model (Model) to be uploaded. The algorithm's output consists of a success flag *UploadPrivate* and the uploaded private model ($Model_p$).

Initially, the *UploadPrivate* indicator is set to false, and the Agent uploads the model to the private chain (Steps 1-2). An authentication process is also required in this procedure (Steps 3-5). If the *jwt* token is valid, the model is stored as $Model_p$ in the

Algorithm 20 Upload to the Private Chain

Require: *jwt token*, $Model$
Ensure: *UploadPrivate,* $Model_p$
1: *UploadPrivate* = False;
2: *Agent* upload the *Model* to private chain
3: **if** *jwt token ineligibility* **then**
4: **return** *Agent jwt token* expired
5: **end if**
6: $Model_p \leftarrow Model$;
7: *UploadPrivate* = *True*;
8: **return** *UploadPrivate, Model*

private chain. Subsequently, the *UploadPrivate* indicator is set to true, and the algorithm returns the *UploadPrivate* flag and the uploaded private model. This process facilitates the secure storage of private models on the private chain, ensuring confidentiality and data privacy.

6.4.2.5 Privacy Preservation in Public Chain

To defend against gradient inversion attacks, we propose Algorithm 21, which details how we counter such attacks at the public chain level. The input parameters of the algorithm include the Client and a results set $\langle C_{ID}, epoch, Gradient, ID\rangle$ containing the gradient values. The output of the algorithm consists of a privacy-preserving results set $\langle C_{ID}, epoch, DP(Gradient), ID\rangle$, where DP(Gradient) represents the gradient values with added Laplacian noise for privacy preservation.

Algorithm 21 Privacy Preserving for Gradient in Public Chain

Require: $Client$, Results set $\langle C_{ID}$, *epoch, Gradient, ID*$\rangle$
Ensure: The privacy preserving set $\langle C_{ID}$, *epoch,* $DP(Gradient)$, *ID*$\rangle$
1: Client finish the training process and send the results set $\langle C_{ID}$, *epoch, Gradient, ID*$\rangle$ to SC.
2: **if** *jwt token ineligibility* **then**
3: **return** *Client jwt token* expired
4: **end if**
5: SC generates the noise $Lap_j(\frac{\Delta Q}{\epsilon})$ according to the $Gradient$.
6: SC add noise $Lap_j(\frac{\Delta Q}{\epsilon})$ to $Gradient$ in the result set to achieve $DP(Gradient)$.
7: SC stores result set $\langle C_{ID}$, *epoch,* $DP(Gradient)$, *ID*$\rangle$ on blockchain.
8: **return** Result set $\langle B_{ID}$, *epoch,* $DP(Gradient)$, *ID*$\rangle$

Initially, the Client completes the training process and sends the results set $\langle C_{ID}$, *epoch, Gradient, ID*$\rangle$ to the Smart Contract (SC). The authentication process is then

initiated. If the jwt token is valid, the SC generates Laplacian noise $Lap_j(\frac{\Delta Q}{\epsilon})$ based on the gradient values and adds it to the gradient values in the results set to achieve $DP(Gradient)$ (Steps 5-6). The SC stores the privacy-preserving results set $\langle C_{ID}, epoch, DP(Gradient), ID\rangle$ on the blockchain for secure storage and tracking. Finally, the algorithm returns the privacy-preserving results set $\langle B_{ID}, epoch, DP(Gradient), ID\rangle$. This process ensures privacy preservation for the gradient values in the public chain, safeguarding the confidentiality and privacy of sensitive data.

6.4.2.6 Privacy Preserving for Poison Attack

Algorithm 22 outlines the privacy-preserving process for poison attacks in the Public Chain. We utilize the traceability of the blockchain to defend against poison attacks. The input parameters of the algorithm include the Attacker, the Requester, and a result set $\langle C_{ID}, epoch, A(Gradient), ID\rangle$ containing the poisoned gradient values. The output of the algorithm is a *DefenseIndex*, indicating whether the poison attack was successfully detected and prevented.

Algorithm 22 Privacy Preserving for Poison Attack in Public Chain

Require: *Attacker, Requester*, Results set $\langle C_{ID}, epoch, A(Gradient), ID\rangle$
Ensure: *DefenseIndex*
1: *DefenseIndex* == *False*
2: *Attacker* send the poison results set $\langle C_{ID}, epoch, A(Gradient), ID\rangle$ to *SC*
3: *SC* upload the results set to public chain
4: *Requester* download the set and upload the global model
5: *SC* find the attacker in public blockchain according to the previous results history
6: *SC* remove the attacker's *jwt* token
7: *SC* resend the previous results set to public chain
8: *Requester* re-download the set and re-upload the global model
9: *DefenseIndex* == *True*
10: **return** *DefenseIndex*

Initially, the *DefenseIndex* is set to false. The Attacker sends the poisoned results set containing *A(Gradient)* to the Smart Contract (SC), which uploads the results set to the public chain. The Requester downloads the results set and uploads the global model. Subsequently, the SC checks the previous results history on the public blockchain to identify the Attacker. If the Attacker is identified, the SC removes the attacker's *jwt* token and resends the previous results set to the public chain. The Requester then re-downloads the results set and re-uploads the global model.

If the poison attack is detected and prevented, the *DefenseIndex* is set to true, and the algorithm returns the *DefenseIndex*. This process ensures the detection and prevention of poison attacks, preserving the security and integrity of the data and thwarting malicious actions.

6.4.2.7 Privacy Preservation in Private Chain

Algorithm 23 outlines the privacy-preserving process for gradients at the Private Chain level. The input parameters of the algorithm include the Client and Agent, and a result set $\langle C_{ID}, epoch, Gradient, ID\rangle$ containing the gradient values. The output of the algorithm is a privacy-preserving result set $\langle C_{ID}, epoch, DP(P(Gradient)), ID\rangle$, where DP(P(Gradient)) represents the gradient values with added Laplacian noise for privacy preservation.

Algorithm 23 Privacy Preserving for Gradient in Private Chain

Require: *Client, Agent*, Results set $\langle C_{ID},\ epoch,\ Gradient,\ ID\rangle$
Ensure: The privacy preserving set Result set $\langle C_{ID},\ epoch,\ DP(P(Gradient)),\ ID\rangle$

1: *Client* get the *Model* on private chain
2: SC use private chain privacy preserving method to protect the training results
3: *Client* start federated learning process
4: *Client* finish the training process and send the results set $\langle C_{ID},\ epoch,\ P(Gradient),\ ID\rangle$ to SC.
5: **if** *jwt token ineligibility* **then**
6: **return** *Client jwt token* expired
7: **end if**
8: SC generates the noise $Lap_j(\frac{\Delta Q}{\epsilon})$ according to the $P(Gradient)$.
9: *Agent* send the results set to the public chain
10: SC add noise $Lap_j(\frac{\Delta Q}{\epsilon})$ to $Gradient$ in the result set to achieve $DP(Gradient)$.
11: SC stores result set $\langle C_{ID},\ epoch,\ DP(P(Gradient)),\ ID\rangle$ on blockchain.
12: **return** Result set $\langle B_{ID},\ epoch,\ DP(P(Gradient)),\ ID\rangle$

Initially, the *Client* retrieves the model from the private chain and commences the federated learning process. Upon completion of the training process, the Client sends the results set containing the gradient values $\langle C_{ID}, epoch, P(Gradient), ID\rangle$ to the Smart Contract (SC). If the *jwt* token provided by the Client is invalid, the algorithm returns an error message stating that the Client's *jwt* token has expired. If the *jwt* token is valid, the SC generates Laplacian noise $Lap_j(\frac{\Delta Q}{\epsilon})$ based on the $P(Gradient)$.

Subsequently, the *Agent* sends the results set to the public chain. The SC adds the Laplacian noise to the gradient values in the results set, resulting in *DP(Gradient)*. The

SC stores the privacy-preserving results set $\langle C_{ID}, epoch, DP(P(Gradient)), ID\rangle$ on the blockchain for secure storage and tracking.

In conclusion, the algorithm returns the privacy-preserving results set $\langle B_{ID}, epoch, DP(P(Gradient)), ID\rangle$. This process ensures privacy preservation for the gradient values in the private chain, safeguarding the confidentiality and privacy of sensitive data.

6.5 Privacy, Security and Time Analysis

6.5.1 Privacy Analysis

In this section, we provide a comprehensive privacy analysis of the proposed blockchain and differential privacy-based framework for defending against Gradient Inversion Attacks (GIA) and Poison Attacks (PA) in Federated Learning (FL). Our analysis focuses on evaluating the ability of the framework to preserve the privacy of the data used in the FL process, as well as assessing the effectiveness of the implemented privacy-preserving techniques in preventing data leakage and unauthorized access.

6.5.1.1 Data Privacy in Federated Learning

Federated Learning inherently preserves data privacy by ensuring that raw data remains on local devices, and only gradient updates are shared among participating devices. Our framework builds upon this inherent privacy-preserving characteristic of FL by implementing additional measures to further protect data privacy, as discussed below:

- Secure and private gradient updates: In the proposed framework, we employ Differential Privacy Stochastic Gradient Descent (DPSGD) to protect the gradient updates. DPSGD ensures that the gradient updates shared among devices are secure and private by adding carefully calibrated noise to the updates, thereby providing strong privacy guarantees. This technique protects the privacy of the data used in the FL process without significantly compromising the model's performance.

- Public and private blockchain integration: Our framework utilizes a combination of public and private blockchains to enhance privacy. The public blockchain is used for securely uploading gradient information through smart contracts, while the private blockchain is employed during the Federated Learning process to protect gradients directly. This dual-blockchain approach enhances the privacy and security of the data used in the FL process.

6.5.1.2 Privacy Preservation in the Blockchain-based Framework

In addition to the privacy-preserving techniques inherent to FL, our proposed framework integrates additional privacy-enhancing mechanisms that leverage the features of blockchain technology and differential privacy:

- Confidentiality of smart contracts: Smart contracts play a vital role in automating the enforcement of security policies within our framework. To protect the privacy of sensitive information processed by the smart contracts, we employ cryptographic techniques, such as zero-knowledge proofs (ZKPs), to ensure that the execution of smart contracts does not reveal any sensitive data to unauthorized parties.

- Anonymity and unlinkability: To prevent adversaries from linking gradient updates to specific devices or users, our framework utilizes anonymization techniques and secure aggregation protocols. These mechanisms ensure that the identity of participating devices remains anonymous during the FL process, making it difficult for adversaries to infer sensitive information about individual users based on the gradient updates.

- Data storage and access control: Our framework addresses privacy concerns related to data storage and access control by implementing decentralized and encrypted data storage solutions, such as distributed hash tables (DHTs) or encrypted data shards. Additionally, we utilize attribute-based encryption (ABE) schemes to enforce fine-grained access control policies, ensuring that only authorized devices can access the stored data.

In conclusion, the privacy analysis demonstrates that our proposed blockchain and differential privacy-based framework effectively preserves the privacy of the data used in Federated Learning. By leveraging a combination of cryptographic techniques, privacy-preserving consensus algorithms, secure data storage solutions, and differential privacy, the framework ensures that sensitive information remains protected from unauthorized access and data leakage throughout the FL process, ultimately contributing to the overall security and trustworthiness of the trained models.

6.5.2 Security Analysis

In this section, we present a thorough security analysis of the proposed blockchain and differential privacy-based framework for defending against Gradient Inversion

Attacks (GIA) and Poison Attacks (PA) in Federated Learning (FL). Our analysis focuses on evaluating the robustness and resilience of the framework against various attack scenarios, as well as assessing the effectiveness of the implemented defense mechanisms in ensuring the integrity and authenticity of the gradient updates and data samples used in the FL process.

6.5.2.1 Gradient Inversion Attack (GIA) Defense

The security of our framework against GIA relies on the integration of Differential Privacy Stochastic Gradient Descent (DPSGD) within the FL process, as well as the use of public and private blockchains. To assess the effectiveness of this defense mechanism, we consider the following attack scenarios:

- Adversaries submit manipulated gradient updates: In this scenario, adversaries attempt to compromise the learning process by submitting gradient updates that have been maliciously modified to invert the learning process. Our framework employs DPSGD to add calibrated noise to the gradient updates, making it difficult for adversaries to successfully manipulate the updates without being detected. Furthermore, the framework leverages the public blockchain and smart contracts to ensure the integrity and authenticity of the uploaded gradient information.

- Adversaries attempt to bypass DPSGD: To bypass the DPSGD protection, adversaries could attempt to infer sensitive information about the original gradients by exploiting the added noise. However, DPSGD provides strong privacy guarantees, making it highly challenging for adversaries to recover the original gradients without violating the privacy constraints.

6.5.2.2 Poison Attack (PA) Defense

The security of our framework against PA relies on the robust traceability mechanism enabled by blockchain technology. This mechanism is designed to detect and exclude attackers injecting malicious data samples into the training process. To evaluate the effectiveness of this defense mechanism, we consider the following attack scenarios:

- Adversaries inject poisoned data samples: In this scenario, adversaries attempt to compromise the model's performance and integrity by injecting carefully crafted malicious data samples into the training process. Our framework leverages the traceability features of blockchain technology to track the origin of the data samples

and identify potential attackers. By detecting and excluding the attackers from the learning process, our framework effectively mitigates the impact of PAs.

- Adversaries attempt to forge their identities: To evade detection and exclusion, adversaries could attempt to create multiple fake identities or impersonate legitimate devices. Our framework addresses this threat by implementing robust identity management and authentication schemes, such as public key infrastructure (PKI) and digital signatures, which make it difficult for adversaries to forge their identities or impersonate legitimate devices.

In conclusion, the security analysis demonstrates that our proposed blockchain and differential privacy-based framework effectively defends against Gradient Inversion Attacks and Poison Attacks in Federated Learning. By utilizing a combination of DPSGD, blockchain technology, and robust traceability mechanisms, the framework ensures the integrity and authenticity of the gradient updates and data samples used in the FL process, ultimately improving the overall reliability and security of the trained models.

6.5.3 Time Complexity Analysis

In this section, we present a time complexity analysis of our proposed blockchain-based defense mechanism for federated learning IoT scenarios, taking into account the use of DPSGD for gradient encryption on both the public and private chains.

Let n be the number of clients participating in the federated learning process, and let t be the number of iterations required for the global model to converge. For each iteration, each client computes its local model update, which involves a time complexity of $O(p)$, where p is the number of model parameters.

After computing the local model update, the clients employ DPSGD to encrypt their gradients. The time complexity of DPSGD is $O(p)$ for each client, as it involves adding noise to each gradient component. As there are n clients, the total time complexity for applying DPSGD across all clients is $O(n * p)$.

Next, the clients submit their encrypted gradients to the public and private chains. The time complexity of committing transactions to the blockchain is generally proportional to the number of transactions. However, by parallelizing the submission and processing of transactions, we can reduce the time complexity of this step to $O(1)$ instead of $O(n)$.

Finally, the global model is updated using the aggregated gradients from both the public and private chains. By employing an efficient aggregation mechanism, we can maintain a time complexity of $O(p)$ for this step.

Taking all the steps into account and considering the parallelization and optimization strategies, the overall time complexity of our proposed defense mechanism for one iteration becomes $O(np)$. Over t iterations, the total time complexity is $O(t(np))$, which demonstrates that our approach scales linearly with the number of clients, model parameters, and iterations, offering a practical and efficient solution for securing federated learning systems in IoT scenarios.

6.6 Results and Analysis

6.6.1 Gradient Inversion Attack Defense Performance

Parameters Description: In our experiments related to defense against gradient inversion attacks, certain parameters, namely λ, δ, and ϵ, were of paramount importance. These parameters influence the defense mechanisms' effectiveness.

- λ: With a default value of 0.95, this parameter strikes a balance between the robustness of the defense mechanism and the accuracy of the model. A higher λ prioritizes model accuracy, while a lower value leans towards privacy.

- δ: This parameter represents the noise scale factor essential for ensuring differential privacy. For our experiments, we adopted default values for δ from the set $\{1e-3, 1e-4, 1e-5\}$. Different δ values analyze the trade-off between preserving privacy and maintaining model utility. Lower values of δ generally signify stronger privacy guarantees.

- ϵ: Denoted as the privacy budget, ϵ quantifies the privacy loss in differential privacy. In our experiments, the default value for ϵ is set to 2. A smaller value of ϵ suggests better privacy, but it might come at the expense of data utility or accuracy.

Attacker Scenarios and Metrics: Our investigation delves into various attacker scenarios:

1. **Uninformed Attacker**: This category encompasses attackers who are oblivious to the private BatchNorm statistics or the private labels of the victim batch. Figure

Figure 6.2: The baseline of our proposed system

6.3 for the public blockchain and Figure 6.5 for the private blockchain visually portray the defense efficacy against such attackers.

2. **Informed Attacker**: This attacker is enlightened about the underlying assumptions, encompassing knowledge about the private BatchNorm statistics and the private labels of the victim batch. The resilience of our defense system against such informed adversaries is captured in Figures 6.4 and 6.6 for public and private blockchains, respectively.

The metrics that we employ for evaluation primarily focus on the resemblance between the data reconstructed by the attacker and the original dataset. A closer match suggests potential vulnerabilities, while significant deviations affirm the defense mechanisms' robustness.

In this part, we use the CIFAR10 dataset to evaluate our proposed system. Figure 6.2 illustrates the baseline architecture of our proposed system. As observed in the figure, without any protection and defense mechanism, an attacker can easily recover the dataset through the gradient inversion attack.

However, when our proposed system is implemented to defend against gradient attacks, the results are significantly different. This can be attributed to the two distinct levels of privacy-preserving protection employed in our proposed system, namely the public chain level and the private chain level. The results obtained from these two levels of protection are presented separately in our study. Additionally, our analysis assumes that the attacker is unaware of the private BatchNorm statistics or the private labels of the victim batch, and also considers the scenario where the attacker possesses such information [23].

Figure 6.3: The defence performance of our proposed system without the assumption at public blockchain level

- Public blockchain level protection: The public blockchain level protection employs Algorithm 21 to defend against gradient inversion attacks. In this context, the smart contract aims to safeguard the results set submitted by the *Client* to the public blockchain.

  Figure 6.3 presents the outcomes when the attacker lacks knowledge of the private BatchNorm statistics or the private labels of the victim batch. The figure demonstrates the effectiveness of our public blockchain level protection, as the attacker is virtually incapable of recovering the original data. Only the truck figure bears a resemblance to the original dataset.

  In contrast, Figure 6.4 illustrates the situation where the attacker is aware of the underlying assumptions. The results in this figure are more discernable; however, the attacker remains unable to recover the data from the gradient. Only the bird figure exhibits similarity to the original figure. Thus, our proposed system appears to successfully defend against gradient attacks at the public blockchain level, regardless of the attacker's knowledge of the private labels of the victim batch.

- Private blockchain level protection: With respect to private blockchain level protection, the smart contract not only employs the public chain privacy-preserving mechanism but also incorporates the DPSGD protection method when the *Client* conducts the federated learning training process. Analogous to the public chain level protection, Figure 6.5 displays the results when the attacker is uninformed about the assumptions. The figure reveals that the attacker cannot reconstruct the image from the gradient.

Figure 6.4: The defence performance of our proposed system with the assumption at public blockchain level

Figure 6.5: The defence performance of our proposed system without the assumption at private blockchain level

Figure 6.6 depicts the outcomes when the attacker possesses knowledge of the assumptions. The figure indicates that regardless of whether the attacker knows the private BatchNorm statistics and the private labels of the victim batch, reconstructing the original data from the gradient remains challenging. This outcome is due to the protection provided by the private chain protection method.

6.6.2 Poison Attack Defence Performance

In this section, we demonstrate the efficacy of our proposed system in defending against poison attacks. The analysis employs two distinct datasets, MNIST and CIFAR10, as previously mentioned. We evaluate our system based on two aspects: training loss per epoch and final result accuracy.

Figure 6.6: The defence performance of our proposed system with the assumption at private blockchain level

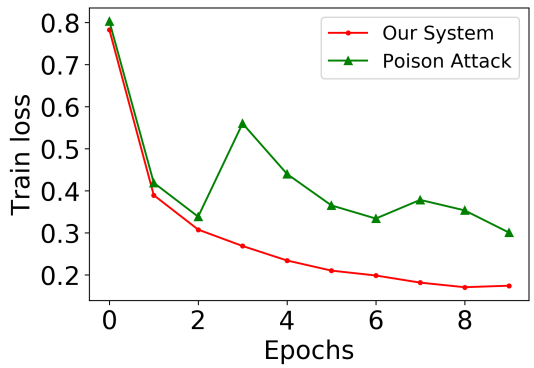

Figure 6.7: Setting with 10 epochs under MNIST dataset

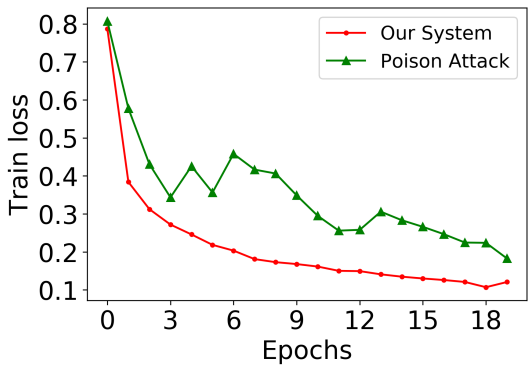

Figure 6.8: Setting with 20 epochs MNIST dataset

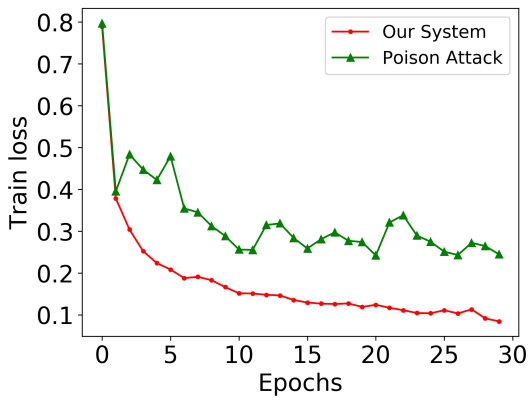

Figure 6.9: Setting with 30 epochs MNIST dataset

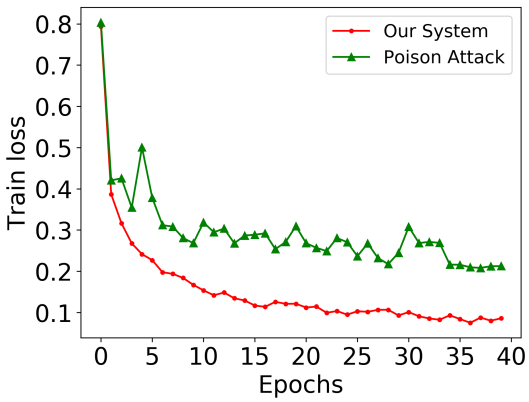

Figure 6.10: Setting with 40 epochs under MNIST dataset

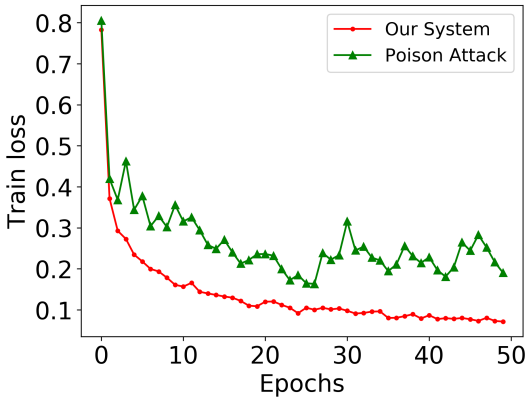

Figure 6.11: Setting with 50 epochs under MNIST dataset

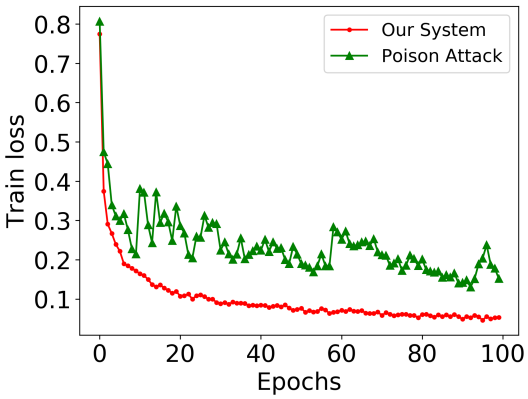

Figure 6.12: Setting with 100 epochs under MNIST dataset

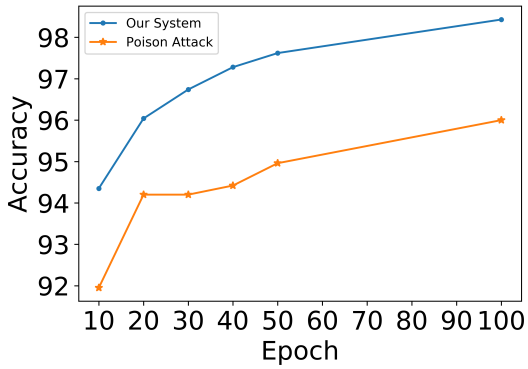

Figure 6.13: Accuracy results of MNIST dataset

- MNIST: For the MNIST dataset, we examine epochs 10, 30, 40, 50, and 100 to display the training loss at each epoch. The green line in the figures represents the normal system's results under poison attack, while the red line corresponds to the system incorporating our proposed blockchain defense method. As illustrated in Figure 6.7-6.12, significant fluctuations in training loss occur, regardless of whether the number of epochs is 10 or 100. The original system's training loss does not drop below 0.2 even after 100 epochs, primarily due to poison attacks in federated learning, causing fluctuations in training loss. Further insights are obtained from the accuracy depicted in Figure 6.13.

  In Figure 6.13, we observe rapid accuracy improvement for both our proposed method and the original system between epochs 10 and 20. However, after 20 epochs, the growth rate in accuracy for both methods slows down. During epochs

20 to 30, the normal method even exhibits a decreasing accuracy trend. After 100 epochs, our proposed method achieves an accuracy of 98.43, while the normal system records an accuracy of 96.

- CIFAR10: For the CIFAR10 dataset, we consider epochs 10, 20, 30, 40, 50, 100, and 200 to showcase the training loss at each epoch. In comparison to the MNIST dataset results, the original system demonstrates stronger fluctuations in Figure 6.14-6.20. From these figures, it is evident that our proposed method offers a superior convergence rate and lower training loss with fewer fluctuations across epochs 10 to 200. Particularly in Figure 6.19 and Figure 6.20, as the number of epochs increases, the normal system displays significant fluctuations in training loss results and only a limited decrease. Analogous to the previous dataset, we present an accuracy figure for the CIFAR10 dataset to provide more detailed results.

  In Figure 6.21, we notice a gradual increase in the accuracy trend of our proposed system, while the original system's accuracy fluctuates. At epochs 50 and 150, the original system's accuracy declines significantly. At epoch 50, our proposed system achieves an accuracy of 55.91, while the original system's accuracy stands at merely 35.12. At epoch 150, our proposed system records an accuracy of 62.58, whereas the original system's accuracy is only 36.16. Finally, at epoch 200, the accuracy of our proposed system reaches 65.2, while the original system attains an accuracy of 49.59.

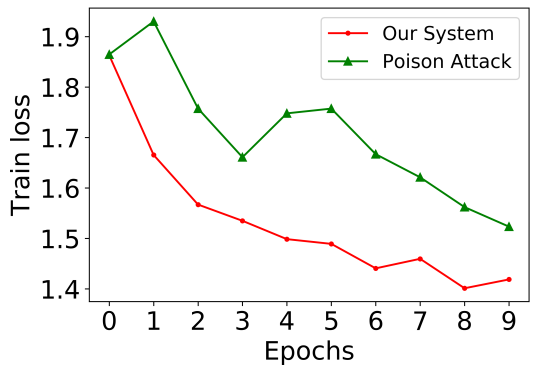

Figure 6.14: Setting with 10 epochs under CIFAR10 dataset

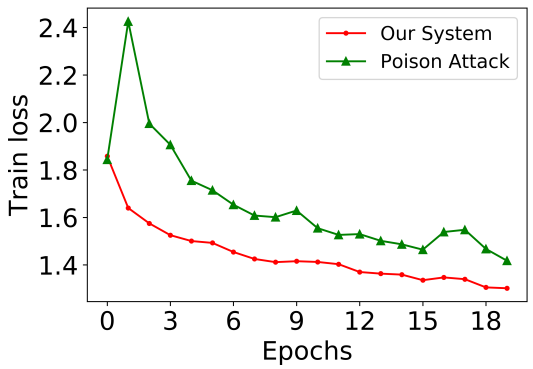

Figure 6.15: Setting with 20 epochs under CIFAR10 dataset

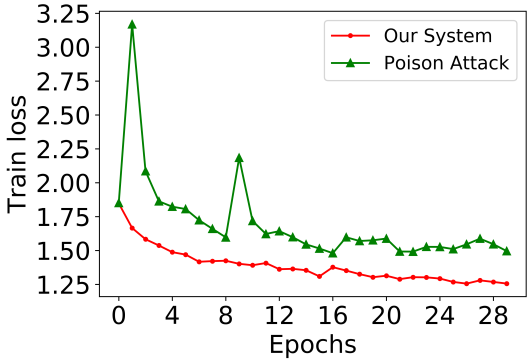

Figure 6.16: Setting with 30 epochs under CIFAR10 dataset

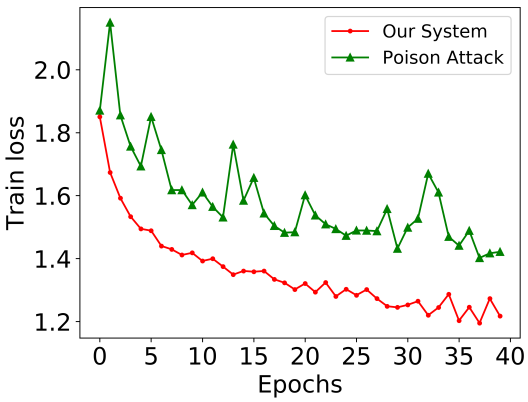

Figure 6.17: Setting with 40 epochs under CIFAR10 dataset

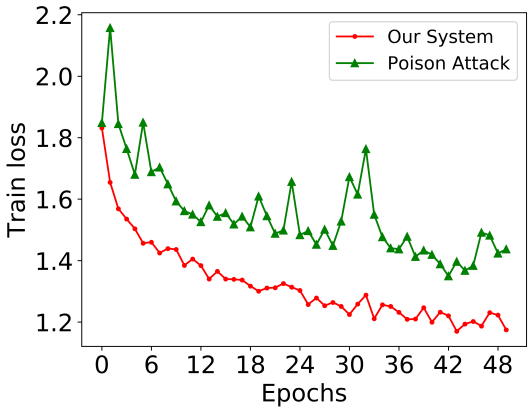

Figure 6.18: Setting with 50 epochs under CIFAR10 dataset

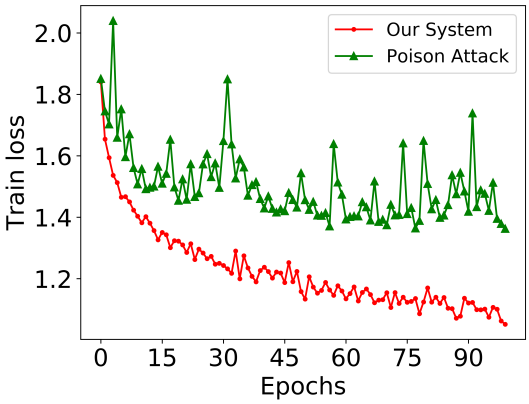

Figure 6.19: Setting with 100 epochs under CIFAR10 dataset

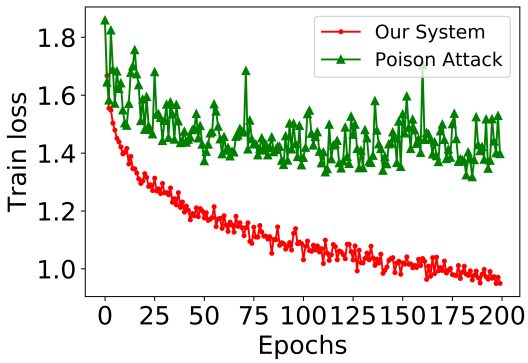

Figure 6.20: Setting with 200 epochs under CIFAR10 dataset

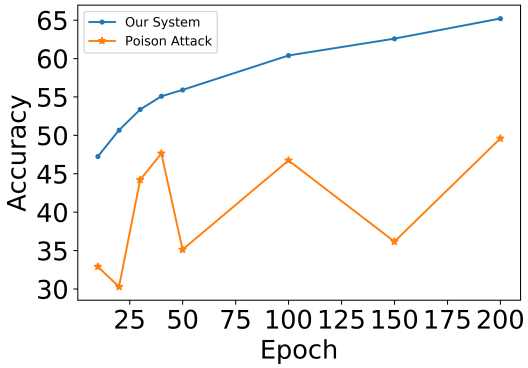

Figure 6.21: Accuracy results of CIFAR10 dataset

6.6.3 Blockchain Platform Time Cost

In this section, we present the experimental results for the blockchain platform time cost, which evaluates the efficiency of our proposed blockchain-based defense mechanism for federated learning IoT scenarios using Hyperledger Fabric 2.X. Our experiments yielded the following observations:

- Blockchain Network Setup Time: The time required to establish the entire blockchain network, including the initialization of channels, peers, and orderers, was approximately 35 seconds. This setup time is a one-time cost for initializing the system and is considered acceptable for practical federated learning IoT applications.
- Consensus Process: The duration associated with the consensus process is contingent upon the particular consensus algorithm implemented in the blockchain platform. In our investigation, subsequent to establishing the blockchain network, the endorsement of all nodes is necessitated by the consensus algorithm, which incurs a time cost of approximately 3 seconds. Our findings indicate that the time cost incurred by the consensus process is reasonably tolerable and acceptable within the context of our federated learning IoT scenario.
- Transaction Processing Time: The time taken for processing and committing transactions, including the local model updates and the aggregation of gradients on the public and private chains, was approximately 2 seconds. This demonstrates the efficiency of Hyperledger Fabric 2.X in handling transactions in our proposed defense mechanism.
- Encryption of Gradient Updates Time: The time cost associated with the encryption of gradient updates using DPSGD was around 1 second. This indicates that our defense mechanism effectively incorporates privacy-preserving techniques without adding significant overhead to the learning process.

Table 6.1 presents the results of the time cost analysis, which includes a comparison of normal federated learning, federated learning with DPSGD defense, and our proposed method. Following the completion of the first iteration, the time cost of normal federated learning is recorded as 45 seconds, while federated learning with DPSGD defense incurs a time cost of 85 seconds, and our proposed system reports a time cost of 134 seconds. The increased time cost observed in our proposed system as compared to federated learning with DPSGD defense is attributable to the fact that our time cost encompasses

Table 6.1: Time Cost of Train with 32 Clients and 199 Iterations

Comparison Methods	t = 0	t = 9	t = 199
Normal Federated Learning	45	450	9000
Federated Learning with DPSGD	85	850	17000
Our Proposed System	134	998	19238

the duration required for blockchain network setup and endorsement of all nodes. At $t = 10$, the results of the analysis suggest that the time cost of our proposed method is more comparable to that of normal federated learning, which recorded a time cost of 998 seconds, as compared to federated learning with DPSGD defense, which incurred a time cost of 850 seconds.

The time costs associated with the Hyperledger Fabric 2.X blockchain platform were within acceptable limits for practical federated learning IoT applications, even when the number of clients and model parameters were large. This demonstrates that our blockchain-based defense mechanism offers an efficient and scalable solution for securing federated learning systems against gradient inversion and poisoning attacks, without significantly impacting the overall performance of the learning process.

6.6.4 Analysis Conclusion

In summary, our proposed blockchain-based defense mechanism exhibits robust defense capabilities against both gradient inversion attacks and poisoning attacks in federated learning IoT scenarios. In the case of gradient inversion attacks, the combination of public and private blockchain level protection, along with the encryption of gradient updates using DPSGD, effectively prevents attackers from reconstructing the figure based on the obtained gradient. As for poisoning attacks, our system demonstrates superior accuracy results and more stable training loss convergence.

Moreover, our experimental results indicate that the time costs associated with the Hyperledger Fabric 2.X blockchain platform, including the network setup, transaction processing, and encryption of gradient updates, are within acceptable limits for practical federated learning IoT applications. This demonstrates that our defense mechanism offers an efficient and scalable solution for securing federated learning systems against gradient inversion and poisoning attacks, without significantly impacting the overall performance of the learning process.

6.7 Summary

In summary, this chapter has proposed a robust blockchain-based solution to defend against gradient inversion attacks and poisoning attacks in federated learning IoT scenarios. By combining public and private blockchain levels of protection, our system effectively mitigates these attacks, ensuring both the privacy and security of gradient updates. Our experimental results demonstrate that our approach significantly hinders attackers from reconstructing figures using the gradients they obtain and provides improved accuracy and stable convergence of training loss under poisoning attacks.

CHAPTER 7

DISCUSSION, FUTURE WORK AND CONCLUSION

7.1 Discussion

This thesis has delved deeply into the complexities of privacy issues in IoT, particularly in the realms of crowdsourcing, multi-agent systems, and federated learning. The analysis has been exhaustive, presenting a vivid portrait of the landscape of privacy-preserving methods, their strengths, and their inherent limitations.

The utilization of differential privacy in frameworks, notably federated learning, has undeniably fortified privacy protections. However, this protective measure is a double-edged sword; while it safeguards sensitive data, the incorporation of noise may compromise the system's performance. Blockchain technology, known for its potential in ensuring data privacy and integrity, emerges as another pivotal component. Yet, its integration into IoT is not without its challenges. Scalability concerns and computational efficiency become especially poignant as the blockchain grows in size.

The intricacies of federated learning, especially when intertwined with IoT, further underscore the nuanced nature of privacy concerns. The decentralized nature of federated learning inherently offers a level of privacy. Still, the interplay between privacy, system performance, and intricacy necessitates a meticulous analysis. It is imperative to address the complexities in federated learning, particularly with the challenges associated with asynchronous updates from multiple devices, skewed data distributions, and potential adversarial attacks.

Our discussion on crowdsourcing in conjunction with IoT elucidated the multifaceted

privacy challenges that arise. Though solutions like data anonymization and aggregation have been instrumental, they are not infallible. The limitations of these techniques emphasize the need for more comprehensive and robust privacy strategies.

Multi-agent systems, when interfaced with IoT, amplify the intricacy of privacy concerns. Techniques rooted in reinforcement learning and trust-centric methodologies highlight the prospects of both learning-driven and decentralized stratagems in fortifying privacy. However, these systems can be vulnerable to different attacks, and their resilience against such threats remains an area that warrants further exploration.

7.2 Limitations

Despite the comprehensive analysis, this thesis has its limitations. The primary limitation revolves around its applicability to a myriad of attacks. While the focus has been predominantly on gradient inversion attacks, other sophisticated attacks may exploit unforeseen vulnerabilities in the proposed solutions.

Another limitation pertains to scalability, especially concerning blockchain technology. As blockchain size increases, potential issues like longer transaction validation times, increased storage requirements, and enhanced computational power demands arise. Addressing these scalability concerns, particularly in real-world IoT applications with millions of interconnected devices, remains a daunting task.

The inherent complexities of federated learning, including the challenges of handling asynchronous updates, addressing skewed data distributions, and mitigating stragglers, have not been extensively explored. These aspects can profoundly impact the overall system performance and privacy, indicating a need for rigorous testing of any proposed solutions against these challenges.

7.3 Future Work

Looking forward, there are several exciting directions for future research in this area. Firstly, while this thesis has largely focused on theoretical analysis and simulations, practical implementations in real-world IoT systems could provide invaluable insights into the functioning and limitations of the proposed privacy-preserving strategies.

Secondly, as edge computing and 5G technologies continue to evolve, exploring their implications for IoT privacy is a promising research direction. For instance, edge com-

puting could potentially enhance privacy by processing data closer to its source, thereby reducing the need for data transmission and the associated privacy risks.

Thirdly, with machine learning techniques becoming increasingly sophisticated, it is important to understand how these advancements can be leveraged to improve privacy protections. For example, homomorphic encryption and secure multi-party computation are areas worth investigating in the context of IoT.

Lastly, the development of robust, universally applicable privacy metrics would contribute significantly to this field, enabling more meaningful comparisons between different privacy-preserving techniques and providing a more solid basis for decision-making in IoT design.

7.4 Conclusion

In conclusion, addressing privacy issues in IoT systems, especially when incorporated with crowdsourcing, multi-agent systems, and federated learning, remains a paramount concern in the advent of an increasingly interconnected digital landscape. This thesis has explored these privacy challenges extensively, providing a critical examination of several privacy-preserving techniques, their strengths, weaknesses, and potential areas for improvement.

Our exploration of integrating crowdsourcing with IoT revealed the complexity of privacy challenges in such scenarios. While several techniques such as data anonymization and data aggregation proved useful, they are not without limitations, underscoring the need for developing more robust and holistic approaches to privacy.

The discussion of multi-agent systems in IoT further amplified the complexity of privacy issues. We explored solutions based on reinforcement learning and trust-based methods, indicating the potential for learning-based and decentralized approaches in enhancing privacy.

The investigation into federated learning with IoT revealed the potential of this decentralized learning method in privacy preservation. Our discussions shed light on various ways to augment federated learning with differential privacy and blockchain technology for better privacy protection. Nevertheless, the trade-offs between privacy, system performance, and complexity require careful consideration.

Looking forward, as edge computing, 5G technology, and more advanced machine learning techniques continue to evolve, the landscape of IoT and associated privacy concerns will likewise continue to change. Future research efforts need to keep pace with

these advancements, exploring their implications for IoT privacy, and innovating newer, more robust privacy-preserving strategies.

In this thesis, we have navigated the intricate landscape of privacy in IoT systems. It is our hope that this work will stimulate further research in this crucial area, aiding in the development of IoT systems that respect and protect user privacy while offering the many benefits that IoT promises. The goal is not just to keep pace with the evolving privacy challenges but to anticipate and address them proactively, thereby ensuring the sustainable growth of IoT systems in various sectors. By integrating privacy-preserving measures into the fabric of IoT design and operation, we can build a more secure and privacy-aware digital future.

APPENDIX

A.1 Notations of This Thesis

Table A.1: Notations

Notations	Description
U_{type}	The type of the user
U_{id}	The ID of the user
P_k	The public key of the user
S_k	The private key of the user
jwt	A jwt token which used to verified user's identity
Org	The organization of the user
T	The requirement of the task
t	The due time of the task
ID_t	The ID of the task
R	The reward of the task
S	The status of the task
Res	Worker's response message set
c	The chosen indicator of the user
SC	The smart contract of the blockchain network
W_i	The unique ID of the worker who finish the task T

Table A.1: Notations

Notations	Description
$Task_n$	The ID of the task
$E_i^{Task_n}$	The energy cost of the worker to finish the task
$T_i^{Task_n}$	The time consumption of the worker to finish the task
$Lap_j(\frac{\Delta Q}{\epsilon})$	The Laplace noise
$Dir()$	The Dirichlet noise
$Agent$	The agent of multi-agent system
$Agent_{id}$	The ID of the agent
U_{pool}	The pool of user
$Pool_{agent}$	The pool of agent
ID_{pool}	The pool of all user
$Requester_{id}$	The ID of the requester
$Stage$	The current stage of the requester
$Advice$	The advice which given by other agent
$Times$	The time of agent been asked for advice
$Reward$	The total reward of the agent
$Rating$	The rating of the agent
U_{name}	The name of the user
$Pool_u$	The pool of the user
$Model$	The model on the public chain
$Model_g$	The global model
$Model_p$	The model on the private chain
$Public_{epoch}$	The epoch of public chain
$Private_{epoch}$	The epoch of private chain
$epoch$	The epoch of federated learning
P_n	The node of private chain
G_n	The node of public chain
$Model_f$	The final model
C_{id}	The ID of client
C_{pool}	The pool of the client
$Pool$	The poll of all users
$batchsize$	The batch size of federated learning
$DP()$	The differential privacy preserving method

Table A.1: Notations

Notations	Description
$Gradient$	The gradient of federated learning training process
B_{ID}	The ID of the block
$A(Gradient)$	The result of attacked gradient
V	The set of nodes representing IoT devices
E	The set of edges representing the communication links between the devices
BC	Blockchain
B_n	The number of blocks
H_i	The hash of the block
T_i	The timestamp of the block
tx	The information of transactions
$\mathscr{M}$	The randomized mechanism
D	One of the data-set
Pr[·]	The probability of [·]
ϵ	The parameter of differential privacy
δ	The parameter of differential privacy

BIBLIOGRAPHY

[1] S. A. ABEYRATNE AND R. P. MONFARED, *Blockchain ready manufacturing supply chain using distributed ledger*, International journal of research in engineering and technology, 5 (2016), pp. 1–10.

[2] F. AL-TURJMAN, M. H. NAWAZ, AND U. D. ULUSAR, *Intelligence in the internet of medical things era: A systematic review of current and future trends*, Computer Communications, 150 (2020), pp. 644–660.

[3] M. S. ALI, M. VECCHIO, M. PINCHEIRA, K. DOLUI, F. ANTONELLI, AND M. H. REHMANI, *Applications of blockchains in the internet of things: A comprehensive survey*, IEEE Communications Surveys & Tutorials, 21 (2018), pp. 1676–1717.

[4] T. ALLADI, V. CHAMOLA, B. SIKDAR, AND K.-K. R. CHOO, *Consumer iot: Security vulnerability case studies and solutions*, IEEE Consumer Electronics Magazine, 9 (2020), pp. 17–25.

[5] K. L. M. ANG, J. K. P. SENG, AND E. NGHARAMIKE, *Towards crowdsourcing internet of things (crowd-iot): Architectures, security and applications*, Future Internet, 14 (2022), p. 49.

[6] S. M. H. BAMAKAN, A. MOTAVALI, AND A. B. BONDARTI, *A survey of blockchain consensus algorithms performance evaluation criteria*, Expert Systems with Applications, 154 (2020), p. 113385.

[7] K. BONAWITZ, H. EICHNER, W. GRIESKAMP, D. HUBA, A. INGERMAN, V. IVANOV, C. KIDDON, J. KONEČNÝ, S. MAZZOCCHI, B. MCMAHAN, ET AL., *Towards federated learning at scale: System design*, Proceedings of machine learning and systems, 1 (2019), pp. 374–388.

[8] E. BORGIA, *The internet of things vision: Key features, applications and open issues*, Computer Communications, 54 (2014), pp. 1–31.

[9] D. CALVARESI, Y. MUALLA, A. NAJJAR, S. GALLAND, AND M. SCHUMACHER, *Explainable multi-agent systems through blockchain technology*, in Explainable, Transparent Autonomous Agents and Multi-Agent Systems: First International Workshop, EXTRAAMAS 2019, Montreal, QC, Canada, May 13–14, 2019, Revised Selected Papers 1, Springer, 2019, pp. 41–58.

[10] Z. CHANG, S. LIU, X. XIONG, Z. CAI, AND G. TU, *A survey of recent advances in edge-computing-powered artificial intelligence of things*, IEEE Internet of Things Journal, 8 (2021), pp. 13849–13875.

[11] S. CHEN, H. XU, D. LIU, B. HU, AND H. WANG, *A vision of iot: Applications, challenges, and opportunities with china perspective*, IEEE Internet of Things journal, 1 (2014), pp. 349–359.

[12] Z. CHEN, H. CUI, E. WU, AND X. YU, *Dynamic asynchronous anti poisoning federated deep learning with blockchain-based reputation-aware solutions*, Sensors, 22 (2022), p. 684.

[13] G. CORMODE, S. JHA, T. KULKARNI, N. LI, D. SRIVASTAVA, AND T. WANG, *Privacy at scale: Local differential privacy in practice*, in Proceedings of the 2018 International Conference on Management of Data, 2018, pp. 1655–1658.

[14] M. CROSBY, P. PATTANAYAK, S. VERMA, V. KALYANARAMAN, ET AL., *Blockchain technology: Beyond bitcoin*, Applied Innovation, 2 (2016), p. 71.

[15] E. CULLINA, K. CONBOY, AND L. MORGAN, *Measuring the crowd: a preliminary taxonomy of crowdsourcing metrics*, in Proceedings of the 11th international symposium on open collaboration, 2015, pp. 1–10.

[16] S. DURI, M. GRUTESER, X. LIU, P. MOSKOWITZ, R. PEREZ, M. SINGH, AND J.-M. TANG, *Framework for security and privacy in automotive telematics*, in Proceedings of the 2nd international workshop on Mobile commerce, 2002, pp. 25–32.

[17] C. DWORK, F. MCSHERRY, K. NISSIM, AND A. SMITH, *Calibrating noise to sensitivity in private data analysis*, in Theory of cryptography conference, Springer, 2006, pp. 265–284.

[18] Ú. ERLINGSSON, V. PIHUR, AND A. KOROLOVA, *Rappor: Randomized aggregatable privacy-preserving ordinal response*, in Proceedings of the 2014 ACM SIGSAC conference on computer and communications security, 2014, pp. 1054–1067.

[19] E. ESTELLÉS-AROLAS AND F. GONZÁLEZ-LADRÓN-DE GUEVARA, *Towards an integrated crowdsourcing definition*, Journal of Information science, 38 (2012), pp. 189–200.

[20] C. FANG, Y. GUO, J. MA, H. XIE, AND Y. WANG, *A privacy-preserving and verifiable federated learning method based on blockchain*, Computer Communications, 186 (2022), pp. 1–11.

[21] M. FRUSTACI, P. PACE, G. ALOI, AND G. FORTINO, *Evaluating critical security issues of the iot world: Present and future challenges*, IEEE Internet of things journal, 5 (2017), pp. 2483–2495.

[22] X. GAN, Y. LI, Y. HUANG, L. FU, AND X. WANG, *When crowdsourcing meets social iot: An efficient privacy-preserving incentive mechanism*, IEEE Internet of Things Journal, 6 (2019), pp. 9707–9721.

[23] J. GEIPING, H. BAUERMEISTER, H. DRÖGE, AND M. MOELLER, *Inverting gradients-how easy is it to break privacy in federated learning?*, Advances in Neural Information Processing Systems, 33 (2020), pp. 16937–16947.

[24] J. GENG, Y. MOU, Q. LI, F. LI, O. BEYAN, S. DECKER, AND C. RONG, *Improved gradient inversion attacks and defenses in federated learning*, IEEE Transactions on Big Data, (2023).

[25] S. GRONAUER AND K. DIEPOLD, *Multi-agent deep reinforcement learning: a survey*, Artificial Intelligence Review, (2022), pp. 1–49.

[26] H. HALPIN AND M. PIEKARSKA, *Introduction to security and privacy on the blockchain*, in 2017 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW), IEEE, 2017, pp. 1–3.

[27] V. HASSIJA, V. CHAMOLA, V. SAXENA, D. JAIN, P. GOYAL, AND B. SIKDAR, *A survey on iot security: application areas, security threats, and solution architectures*, IEEE Access, 7 (2019), pp. 82721–82743.

[28] J. HOWE ET AL., *The rise of crowdsourcing*, Wired magazine, 14 (2006), pp. 1–4.

[29] M. HUANG, S. CAO, X. LI, K. HUANG, AND X. ZHANG, *Defending data poisoning attack via trusted platform module and blockchain oracle*, in ICC 2022-IEEE International Conference on Communications, IEEE, 2022, pp. 1245–1250.

[30] Y. HUANG, S. GUPTA, Z. SONG, K. LI, AND S. ARORA, *Evaluating gradient inversion attacks and defenses in federated learning*, Advances in Neural Information Processing Systems, 34 (2021), pp. 7232–7241.

[31] J. JEON, K. LEE, S. OH, J. OK, ET AL., *Gradient inversion with generative image prior*, Advances in neural information processing systems, 34 (2021), pp. 29898–29908.

[32] S. JI, J. ZHANG, Y. ZHANG, Z. HAN, AND C. MA, *Lafed: A lightweight authentication mechanism for blockchain-enabled federated learning system*, Future Generation Computer Systems, 145 (2023), pp. 56–67.

[33] A. KANTAMNENI, L. E. BROWN, G. PARKER, AND W. W. WEAVER, *Survey of multi-agent systems for microgrid control*, Engineering applications of artificial intelligence, 45 (2015), pp. 192–203.

[34] A. KAPITONOV, S. LONSHAKOV, A. KRUPENKIN, AND I. BERMAN, *Blockchain-based protocol of autonomous business activity for multi-agent systems consisting of uavs*, in 2017 Workshop on Research, Education and Development of Unmanned Aerial Systems (RED-UAS), IEEE, 2017, pp. 84–89.

[35] M. A. KHAN AND K. SALAH, *Iot security: Review, blockchain solutions, and open challenges*, Future generation computer systems, 82 (2018), pp. 395–411.

[36] S. A. KHOWAJA, I. H. LEE, K. DEV, M. A. JARWAR, AND N. M. F. QURESHI, *Get your foes fooled: Proximal gradient split learning for defense against model inversion attacks on iomt data*, IEEE Transactions on Network Science and Engineering, (2022).

[37] H. KIM, J. PARK, M. BENNIS, AND S.-L. KIM, *Blockchained on-device federated learning*, IEEE Communications Letters, 24 (2019), pp. 1279–1283.

[38] K. KIMANI, V. ODUOL, AND K. LANGAT, *Cyber security challenges for iot-based smart grid networks*, International journal of critical infrastructure protection, 25 (2019), pp. 36–49.

[39] T. LABEODAN, K. ADUDA, G. BOXEM, AND W. ZEILER, *On the application of multi-agent systems in buildings for improved building operations, performance and smart grid interaction–a survey*, Renewable and Sustainable Energy Reviews, 50 (2015), pp. 1405–1414.

[40] J. LENG, G. RUAN, P. JIANG, K. XU, Q. LIU, X. ZHOU, AND C. LIU, *Blockchain-empowered sustainable manufacturing and product lifecycle management in industry 4.0: A survey*, Renewable and sustainable energy reviews, 132 (2020), p. 110112.

[41] C. LEPORE, M. CERIA, A. VISCONTI, U. P. RAO, K. A. SHAH, AND L. ZANOLINI, *A survey on blockchain consensus with a performance comparison of pow, pos and pure pos*, Mathematics, 8 (2020), p. 1782.

[42] M. LI, J. WENG, A. YANG, W. LU, Y. ZHANG, L. HOU, J.-N. LIU, Y. XIANG, AND R. H. DENG, *Crowdbc: A blockchain-based decentralized framework for crowdsourcing*, IEEE Transactions on Parallel and Distributed Systems, 30 (2018), pp. 1251–1266.

[43] W. LI, Z. SU, R. LI, K. ZHANG, AND Y. WANG, *Blockchain-based data security for artificial intelligence applications in 6g networks*, IEEE Network, 34 (2020), pp. 31–37.

[44] Z. LI, S. BAHRAMIRAD, A. PAASO, M. YAN, AND M. SHAHIDEHPOUR, *Blockchain for decentralized transactive energy management system in networked microgrids*, The Electricity Journal, 32 (2019), pp. 58–72.

[45] C. LIANG, B. SHANMUGAM, S. AZAM, A. KARIM, A. ISLAM, M. ZAMANI, S. KAVIANPOUR, AND N. B. IDRIS, *Intrusion detection system for the internet of things based on blockchain and multi-agent systems*, Electronics, 9 (2020), p. 1120.

[46] W. Y. B. LIM, N. C. LUONG, D. T. HOANG, Y. JIAO, Y.-C. LIANG, Q. YANG, D. NIYATO, AND C. MIAO, *Federated learning in mobile edge networks: A comprehensive survey*, IEEE Communications Surveys & Tutorials, 22 (2020), pp. 2031–2063.

[47] Q. LIU, Y. TIAN, J. WU, T. PENG, AND G. WANG, *Enabling verifiable and dynamic ranked search over outsourced data*, IEEE Transactions on Services Computing, (2019).

[48] G. LU, Z. XIONG, R. LI, AND W. LI, *Decentralized federated learning: A defense against gradient inversion attack*, in Wireless Internet: 15th EAI International Conference, WiCON 2022, Virtual Event, November 2022, Proceedings, Springer, 2023, pp. 44–56.

[49] Y. LU, X. HUANG, Y. DAI, S. MAHARJAN, AND Y. ZHANG, *Blockchain and federated learning for privacy-preserved data sharing in industrial iot*, IEEE Transactions on Industrial Informatics, 16 (2019), pp. 4177–4186.

[50] F. LUO, Z. Y. DONG, G. LIANG, J. MURATA, AND Z. XU, *A distributed electricity trading system in active distribution networks based on multi-agent coalition and blockchain*, IEEE Transactions on Power Systems, 34 (2018), pp. 4097–4108.

[51] Y. MA, Y. SUN, Y. LEI, N. QIN, AND J. LU, *A survey of blockchain technology on security, privacy, and trust in crowdsourcing services*, World Wide Web, 23 (2020), pp. 393–419.

[52] H. A. MADNI, R. M. UMER, AND G. L. FORESTI, *Blockchain-based swarm learning for the mitigation of gradient leakage in federated learning*, IEEE Access, 11 (2023), pp. 16549–16556.

[53] A. R. MADUSHANKI, M. N. HALGAMUGE, W. S. WIRASAGODA, AND A. SYED, *Adoption of the internet of things (iot) in agriculture and smart farming towards urban greening: A review*, International Journal of Advanced Computer Science and Applications, 10 (2019), pp. 11–28.

[54] Y. MEZQUITA, A. S. GAZAFROUDI, J. M. CORCHADO, M. SHAFIE-KHAH, H. LAAKSONEN, AND A. KAMIŠALIĆ, *Multi-agent architecture for peer-to-peer electricity trading based on blockchain technology*, in 2019 XXVII International Conference on Information, Communication and Automation Technologies (ICAT), IEEE, 2019, pp. 1–6.

[55] M. H. MIRAZ, M. ALI, P. S. EXCELL, AND R. PICKING, *A review on internet of things (iot), internet of everything (ioe) and internet of nano things (iont)*, 2015 Internet Technologies and Applications (ITA), (2015), pp. 219–224.

[56] I. MISTRY, S. TANWAR, S. TYAGI, AND N. KUMAR, *Blockchain for 5g-enabled iot for industrial automation: A systematic review, solutions, and challenges*, Mechanical systems and signal processing, 135 (2020), p. 106382.

[57] S. NAKAMOTO, *Bitcoin: A peer-to-peer electronic cash system*, Decentralized Business Review, (2008), p. 21260.

[58] D. NGUYEN, M. DING, P. PATHIRANA, A. SENEVIRATNE, J. LI, AND V. POOR, *Cooperative task offloading and block mining in blockchain-based edge computing with multi-agent deep reinforcement learning*, IEEE Transactions on Mobile Computing, (2021).

[59] D. C. NGUYEN, M. DING, Q.-V. PHAM, P. N. PATHIRANA, L. B. LE, A. SENEVIRATNE, J. LI, D. NIYATO, AND H. V. POOR, *Federated learning meets blockchain in edge computing: Opportunities and challenges*, IEEE Internet of Things Journal, 8 (2021), pp. 12806–12825.

[60] B. NIU, Z. ZHANG, X. LI, AND H. LI, *Privacy-area aware dummy generation algorithms for location-based services*, in 2014 IEEE International Conference on Communications (ICC), IEEE, 2014, pp. 957–962.

[61] K. K. PATEL, S. M. PATEL, AND P. SCHOLAR, *Internet of things-iot: definition, characteristics, architecture, enabling technologies, application & future challenges*, International journal of engineering science and computing, 6 (2016).

[62] Y. QI, M. S. HOSSAIN, J. NIE, AND X. LI, *Privacy-preserving blockchain-based federated learning for traffic flow prediction*, Future Generation Computer Systems, 117 (2021), pp. 328–337.

[63] Y. QIAN, Y. JIANG, M. S. HOSSAIN, L. HU, G. MUHAMMAD, AND S. U. AMIN, *Privacy-preserving based task allocation with mobile edge clouds*, Information Sciences, 507 (2020), pp. 288–297.

[64] T. QIU, N. CHEN, K. LI, M. ATIQUZZAMAN, AND W. ZHAO, *How can heterogeneous internet of things build our future: A survey*, IEEE Communications Surveys & Tutorials, 20 (2018), pp. 2011–2027.

[65] P. RATTA, A. KAUR, S. SHARMA, M. SHABAZ, AND G. DHIMAN, *Application of blockchain and internet of things in healthcare and medical sector: applications, challenges, and future perspectives*, Journal of Food Quality, 2021 (2021), pp. 1–20.

[66] Y. REN, W. LIU, A. LIU, T. WANG, AND A. LI, *A privacy-protected intelligent crowdsourcing application of iot based on the reinforcement learning*, Future Generation Computer Systems, 127 (2022), pp. 56–69.

[67] S. SAVAZZI, M. NICOLI, AND V. RAMPA, *Federated learning with cooperating devices: A consensus approach for massive iot networks*, IEEE Internet of Things Journal, 7 (2020), pp. 4641–4654.

[68] M. SELIEM, K. ELGAZZAR, AND K. KHALIL, *Towards privacy preserving iot environments: a survey*, Wireless Communications and Mobile Computing, 2018 (2018).

[69] V. SHARMA, I. YOU, D. N. K. JAYAKODY, AND M. ATIQUZZAMAN, *Cooperative trust relaying and privacy preservation via edge-crowdsourcing in social internet of things*, Future Generation Computer Systems, 92 (2019), pp. 758–776.

[70] M. SHAYAN, C. FUNG, C. J. YOON, AND I. BESCHASTNIKH, *Biscotti: A blockchain system for private and secure federated learning*, IEEE Transactions on Parallel and Distributed Systems, 32 (2020), pp. 1513–1525.

[71] A. R. SHORT, H. C. LELIGOU, AND E. THEOCHARIS, *Execution of a federated learning process within a smart contract*, in 2021 IEEE International Conference on Consumer Electronics (ICCE), IEEE, 2021, pp. 1–4.

[72] Z. SUN, Y. WANG, Z. CAI, T. LIU, X. TONG, AND N. JIANG, *A two-stage privacy protection mechanism based on blockchain in mobile crowdsourcing*, International Journal of Intelligent Systems, 36 (2021), pp. 2058–2080.

[73] Z. TIANQING, W. ZHOU, D. YE, Z. CHENG, AND J. LI, *Resource allocation in iot edge computing via concurrent federated reinforcement learning*, IEEE Internet of Things Journal, 9 (2021), pp. 1414–1426.

[74] V. TOLPEGIN, S. TRUEX, M. E. GURSOY, AND L. LIU, *Data poisoning attacks against federated learning systems*, in Computer Security–ESORICS 2020: 25th European Symposium on Research in Computer Security, ESORICS 2020, Guildford, UK, September 14–18, 2020, Proceedings, Part I 25, Springer, 2020, pp. 480–501.

[75] D. UNAL, M. HAMMOUDEH, M. A. KHAN, A. ABUARQOUB, G. EPIPHANIOU, AND R. HAMILA, *Integration of federated machine learning and blockchain for the*

provision of secure big data analytics for internet of things, Computers & Security, 109 (2021), p. 102393.

[76] L. WANG, D. ZHANG, D. YANG, B. Y. LIM, X. HAN, AND X. MA, *Sparse mobile crowdsensing with differential and distortion location privacy*, IEEE Transactions on Information Forensics and Security, 15 (2020), pp. 2735–2749.

[77] M. WANG, T. ZHU, T. ZHANG, J. ZHANG, S. YU, AND W. ZHOU, *Security and privacy in 6g networks: New areas and new challenges*, Digital Communications and Networks, 6 (2020), pp. 281–291.

[78] M. WANG, T. ZHU, X. ZUO, M. YANG, S. YU, AND W. ZHOU, *Differentially private crowdsourcing with the public and private blockchain*, IEEE Internet of Things Journal, (2023).

[79] Q. WANG, Y. ZHANG, X. LU, Z. WANG, Z. QIN, AND K. REN, *Real-time and spatio-temporal crowd-sourced social network data publishing with differential privacy*, IEEE Transactions on Dependable and Secure Computing, 15 (2016), pp. 591–606.

[80] Y. WANG, Q. HE, X. ZHANG, D. YE, AND Y. YANG, *Efficient qos-aware service recommendation for multi-tenant service-based systems in cloud*, IEEE Transactions on Services Computing, 13 (2017), pp. 1045–1058.

[81] Z. WANG, J. HU, R. LV, J. WEI, Q. WANG, D. YANG, AND H. QI, *Personalized privacy-preserving task allocation for mobile crowdsensing*, IEEE Transactions on Mobile Computing, 18 (2018), pp. 1330–1341.

[82] R. WANT, B. N. SCHILIT, AND S. JENSON, *Enabling the internet of things*, Computer, 48 (2015), pp. 28–35.

[83] P. XIONG, G. LI, W. REN, AND T. ZHU, *Lopo: a location privacy preserving path optimization scheme for spatial crowdsourcing*, Journal of Ambient Intelligence and Humanized Computing, (2021), pp. 1–16.

[84] F. YANG, Y. QIAO, S. WANG, C. HUANG, AND X. WANG, *Blockchain and multi-agent system for meme discovery and prediction in social network*, Knowledge-Based Systems, 229 (2021), p. 107368.

[85] M. YANG, T. ZHU, K. LIANG, W. ZHOU, AND R. H. DENG, *A blockchain-based location privacy-preserving crowdsensing system*, Future Generation Computer Systems, 94 (2019), pp. 408–418.

[86] X. YAO, F. FARHA, R. LI, I. PSYCHOULA, L. CHEN, AND H. NING, *Security and privacy issues of physical objects in the iot: Challenges and opportunities*, Digital Communications and Networks, 7 (2021), pp. 373–384.

[87] D. YE, Q. HE, Y. WANG, AND Y. YANG, *An agent-based integrated self-evolving service composition approach in networked environments*, IEEE Transactions on Services Computing, 12 (2016), pp. 880–895.

[88] D. YE, M. ZHANG, AND D. SUTANTO, *Cloning, resource exchange, and relation-adaptation: an integrative self-organisation mechanism in a distributed agent network*, IEEE Transactions on Parallel and Distributed Systems, 25 (2013), pp. 887–897.

[89] D. YE, T. ZHU, S. SHEN, AND W. ZHOU, *A differentially private game theoretic approach for deceiving cyber adversaries*, IEEE Transactions on Information Forensics and Security, 16 (2020), pp. 569–584.

[90] Y. YU, L. GUO, S. LIU, J. ZHENG, AND H. WANG, *Privacy protection scheme based on cp-abe in crowdsourcing-iot for smart ocean*, IEEE Internet of Things Journal, 7 (2020), pp. 10061–10071.

[91] Q. ZHANG, L. T. YANG, Z. CHEN, P. LI, AND M. J. DEEN, *Privacy-preserving double-projection deep computation model with crowdsourcing on cloud for big data feature learning*, IEEE Internet of Things Journal, 5 (2017), pp. 2896–2903.

[92] T. ZHU, D. YE, W. WANG, W. ZHOU, AND P. YU, *More than privacy: Applying differential privacy in key areas of artificial intelligence*, IEEE Transactions on Knowledge and Data Engineering, (2020).

[93] T. ZONTA, C. A. DA COSTA, R. DA ROSA RIGHI, M. J. DE LIMA, E. S. DA TRINDADE, AND G. P. LI, *Predictive maintenance in the industry 4.0: A systematic literature review*, Computers & Industrial Engineering, 150 (2020), p. 106889.

[94] W. ZOU, D. LO, P. S. KOCHHAR, X.-B. D. LE, X. XIA, Y. FENG, Z. CHEN, AND B. XU, *Smart contract development: Challenges and opportunities*, IEEE Transactions on Software Engineering, 47 (2019), pp. 2084–2106.