Contribution to security and privacy in the Blockchain-based Internet of Things : Robustness, Reliability, and Scalability

Julio César Pérez Garcia

AVIGNON UNIVERSITÉ

Thesis

presented at the Université d'Avignon
in Partial Fulfillment of the Requirements for the Degree of

DOCTOR OF PHILOSOPHY

in
Ecole Doctorale 536 : AGROSCIENCES & SCIENCES
Speciality: Computer Science

By

Julio César Pérez García

Contribution to security and privacy in the Blockchain-based Internet of Things: Robustness, Reliability, and Scalability

Research Unity : EA 4128 LIA – Laboratoire Informatique d'Avignon

Jury members :
Reviewers :
Mme. Melek Önen, Teacher-Researcher, EURECOM, France
M. Axel Küpper, Teacher-Researcher, Technische Universität Berlin — T-Labs, Germany
Examiners :
M. Zoubir Mammeri, Teacher-Researcher, IRIT, Université Paul Sabatier, France
M. Jean-Marie Bonnin, Teacher-Researcher, IMT Atlantique, France
M. Yezekael Hayel, Teacher-Researcher, LIA, Université d'Avignon, France
Advisor:
M. Abderrahim Benslimane, Teacher-Researcher, LIA, Université d'Avignon, France

LABORATOIRE INFORMATIQUE D'AVIGNON

Abstract

The Internet of Things (IoT) is a diverse network of objects or "things" typically interconnected via the Internet. Given the sensitivity of the information exchanged in IoT applications, it is essential to guarantee security and privacy. This problem is aggravated by the open nature of wireless communications, and the power and computing resource limitations of most IoT devices. At the same time, existing IoT security solutions are based on centralized architectures, which raises scalability issues and the single point of failure problem, making them susceptible to denial-of-service attacks and technical failures. Blockchain has emerged as an attractive solution to IoT security and centralization issues. Blockchains replicate a permanent, append-only record of all transactions occurring on a network across multiple devices, keeping them synchronized through a consensus protocol.

Blockchain implementation may involve high computational and energy costs for devices. Consequently, solutions based on Fog/Edge computing have been considered in the integration with IoT. This approach shifts the higher computational load and higher energy consumption to the devices with higher resource availability, *i.e.,* Fog/Edge devices. However, the cost of Blockchain utilization must be optimized, especially in the consensus protocol, which significantly influences the overall system performance. Permissioned Blockchains align better with the requirements of IoT applications than Permissionless Blockchains, due to their high transaction processing rate and scalability. This is because the consensus nodes, *i.e.,* Validators, are known and predetermined. In existing consensus protocols used in Permissioned Blockchains, the Validators are usually a predefined or randomly selected set of nodes, which affects both system performance and fairness among users.

The objective of this work is to propose solutions to improve security and privacy within IoT by integrating Blockchain technology, as well as to maximize fairness levels during consensus. The study is organized into two distinct parts: one addresses critical aspects of IoT security and proposes Blockchain-based solutions, while the other part focuses on optimizing fairness among users during the execution of the consensus algorithm on the Blockchain. We present an authentication mechanism inspired by the μTesla authentication protocol, which uses symmetric keys that form a hashchain and achieves asymmetric properties by unveiling the key used a while later. With this mechanism and the use of the Blockchain to store the keys and facilitate authentication, our proposal ensures robust and efficient authentication of devices, without the need for a trusted third party. In addition, we introduce a Blockchain-based key management system for group communications adapted to IoT contexts. The use of Elliptic Curve Cryptography ensures a low computational cost while enabling secure distribution of group keys. In both security solutions, we provide formal and informal proofs of security under the defined attack model. A performance impact analysis and a comparison with existing solutions are also conducted for the proposed solutions, showing that the proposed solutions are secure and efficient and can be used in multiple IoT applications.

The second part of the work proposes an algorithm to select Validator nodes in Permissioned Blockchains maximizing Social Welfare, using α-Fairness as the objective function. A mathematical model of the problem is developed, and a method for finding the solution in a distributed manner is proposed, employing metaheuristic Evolutionary algorithms and a Search-space partitioning strategy. The security of the proposed algorithm and the quality of the solutions obtained are analyzed. As a result of this work, two security protocols for IoT based on Blockchain are introduced, along with a distributed algorithm for maximizing Social Welfare among users in a Permissioned Blockchain network.

Résumé

L'Internet des Objets (IoT, *Internet of Things*) est un réseau diversifié d'objets interconnectés, généralement via l'internet. En raison de la sensibilité des informations échangées dans les applications de IoT, il est essentiel de garantir la sécurité et le respect de la vie privée. Ce problème est aggravé par la nature ouverte des communications sans fil et par les contraintes de puissance et de ressources computationnelles de la plupart des appareils IoT. Parallèlement, les solutions de sécurité IoT existantes sont basées sur des architectures centralisées, ce qui pose des problèmes d'évolutivité et de point de défaillance unique, les rendant sensibles aux attaques par déni de service et aux défaillances techniques. La Blockchain est considérée comme une solution attractive aux problèmes de sécurité et de centralisation de IoT. Les Blockchains reproduisent un enregistrement permanent, en annexe seulement, de toutes les transactions effectuées sur un réseau entre plusieurs appareils, en les maintenant synchronisées par un protocole de consensus.

L'utilisation de la Blockchain peut impliquer des coûts de calcul et d'énergie élevés pour les appareils. Par conséquent, des solutions basées sur *Fog/Edge Computing* ont été envisagées dans le cadre de l'intégration avec l'IoT. Cette approche transfère la charge de calcul et la consommation d'énergie plus élevées vers les dispositifs ayant une plus grande disponibilité de ressources, les dispositifs Fog/Edge. Toutefois, le coût de l'utilisation de la Blockchain doit être optimisé, en particulier dans le protocole de consensus, qui influe considérablement sur les performances globales du système. Les Blockchains avec permission correspondent mieux aux exigences des applications IoT que les Blockchains sans permission, en raison de leur taux élevé de traitement des transactions et de leur scalabilité. En effet, les nœuds de consensus, les validateurs, sont connus et prédéterminés. Dans les protocoles de consensus existants utilisés dans les Blockchains avec permission, les validateurs sont généralement un ensemble de nœuds prédéfinis ou sélectionnés de manière aléatoire, ce qui affecte à la fois les performances du système et l'équité (*Fairness*) entre les utilisateurs.

L'objectif de ce travail est de proposer des solutions pour améliorer la sécurité et la vie privée dans IoT en intégrant la technologie Blockchain, ainsi que pour maximiser les niveaux de fairness pendant le consensus. L'étude est organisée en deux parties distinctes : l'une traite des aspects critiques de la sécurité de IoT et propose des solutions basées sur la Blockchain, tandis que l'autre se concentre sur l'optimisation de la *Fairness* entre les utilisateurs lors de l'exécution de l'algorithme de consensus sur la Blockchain. Nous présentons un mécanisme d'authentification inspiré du protocole d'authentification μTesla, qui utilise des clés symétriques formant une chaîne de hachage et obtient des propriétés asymétriques en dévoilant la clé utilisée un peu plus tard. Grâce à ce mécanisme et à l'utilisation de la Blockchain pour stocker les clés et faciliter l'authentification, notre proposition garantit une authentification robuste et efficace des appareils, sans qu'il soit nécessaire de recourir à un tiers de confiance. En outre, nous présentons un système de gestion des clés basé sur la Blockchain pour les communications de groupe, adapté aux contextes de IoT. L'utilisation de la cryptographie à courbe elliptique garantit un faible coût de calcul tout en permettant une distribution sécurisée des clés de groupe. Dans les deux solutions de sécurité, nous fournissons des preuves formelles et informelles de la sécurité dans le modèle d'attaque défini. Une analyse de l'impact sur la performance et une comparaison avec les solutions existantes sont également menées pour les solutions proposées, montrant que les solutions proposées sont sûres et efficaces et peuvent être utilisées dans de multiples applications IoT.

La deuxième partie du travail propose un algorithme pour sélectionner les nœuds de validation dans les Blockchains permises maximisant le bien-être social, en utilisant α-Fairness comme fonction objective. Un modèle mathématique du problème est développé, et une méthode pour

trouver la solution de manière distribuée est proposée, en utilisant des algorithmes Evolutionnaires métaheuristiques et une stratégie de division de l'espace de recherche. La sécurité de l'algorithme proposé et la qualité des solutions obtenues sont analysées. Le résultat de ce travail est l'introduction de deux protocoles de sécurité pour l'IoT basés sur la Blockchain, ainsi qu'un algorithme distribué pour maximiser le bien-être social parmi les utilisateurs dans un réseau Blockchain avec permission.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Context and Motivation

Today, the Internet of Things (IoT) is attracting a multitude of industrial and research interests. Smaller and smarter devices are being deployed in multiple IoT domains, with applications ranging from precision agriculture, infrastructure monitoring, personal healthcare, and smart cities to military applications. Data gathered by IoT devices can contain sensitive and private information, such as health data, geographic locations, and personal preferences. Exposure or misuse of this information could have serious implications for the privacy and security of users. This has led to the appearance of many security threats that exploit weaknesses in existing IoT infrastructures [1].

Security in the IoT environment presents particular challenges, as IoT devices are inherently vulnerable to attack because they are low-cost and battery-dependent, with little computational storage and memory capacity. In addition, the heterogeneity and ubiquity of these devices make security management complex. These security challenges are aggravated by the unique characteristics of IoT devices, such as their constant connectivity, typically over a wireless channel. These limitations mean that classical cryptographic solutions for security and privacy cannot be directly applied to IoT devices, making it necessary to adopt lightweight cryptographic techniques, such as those based on Elliptic Curves, to ensure efficient and effective encryption methods within the IoT ecosystem. Compared to other public key cryptography schemes, such as the well-known Rivest Shamir Adleman (RSA) public key algorithm, Elliptic Curves Cryptography has faster computation time, smaller keys, and lower memory and bandwidth usage, rendering it very suitable for IoT [2].

Basic security services, such as data privacy, integrity, and authentication, are necessary in IoT communications. Secure, robust, and efficient key management is the basis for ensuring these services. Existing solutions to address key management are typically centralized, where digital certificates are issued by a Certification Authority (CA). However, this centralization poses significant problems, such as a single point of failure, which makes the system susceptible to disruption in the event of denial-of-service (DOS) attacks or technical failures. In addition, a completely centralized network infrastructure leads to higher latency in end-to-end communications, which can hinder vertical IoT applications such as smart cities and healthcare. IoT architectures that employ Fog/Edge computing alleviate the latency issues inherent in a centralized IoT. To improve privacy and security in Fog/Edge architectures, as well as in centralized network architectures, a more decentralized approach is seen as the solution to enable long-term IoT growth and avoid single points of failure [3].

In response to these IoT centralization and security issues, Blockchain technology has emerged as a promising solution. Blockchain is a distributed digital record of cryptographically signed transactions grouped into blocks. Each block is linked to the previous one after validation and subjected to a consensus decision in a peer-to-peer network. As new blocks are added, it becomes more difficult to modify the previous ones (which creates resistance to manipulation). New blocks are replicated through copies of the ledger within the network [4].

The integration of Blockchain and IoT presents multiple challenges such as choosing the right consensus protocol. Blockchain systems are divided into two categories based on participant access: Permissioned Blockchains and Permissionless Blockchains. In Permissionless

Blockchains, each participant keeps a copy of the blockchain data structure in memory and must also participate in the consensus. Generally, these algorithms use proof-based consensus which is very computationally expensive, which makes these blockchains unsuitable for IoT environments. On the contrary, in Permissioned Blockchains only a limited number of known participants carry a copy of the entire blockchain, and the consensus is based on voting, which decreases the computational cost and increases the transaction processing rate, making it more suitable for IoT. The use of Permissioned Blockchains together with Edge/Fog Computing offers an attractive alternative to improve the efficiency of blockchain-based IoT applications. In this approach, IoT devices do not directly perform blockchain-related tasks; instead, these tasks are performed on Edge/Fog devices with better computational performance [5].

Existing consensus mechanisms focus primarily on optimizing processing efficiency and often neglect considerations of fairness among users. This neglect can potentially lead to user demotivation and dissatisfaction. Therefore, efforts to achieve fairness in Blockchain processes are essential to maintain a fair and participatory ecosystem. Some efforts have been made to include fairness in various blockchain processes such as the selection of transactions to be included in a blockchain or the number of transactions in a block for every user. However, in other processes, this fairness is not considered for example in the selection of Validators in the Permissioned Blockchain, which is done by the traditional approach in the form of lotteries or in a rotating way among the nodes that conform to the Blockchain [6].

1.2 Research Objectives

This work investigates and proposes security solutions adapted to IoT scenarios, using lightweight cryptographic techniques that take into account the limitations of IoT devices. Furthermore, we aim to provide a solution for Validator selection in Permissioned Blockchains that considers and optimizes fairness among users. These approaches aim to comprehensively address the security and performance challenges in the context of integrating Blockchain into the Internet of Things, contributing to the advancement and improvement of this fundamental technology in society.

1.3 Thesis Structure

This thesis consists of two major parts, the first part focuses on Blockchain-based security solutions for IoT, and the second part focuses on the Fairness in the Validator Selection Process. The following is an overview of each chapter:

- We begin with Chapter 2, entitled Background and Literature Review, where we establish a background and perform an extensive literature review. This chapter delves into the Internet of Things paradigm, addressing its architecture and necessary characteristics. It then moves on to a comprehensive review of IoT security, covering essential requirements, prevailing challenges, and potential security threats. The chapter also includes an exploration of cryptographic primitives, distinguishing between public-key and symmetric cryptography, as well as an introduction to elliptic curve cryptography. In addition, the chapter delves into an overview of blockchain technology, covering its key properties, classification, consensus protocols, benefits of integration with IoT, and various integration schemes. It also discusses the concept of fairness in blockchain, incorporating discussions on social welfare and fairness metrics. Finally, the chapter culminates with a concluding summary, synthesizing the critical points discussed in the chapter.

- Chapter 3, entitled μTesla-based Authentication for Reliable and Secure Broadcast Communications in the Internet of Drones (IoD) using Blockchain, introduces an authentication scheme inspired by μTesla, which uses Blockchain technology to solve the limitations

of the original protocol while exploiting the features of it. This chapter constitutes the first component of the first part of the thesis and presents some work related to authentication protocols and Blockchain-based existing solutions for broadcast authentication. In addition, some preliminaries related to the cryptographic protocols involved in our approach are introduced. We analyze the security and performance of the proposed solution. Simulation results show that the proposed solution outperforms several approaches in the literature, achieving low authentication delay with a low information exchange while maintaining low computational requirements.

- Chapter 4, entitled Blockchain-based Group Key Mechanism for IoT with Anonymity of Group Members, constitutes the last component of the first part of the thesis. We provide a protocol for a group key management scheme ensuring device anonymity. The proposed scheme utilizes blockchain to facilitate secure group formation and key establishment processes between a Central Authority (CA) and IoT devices. Additionally, the public keys of nodes and groups are stored in the blockchain without reference to the real identity of the nodes. This approach utilizes Elliptic Curve cryptography, allowing it to be performed by limited devices. This chapter also surveys the most relevant existing group key management solutions, presents some preliminaries, and explains the proposed scheme. Analogous to the previous chapter we analyze the security and performance of the scheme. Simulation results demonstrate the efficiency of the proposed scheme, outperforming several existing approaches in the literature in terms of computation and communication costs.

- In the second part of this thesis, we address the problem of Maximizing Social Welfare among Blockchain users. In Chapter 5 entitled Maximizing Social Welfare through Fair Validators Selection in Permissioned Blockchains, we provide a survey of the most relevant Permissioned Blockchains, properties, and the consensus algorithms most commonly used in them. This chapter also introduces Social Welfare and α-Fairness as utility functions to combine fairness and efficiency. In addition, the Evolutionary algorithms employed in the proposed solution are described. The problem of Validator selection turns out to be an NP-hard problem. We propose a solution to solve the problem in a distributed way using Evolutionary algorithms and a Search-space division strategy. An analysis of the security aspects and the impact of the proposed algorithm on system performance. The numerical results show the suggested algorithm enhances Social Welfare for users with rapid convergence. These attributes collectively establish the proposed algorithm as both efficient and secure, making it suitable for a diverse set of Permissioned Blockchain applications.

  Finally, this work ends with a Conclusion and some insights into Future Works.

1.4 List of publications

This work is based on the following publications submitted in International journals:

1. [7] **J. C. Perez-Garcia**, A. Benslimane, A. Braeken, and Z. Su, "μTesla-based Authentication for Reliable and Secure Broadcast Communications in IoD using Blockchain," in IEEE Internet of Things Journal, 2022.

2. (Submitted to IEEE Transactions on Information Forensics and Security) **J. C. Perez-Garcia**, A. Braeken, and A. Benslimane, "Blockchain-Based Group Key Management Scheme for IoT with Anonymity of Group Members".

3. (Submitted to IEEE Transactions on Mobile Computing) **J. C. Perez-Garcia** and A. Benslimane, "FairVSP: Maximizing Social Welfare through Fair Validators Selection in Permissioned Blockchains".

Other publications by the author, presented at International Conferences:

1. [8] **J. C. Perez-Garcia**, A. Benslimane and Z. Su, "Analysis on the AoI in Blockchain-based IoT Networks with Different Sensing Mechanisms," ICC 2022 - IEEE International Conference on Communications, Seoul, Korea, Republic of, 2022, pp. 4763-4768.

2. [9] **J. C. Perez-Garcia**, A. Benslimane and S. Boutalbi, "Blockchain-based system for e-voting using Blind Signature Protocol," 2021 IEEE Global Communications Conference (GLOBECOM), Madrid, Spain, 2021, pp. 1-6.

3. [10] S. Boutalbi, **J. C. Perez-Garcia** and A. Benslimane, "Blockchain-based secure Handover for IoT using Zero-Knowledge Proof protocol," 2021 IEEE Global Communications Conference (GLOBECOM), Madrid, Spain, 2021, pp. 1-6.

Chapter 2

Background and Literature Review

2.1 Introduction

The foundation of this work is laid upon a comprehensive exploration of the existing literature and background in the domain of the Internet of Things (IoT) and its integration with Blockchain technology. The subsequent sections provide a detailed overview of key aspects, starting with the Internet of Things Paradigm, which envisions a world where interconnected devices autonomously collect, exchange, and act upon data. The Architecture of IoT is then scrutinized, unraveling the layers and components that constitute the framework governing these interconnected systems.

The examination of the Requirements of the IoT Architecture follows, emphasizing the need for scalability, interoperability, energy efficiency, and real-time processing to ensure the effectiveness of IoT systems. Security emerges as a paramount concern in the Overview of IoT Security, where the challenges posed by unauthorized access, data breaches, and potential risks to the IoT ecosystem are thoroughly explored.

Delving deeper into the security landscape, the discussion extends to Security Attacks in IoT, identifying and analyzing various types of threats, from denial-of-service to man-in-the-middle attacks. Cryptographic Primitives Overview establishes a foundation for the subsequent exploration of security mechanisms by elucidating the fundamental building blocks of cryptographic techniques. A critical decision in designing secure IoT systems lies in the choice between Public-key vs Symmetric-Key Cryptography, and this section evaluates the strengths and weaknesses of each approach, considering factors such as computational efficiency and key management. The exploration then extends to the advantages of Elliptic Curve Cryptography, particularly its suitability for resource-constrained IoT devices.

The intersection of Blockchain technology with IoT introduces a paradigm shift, explored in Blockchain Technology Overview. This decentralized and tamper-resistant ledger is dissected further in Blockchain Properties, which examines decentralization, immutability, transparency, and consensus mechanisms as inherent characteristics. Consensus Protocols form a crucial element in ensuring the integrity of a Blockchain, with this section exploring various mechanisms. Benefices of the Integration Blockchain-IoT are subsequently discussed, highlighting synergistic advantages such as enhanced security, improved data integrity, and transparency. The concluding section, Fairness Metrics, introduces metrics for measuring Social Welfare and Fairness. These fundamental ideas serve as the basis for the development and better understanding of the proposed security solutions integrating Blockchain and IoT technologies.

2.2 Internet of Things Paradigm

The term ”Internet of Things” (IoT) was first coined in 1999 by K. Ashton [11] as a bridge to connect supply chain RFID to the Internet. Many definitions have subsequently been proposed in the literature [12], [13], all of them agreeing that IoT consists of networked devices that sense and gather data from their surroundings, which are then used to perform automated functions to assist human users. The meaning of the Internet of Things has expanded and now encompasses a wide variety of technologies, objects, and protocols to the extension of the Internet and the Web into the physical realm.

IoT is transforming, sometimes drastically, all industries and markets. In healthcare [14], medical devices IoT-enabled medical devices and wearables record vitals and facilitate remote consultations, opening the way to an era of personalized and proactive healthcare. Precision agriculture harnesses IoT sensors to optimize crop management, leveraging real-time data on soil conditions and weather patterns for sustainable farming practices [15]. Environmental monitoring benefits from IoT technology, providing invaluable insights into air and water quality, and ultimately aiding efforts to combat pollution and climate change [16].

On the other hand, Smart Cities utilize IoT infrastructure to streamline energy consumption, traffic flow, and public safety, making more efficient and sustainable urban environments. Industrial automation, enabled by IoT, ensures seamless operations through predictive maintenance and real-time monitoring of machinery [17]. Supply chain management is transformed with IoT devices tracking goods in transit, guaranteeing quality, and enabling agile inventory management [18]. In smart homes and buildings, IoT systems govern lighting, security, and climate control, optimizing energy usage and enhancing comfort. Retail experiences are enriched by IoT-driven insights, tailoring marketing strategies, and improving customer engagement [19]. Transportation and logistics benefit from IoT sensors optimizing routes, reducing fuel consumption, and bolstering safety measures [20]. All these examples underscore the vast potential of IoT, fundamentally reshaping industries and improving our collective quality of human life.

2.2.1 Architecture of IoT

IoT applications require the connection of billions or trillions of devices through the Internet [21]. Therefore, architectures must be flexible to cope with different requirements for each particular application. There are several different architectures that have not fully converged into a unique reference model. Some of these approaches are shown in Fig. 2.1. The basic model is the three-layer architecture (Fig. 2.1a). Other models (Fig. 2.1b and Fig. 2.1c) have added more abstraction layers to the IoT architecture. After conducting a comprehensive review of various existing IoT architectures, we adopt the Four-layer (Middleware-based) architecture, with four layers as the foundational framework in this thesis. This architecture encompasses, from bottom to top, the Perception layer, Networking layer, Middleware layer, and Application layer.

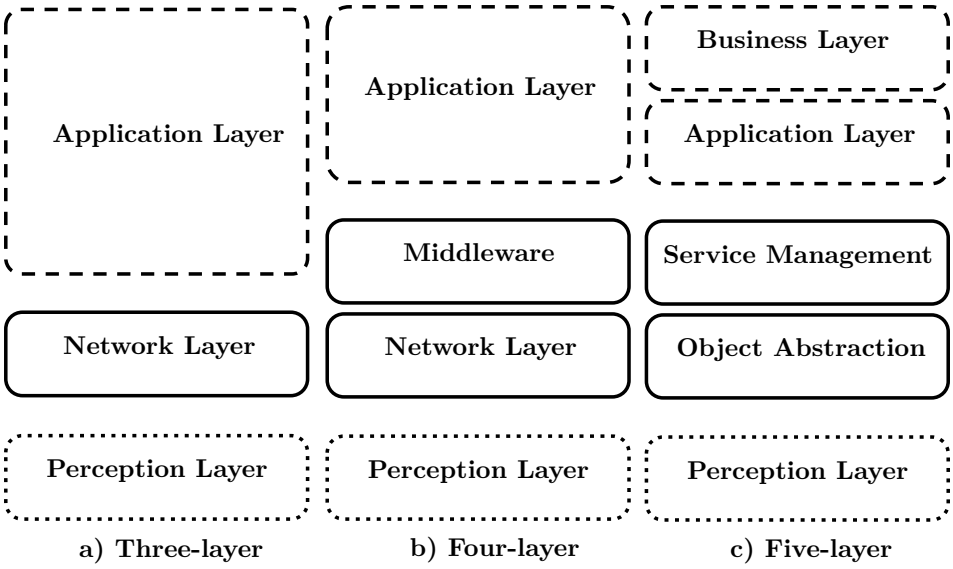

Figure 2.1: IoT Architectures

At the base lies the Perception Layer, which consists of smart sensors and actuators collecting and processing environmental information to perform functions, such as querying temperature,

location, motion, and acceleration, while actuators facilitate actions based on the received information. At the Perception layer, most devices are equipped with 8-bit or 16-bit microcontrollers with very little RAM and storage capacities and can connect to the Internet either via ethernet or low-powered wireless communications such as IEEE 802.15.4. Table 2.1 illustrates the main technical details of some Wireless Sensor Nodes [22].

Above the Perception Layer, the Network Layer takes charge of transmitting this gathered data to the subsequent layer for processing and analysis. This involves the utilization of diverse communication protocols such as Wi-Fi, LPWAN, Bluetooth, and cellular networks, ensuring secure and efficient data transmission. The Middleware pairs a service with its requester based on addresses and names. This layer enables the IoT application programmers to work with heterogeneous objects without consideration of a specific hardware platform. Also, this layer processes received data, makes decisions, and delivers the required services over the network protocols. On the uppermost layer, the Application Layer is the interface through which end-users interact with the IoT system. It encompasses the applications, services, and interfaces that enable users to access and utilize the data generated by IoT devices.

The selection of this four-layer reference architecture offers the most finely-grained model among the available candidates and enables a meticulous and nuanced analysis of privacy protection across different layers. Numerous extant architectures [23]–[25] incorporate all four layers, either as distinct entities or as integrated components. This approach mitigates the potential ambiguity and lack of differentiation that may arise when layers are not distinct [26].

Table 2.1: Wireless Sensor Node Platforms. Source [22].

	MICAz	Tmote Sky	Imote2
Microcontroller	ATmega128L	MSP430	PXA271
Word size	8-bit	16-bit	32-bit
Memory	128KB PF 516KB MF 4KB EEPROM	10KB RAM 48KB FLASH 1024kB External FLASH	256KB SRAM 32MB SDRAM 32MB FLASH
Power	3.0 V	3.0 V	3.85 V
Current Draw	8 mA	1.8 mA	66 mA
Radio	CC2420	CC2420	CC2420
Data Rate	250 kbps	250 kbps	250 kbps

PF: Program FLASH, **MF:** Measurement FLASH,

2.2.2 Requirements of the IoT Architecture

IoT technology has the potential to facilitate numerous business opportunities, however, there are some requirements for future technologies that are challenging. The following are brief introductions to these requirements [27]:

Energy Awareness: Most IoT-enabled smart devices can sense, receive, operate, and process information continuously to facilitate intelligent decisions [28]. Energy plays an important role in IoT resources due to the processing and transmission of a large amount of information with limited power and energy-constrained devices. Hence, energy consumption is one of the prominent requirements to improve the network and lifetime of the IoT nodes.

Quality of Services (QoS): This is one of the important criteria that establish better services for the providers and the users. In this architecture, highlighting services and information

retrieval plays an important role in the field of medical and industrial applications. Hence, improving the QoS is one of the design requirements to improve the overall performance in several fields [29].

Massive Data Management and Control: In order to ensure robust communication, the smart devices in the IoT network should be accessed and organized remotely. Thus, the resources that are connected should share the information and be updated inside the network in real-time [30]. Also, the load processed in this system should balance properly to ensure the proper communication between the user and the IoT-enabled devices. On the other hand, the volume of data generated by IoT devices can be enormous and difficult to manage in terms of elaboration, communication/transmission, and storage. Scalable infrastructures are necessary to efficiently handle this massive growing volume of data [31].

Interoperability: This is one of the essential needs for the development of IoT that has the ability of the systems or devices to communicate with each other without the consideration of the technical or manufacturer specifications. Hence, the future as well as the current IoT-enabled devices should be able to adapt and establish a connection with various wireless technologies to make the IoT system more diverse [32]. The landscape of standards for the IoT is full of open solutions, backed by a diverse array of independent and multinational governing bodies, alliances, and organizations. These standards encompass various facets of IoT products, services, and systems, ranging from communication technologies to architectural frameworks. While some standards adopt a neutral, cross-domain approach, others are tailored for specific vertical domains. Regrettably, the unregulated proliferation of standards, compounded by the absence of universally accepted norms, has resulted in fragmentation. This issue can potentially impede the widespread adoption of IoT and hinder seamless integration across multiple application domains [33].

Lack of Skills: The complexity and the heterogeneity of the technologies involved in an IoT domain require specific skills for the design, and implementation, but also for the operations of the deployed solutions. Acquiring such proficiency proves to be a challenging endeavor for organizations. In this context, the IoT ecosystem assumes a pivotal role, as it has the potential to ensure that the requisite skills are made available and acquired in an efficient and effective manner [34].

Privacy: Privacy concerns arise from the substantial volume of data produced by IoT devices, potentially revealing sensible details about the living context and habits of device owners/users. This data collection may occur without explicit user consent and, when shared by IoT platforms, could be exposed to third parties, thereby stripping users of control over which data is accessible and by whom [35]. Although administrative policies are in place to safeguard the privacy of IoT users, the imperative lies in devising solutions that guarantee privacy through a design-centric approach.

Security: Cybersecurity is the main barrier to IoT, unlike conventional web security, IoT security introduces new factors and conditions that increase potential threats. Interconnectivity with other devices, the ubiquity and dynamism of devices, and the heterogeneity of networks complicate the security and privacy management of stored and exchanged data. In addition, these devices often have limited computing power, low memory, and are battery-dependent, which prevents the direct adoption of the computationally complex cryptographic protocols that protect traditional networks. Therefore, to effectively address IoT security, novel security models are essential. These models must take into account the security requirements of each application as well as the limitations of IoT devices [36]. In this context, this thesis proposes

solutions to ensure the security and privacy of data stored and exchanged by IoT devices, considering the intrinsic limitations of the devices.

2.3 Overview of IoT Security

This section presents the requirements, challenges, and threats presented by IoT networks.

2.3.1 Security Requirement

In the IoT infrastructure, the information generated is generally transmitted over the Internet. IoT is operated in the open access and is capable of making communication over public networks. Thus, certain requirements need to be satisfied to make the data secure in IoT applications. Those requirements for IoT applications are discussed as follows:

Integrity Integrity focuses on the ability to be certain that the information contained within the message cannot be modified while in storage or transit. This allows data modification to be processed only by the authorized person and neglects the unauthorized access to the network. In the IoT system, the adversaries could get the information exchanged by two devices and establish a new connection, thereby corrupting the information by the third user.

Confidentiality: Confidentiality ensures that only the intended recipient can decrypt the message and read its contents. A critical facet of secure communication addresses several critical issues. Firstly, it restricts information transmission exclusively to authorized users. Additionally, it mandates the encryption of sensor identities and the public keys of users. Moreover, it emphasizes the enhancement of key exchange mechanisms. Through these measures, only authorized users gain access to the data, thereby preventing unauthorized entry and deterring illicit information exchanges. When confidentiality is reinforced through authentication, cryptographic and encryption systems are employed to safeguard the integrity of the information.

Availability Availability is a crucial aspect that demands both information and sensor devices remain accessible to authorized parties. Due to various threats in wireless communication, instances may arise where functionality and services are disrupted or inaccessible in certain parts of the network. Ensuring the availability of services is imperative, even in the event of node disruptions, thereby ensuring uninterrupted system functionality.

Access Control It deals with access and identity management that controls how the nodes can interact with other nodes in the network. It allows only authorized (authenticated) users to access the data.

2.3.2 Security Challenges

The integration of resource-constrained networks with the robustness of the Internet poses a considerable challenge, given the resulting heterogeneity of both networks. This complexity complicates the design of protocols and the operation of systems. IoT deployments are often characterized by communication channels with limited bandwidth and prone to loss.

As stated earlier, IoT devices frequently operate under constraints of CPU processing power, available memory, and energy resources. These characteristics exert a direct influence on the design of protocols tailored to the IoT domain. For example, the imposition of small packet-size limits at the physical layer (e.g., 127 Bytes in IEEE 802.15.4) can lead to either hop-by-hop fragmentation and reassembly, or a reduction in the maximum transmission unit (MTU) at the IP layer. The former may introduce potential vulnerabilities to state-exhaustion attacks

due to excessive fragmentation of large packets often necessitated by security protocols. The latter scenario may result in increased fragmentation at the IP layer, a phenomenon commonly associated with heightened packet loss and the subsequent need for retransmission.

To optimize memory usage and bandwidth, it is imperative to minimize the size and number of messages. In this context, layered approaches involving multiple protocols may inadvertently lead to suboptimal performance in resource-constrained devices, as they aggregate the headers of distinct protocols. Moreover, in certain settings, protocol negotiation may augment the volume of exchanged messages. To enhance performance in fundamental procedures like bootstrapping, it may be judicious to execute these procedures at a lower layer.

This is especially pronounced when the basic cryptographic components necessitate frequent use, or when the underlying application requires low latency.

While ongoing efforts aim to mitigate the resource consumption of security protocols by leveraging more efficient cryptographic primitives such as Elliptic Curve Cryptography (ECC), these advancements represent only an initial step in reducing the computational and communication overhead of Internet protocols. The question remains whether alternative approaches can be employed to optimize key agreement in these heavily resource-constrained environments.

An additional critical consideration pertains to the limited energy budget available to IoT nodes. Careful protocol design and usage are imperative to curtail energy consumption during normal operation and mitigate energy depletion under Denial-of-Service (DoS) attacks. Given that the energy consumption of IoT devices deviates from that of other device classes, assessments of protocol energy consumption must be tailored to specific IoT implementations.

The stringent memory and processing constraints inherent to IoT devices inherently mitigate resource-exhaustion attacks. This is particularly true in unattended Thing-to-Thing (T2T) communication, where such attacks are challenging to detect prior to the service becoming unavailable (e.g., due to battery or memory depletion). To counteract DoS attacks, protocols such as DTLS, IKEv2, HIP, and Diet HIP implement return routability checks through a cookie mechanism. This mechanism delays the establishment of a state at the responding host until the address of the initiating host is verified. The effectiveness of these defenses is contingent on the routing topology of the network. Return routability checks prove particularly efficacious when hosts are incapable of receiving packets addressed to other hosts, and when IP addresses convey meaningful information, as is the case in the present Internet landscape. However, their efficacy diminishes in broadcast media or in scenarios where attackers can influence the routing and addressing of hosts (e.g.; if hosts contribute to the routing infrastructure in ad hoc networks and meshes).

2.3.3 Security Attacks in IoT

In the design of an IoT security protocol, in addition to considering the challenges mentioned above, it must withstand possible attacks by an adversary that directly affect the security requirements addressed in Section 2.3.1. Attackers are rapidly adapting to changes and finding more ways to exploit vulnerabilities that exist in IoT devices and architecture. The security attacks reported in IoT can be broadly classified into four categories: Physical Attacks, Network Attacks, Software Attacks, and Data Attacks as depicted in Fig. 2.2. In this section, we give a detailed overview of Physical and Network Attacks attacks along with a literature review of the countermeasures adopted to deal with each of these attacks. A comprehensive survey of all category attacks can be found in [37].

Physical attacks

Physical attacks can be launched if the attacker remains physically close to the network or devices of the system [37]. The common forms of physical attacks are listed below:

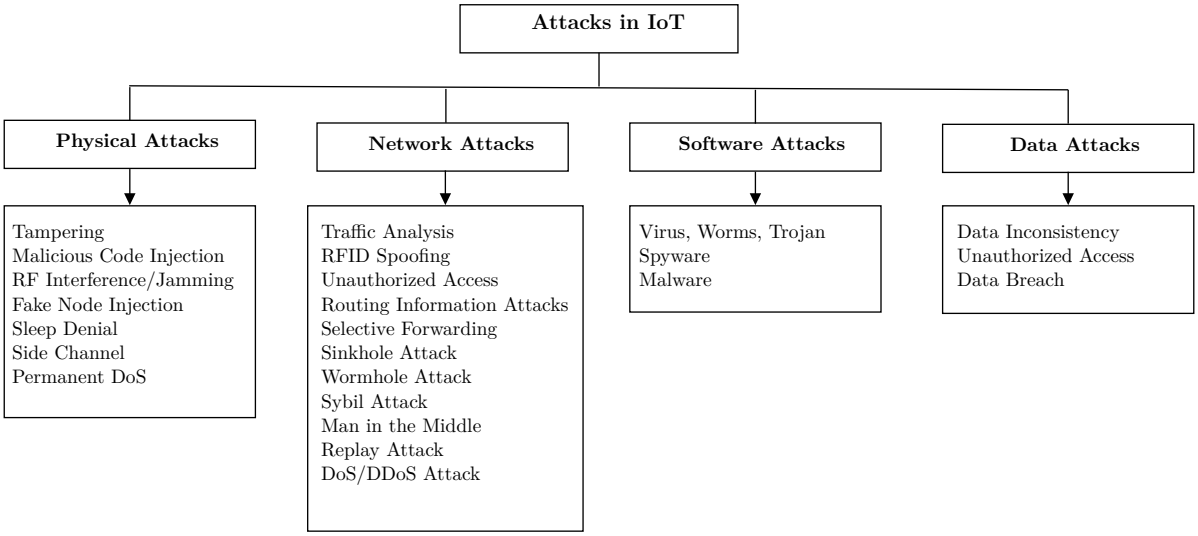

Figure 2.2: Attacks in IoT

- Tampering: Refers to the act of physically modifying a device (e.g. RFID) or communication link.
- Malicious Code Injection: Here the attacker injects malicious code onto a physical device by compromising it which may help him/her launch other attacks too.
- RF Interference/Jamming: The attacker creates and sends noise signals over the Radio Frequency (RF)/WSN signals to launch DoS attacks on the RFID tags/sensor nodes thereby hindering communication.
- Fake Node Injection: The attacker drops a fake node between two legitimate network nodes to control data flow between them.
- Sleep Denial Attack: The attacker keeps the battery-powered devices awake by feeding them with the wrong inputs. This causes exhaustion in their batteries leading to shutdown.
- Side Channel Attack: In this attack, the attacker collects the encryption keys by applying timing, power, or fault attack on the devices of the system. With the help of these keys, it can encrypt/decrypt confidential data.
- Permanent Denial of Service (PDoS): Also known as *phlashing*, is a type of DoS attack, wherein an IoT device is completely damaged via hardware sabotage. The attack is launched by destroying firmware or uploading a corrupted BIOS using malware.

Many solutions have so far been reported for mitigating the above attacks on IoT devices. Work proposed in [38] focuses on the development of a mutual authentication protocol for small-sized IoT devices. This protocol leverages Physically Unclonable Functions (PUF) and exploits the inherent variability within Integrated Circuit (IC) structures. Authentication is achieved through a challenge-response mechanism that relies on the physical microstructure of the device, making it extremely challenging to replicate, thereby mitigating threats like tampering and malicious code injection.

Gomes et al. in [39] have proposed a comprehensive solution at the architectural level to enhance energy efficiency while bolstering security capabilities in IoT devices. Their approach involves a heterogeneous architecture implemented on a customizable and trustable end device mote (CUTE mote). This architecture combines a reconfigurable computing unit (RCU) with

an IEEE 802.15.4 radio transceiver and a hardcore micro-controller unit (MCU) hosting Contiki-OS. Experimental results indicate that this architecture, when integrated with the Smart-Fusion 2 SoC hardware platform, is proficient in defending against physical attacks such as jamming, thus fortifying IoT device security.

In the realm of distributed IoT applications, ensuring secure connections between peer sensor nodes and end users is paramount. Porambage et al. in [40] have introduced the pervasive authentication protocol (PAuthKey) tailored for Wireless Sensor Networks (WSNs). This protocol involves the acquisition of implicit certificates from the Cluster Head (CH), enabling the establishment of secure links between peer sensor nodes and end users. The authentication scheme relies on the relative positioning of sensor nodes, effectively safeguarding against node compromise, masquerade, impersonation attacks, and fake node injection.

Additionally, proactive measures have been introduced to mitigate threats such as sleep denial attacks and side-channel attacks. In [39] heterogeneous architecture makes sleep denial attacks infeasible, while the proposed solution in [41] has developed a Support Vector Machine (SVM) classification algorithm to detect resource depletion, a primary trigger for sleep denial attacks. Moreover, PUF-based authentication, introduced in [38] offers inherent resistance to side-channel attacks, as its physical microstructure and variability make forging practically impossible.

In the context of Permanent Denial of Service (PDoS) attacks, which can have severe implications for Industrial Control Systems (ICS), Sicari et al. have introduced REATO in [42]. This solution addresses various DoS attacks, including data-focused attacks, within an IoT environment. REATO is based on a cross-domain and flexible middleware called NetwOrked Smart object (NOS). It employs a validation mechanism involving HTTP connection requests to NOS, with encrypted information transmitted upon successful validation. Experimental implementation confirms the efficacy of this technique in identifying and mitigating various DoS attacks in IoT systems.

Network Attacks

Network attacks are performed by manipulating the IoT network systems to cause damage. It can easily be launched without being close to the network. The most common forms of network attacks are summarized below:

- Traffic Analysis Attack: Confidential information or other data flowing to and from the devices are sniffed by the attacker, even without going close to the network in order to gain network information.
- Routing Information Attacks: These are direct attacks where the the attacker spoofs or alters routing information and creates a nuisance by activities like creating routing loops or sending error messages.

  Selective Forwarding: In this attack, a malicious node may simply alter, drop, or selectively forward some messages to other nodes in the network. Therefore, the information that reaches the destination is incomplete.
- Sinkhole Attack: In this attack, an attacker compromises a node closer to the sink (known as sinkhole node) and makes it look attractive to other nodes in the network thereby luring network traffic towards it.
- Wormhole Attack: In a wormhole attack, an attacker maliciously prepares a low-latency link and then tunnels packets from one point to another through this link.
- Sybil Attack: a single malicious node claims multiple identities (known as Sybil nodes) and locates itself at different places in the network. This leads to unfair resource allocation.

- Man in the Middle Attack (MiTM): an attacker manages to eavesdrop or monitor the communication between two IoT devices and access their private data.
- Replay Attack: An attacker may capture a signed packet and resend the packet multiple times to the destination. This keeps the network busy leading to a DoS attack.
- Denial/Distributed Denial of Service (DoS/DDoS) Attacks: Unlike DoS attack, in DDoS, multiple compromised nodes attack a specific target by flooding messages, or connection requests to slow down or even crash the system server/network resource

Many approaches have been proposed to face those attacks on the Network layer. In the work by Liu et al. [43], an efficient and privacy-preserving traffic obfuscation (EPIC) framework is proposed to safeguard smart homes against traffic analysis. This framework incorporates a secure multihop routing protocol that ensures strong differential privacy by guaranteeing the unlinkability of traffic flow to specific smart homes, as well as between source and destination.

Guin et al. have developed a Physically Unclonable Function (PUF) based on on-board SRAM in [44], which generates a unique device footprint as the device ID. This ID-matching technique reduces the probability of impersonation by an adversary, thereby mitigating the risk of spoofing and unauthorized access.

In [45] is introduced the Secure Routing Protocol for Low power and Lossy Networks (SRPL), employing hash chain authentication along with the concept of rank threshold. SRPL mandates authentication based on hash values to prevent malicious nodes from exploiting control messages, effectively countering routing attacks. This authentication technique, combined with rank threshold, is also effective against selective forwarding and sinkhole attacks.

To address forwarding misbehavior, the solution in [46] proposes a monitor-based approach named CMD, utilizing RPL as the routing protocol. CMD enables each node to monitor the packet loss rate of the preferred parent compared to that of its one-hop neighbors, facilitating the detection of forwarding misbehaviors. Additionally, Cervantes et al. in [47], introduce an intrusion detection system tailored for Sinkhole attacks over 6LoWPAN for the Internet of Things (INTI). This system employs reputation, watchdog, and trust strategies to detect potential attackers by analyzing the behavior of each node in the network, ultimately revealing the identity of the attacking node and isolating the detected sinkhole.

To counteract Man-in-the-Middle (MitM) attacks, in [48] is proposed a secure MQTT and MQTT-SN protocols, both employing Attribute-Based Encryption (ABE) with Elliptic Curve Cryptography (ECC) to ensure secure device-to-device communication. Additionally, Authors in [49] address MitM attacks by authenticating inter-device communication, where each sensor participates in the generation and distribution of session keys. This scheme, utilizing a decentralized approach for key generation, significantly enhances performance for resource-constrained IoT devices.

In [50] is introduced SecTrust-RPL, a trust-aware RPL routing protocol designed to detect and isolate nodes launching Sybil attacks. The SecTrust framework embedded in ContikiRPL serves as the trust engine for routing decisions and malicious node detection, relying solely on trust among nodes to quickly isolate detected malicious nodes from the network. On the other hand, in [51], authors present a signcryption technique based on Identity Based Cryptography (IBC) that simultaneously ensures confidentiality, integrity, and authentication. This technique effectively integrates encryption and signature schemes, eliminating the need for access to a trusted third party, thus bolstering resilience against replay attacks.

Authors in [52] propose a defensive framework against network Denial of Service (DoS) and Distributed DoS (DDoS) attacks, focusing on message flooding. Their algorithm employs an EDoS server to analyze incoming traffic and categorize it as suspicious, effectively distinguishing between DoS and DDoS attacks. In a separate work [53], a Software Defined Internet of Things (SD-IoT) framework is introduced, employing the SDx paradigm to detect and mitigate DDoS

attacks using cosine similarity vectors. The framework determines the occurrence of a DDoS attack by comparing threshold values, subsequently identifying and blocking the attacker at the source.

As a result of this analysis, we can conclude that there are no solutions that address all security issues in every scenario. Moreover, the majority of the solutions proposed in the literature employ cryptographic techniques to mitigate various attacks in both the Physical and Network layers. Given the constraints of IoT devices, it is imperative to carefully select and evaluate the cryptographic techniques that can be applied to them.

2.4 Cryptographic Primitives Overview

Cryptography is the study of mathematical techniques related to aspects of information security such as confidentiality, data integrity, entity authentication, and data origin authentication [54]. Cryptographic goals are confidentiality, authentication, data integrity, and non-repudiation. Confidentiality refers to the protection of sensitive information from unauthorized access or disclosure. It involves the use of encryption techniques to render data unreadable to anyone who does not possess the appropriate decryption key. This ensures that even if an attacker intercepts the encrypted data, they cannot decipher its content without the proper cryptographic key, thereby maintaining the confidentiality of the information.

Authentication is the process of verifying the identity of a party involved in a communication or transaction. In cryptography, it is essential to confirm that the sender or recipient of a message is who they claim to be. This is typically achieved through the use of digital signatures, certificates, or Message Authentication Codes (MAC). Authentication ensures that messages or transactions are not manipulated by malicious parties, enhancing the trustworthiness of cryptographic communications.

Data integrity in cryptography guarantees that the data remains unaltered during transmission or storage. It involves the use of cryptographic techniques such as hash functions to generate unique checksums or hashes for data. These hashes act as digital fingerprints, and any modification to the data would result in a different hash value. By comparing the received hash with the original one, recipients can verify the integrity of the data and detect any unauthorized alterations.

Non-repudiation is a cryptographic concept that prevents individuals from denying their involvement in a communication. It provides proof of the origin of a message or transaction and ensures that the sender cannot later deny their participation. Digital signatures are commonly used to achieve non-repudiation, as they bind the identity of the sender to the message and confirm the message's authenticity.

To meet these criteria, several cryptographic techniques are often used together according to the requirements of the application. Cryptographic techniques are typically divided into two generic types: Symmetric-key Cryptography (SKC) and Asymmetric-key or public-key Cryptography (PKC). Figure 2.3 provides a schematic listing of the cryptographic primitives considered and how they relate. This chapter provides an in-depth exploration of certain cryptographic primitives. For a comprehensive understanding, a more detailed explanation is presented in [22].

2.4.1 Public-key vs Symmetric-Key Cryptography

Symmetric-key cryptography also referred to as shared-key, single-key, or secret-key cryptography, relies on a singular secret key that is initially shared between the sender and receiver. This shared key facilitates the encryption and decryption of messages exchanged between them. The process of predistributing this key poses significant complexity. It is important to note that symmetric-key cryptography cannot establish non-repudiation, as both the sender and receiver

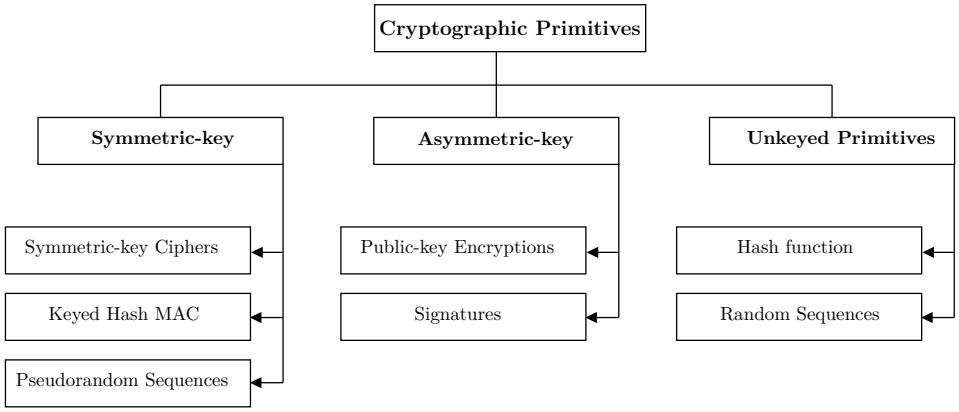

Figure 2.3: Taxonomy of Cryptographic Primitives

utilize the same key, as both sender and receiver use the same key, messages cannot be verified to have come from a particular user.

In contrast, asymmetric cryptography, or public-key cryptography, operates with a pair of keys: a private, confidential key and a public key openly available. Operations conducted with the private key can only be reversed using the corresponding public key, and vice versa. This nice property makes all PKC-based algorithms useful for secure broadcasting and authentication purposes. It is also an invaluable tool for allowing the secure exchange of secret keys between previously unknown partners.

The computational overhead associated with Public-Key Cryptography (PKC) has posed challenges to its implementation in resource-constrained devices. Conversely, SKC offers benefits in terms of reduced communication and computational costs. It may be inferred that SKC is better suited for IoT applications where confidentiality or data integrity is the primary concern. In order to implement SKC in this context, a mechanism for distributing shared keys is essential. Key predistribution methods can be categorized into three types:

- A single network-wise secret key: this causes a single- point failure, *i.e.,*if the secret key of a node is revealed then the entire network is broken.
- A pairwise key between two nodes: the pairwise keying is very difficult and inefficient, *i.e.,* each node must share $\frac{n(n-1))}{2}$ secret keys, where n is the number of the nodes. This creates a problem with managing and ensuring the security of all these keys. If the secret key of a node is revealed, then the other node with the same key is also compromised.
- A group key among a set of nodes: group keying is more inefficient than pairwise keying as it requires heavy computational overhead and interactions with more than two rounds among nodes. If the group key of a node in a group is revealed, then all the group of nodes is compromised.

Minimizing the effects of secret key exposure is an important factor. In fact, the security schemes should guarantee that no matter how many nodes are captured, the secret information extracted from the compromised nodes cannot affect the security among non-compromised nodes, *i.e.,* communications among non-compromised nodes remain secure.

Given the constraints commonly associated with IoT devices, it becomes imperative to employ cryptographic solutions that either offer non-repudiation in the case of symmetric cryptography, or utilize computationally efficient, low-power, and storage-friendly options in asymmetric solutions. This thesis introduces two security solutions aligning with these principles. For the asymmetric approach, a low computational cost is attained through the application of Elliptic Curve Cryptography and the utilization of implicit certificates. Conversely, the symmetric solution employs MAC codes to establish authentication.

2.4.2 Elliptic Curve Cryptography

Elliptic Curve Cryptography (ECC) is an efficient PKC scheme with the most suitable adaptations for low-performing resource-constrained networking devices [55]. Compared to other expensive PKC schemes, like the well-known Rivest Shamir Adleman (RSA) public-key algorithm, ECC has faster computational time, smaller keys, and less memory and bandwidth utilization. This is explained in Table 2.2, where ECC can obtain a similar security level as RSA using much smaller keys.

While designing ECC algorithms, the Elliptic Curves (ECs) are defined over a finite field by an equation using two variables with coefficients, which are the elements of the finite field [56]. Consequently, all the variables, coefficients and curve points fall below the same finite abelian group, $\mathbb{G}$. The resultant points of the curve operations are also restricted in the same abelian group. A special point $\mathcal{O}$, known as the zero element or point of infinity, is considered the identity element of the group. ECC is formulated with EC point addition, point scalar multiplication, and, additive and multiplicative inverses on ECs over prime integer fields or binary polynomial fields. Modulo arithmetic is the foundation for all the EC point operations. The implementation of ECC on constrained devices is performed over prime integer fields since binary polynomial field operations are too costly for low-power devices.

ECs are defined over prime fields Z_p, where p is a large prime number. The variables and coefficients will have values between 0 and $p-1$ and calculations are performed in modulo p. Let $a, b \in Z_p$ and $4a^3 + 27b^2 \not\equiv 0 \ mod \ p$. Then the EC is defined in Eq. 2.1.

$$E : y^2 \equiv x^3 + ax + b \ mod \ p \tag{2.1}$$

Once p, a, and b are selected, a group of EC points $E_p(a, b)$ are defined so they satisfy Eq. 2.1. Then a base point generator $\mathbb{G} = (x_1, y_1)$ is chosen so that the order of $\mathbb{G}$ is a very large value n and $n \times \mathbb{G} = \mathcal{O}$. The key building block of ECC is the scalar point multiplication which is $Q = k \times P$, where k is a positive integer and P and Q are points in the EC. The value $k \times P$ is computed by adding point P for $k-1$ times and the resulting point Q is obtained. However, the recovery of k, knowing the points P and Q is a hard or computationally infeasible problem which is known as the Elliptic Curve Discrete Logarithmic Problem (ECDLP). In real-time applications, k is made large in order to overcome guessing and brute force attacks.

Table 2.2: Security and Key-length Comparison. Source [57].

Security level	**Public key size (bits)**		**Certificate size (bits)**		
	ECC	RSA	ECQV	ECDSA	RSA
80	160	1024	193	577	2048
112	224	2048	225	673	4096
128	256	3072	257	769	6144
192	384	7680	385	1153	15360
256	512	15360	522	1564	30720

ECC for Authentication and Key Management

With the heterogeneous devices and distributed nature, the authentication protocols in IoT should not only be resistant to malicious attacks, but they should also be lightweight to enable deployment in less-performing IoT devices. Authentication constitutes a pivotal facet within key establishment protocols, which can be categorized into symmetric and asymmetric techniques.

Under asymmetric techniques, there are four variants such as static public key authentication, certificate-based authentication, cryptographically generated identifiers, and identity-based authentication. In every case, a node proves its identity by providing proof of knowledge of the corresponding private key. In the first two categories, the authentication is implicitly ensured by the ownership of corresponding public-private keys or certificates. In the third category, the authentication identifiers are generated using the public key of the node. In the last asymmetric technique, opposite to the previous category, a node's public key is derived from its identity. ECC-based implicit certificates and their utilization for authentication and key establishment in resource-constrained IoT devices

ECC Based Implicit Certificates

Digital certificates advocate the establishment of identity in secure communications. Similar to the conventional or explicit certificates such as X.509, implicit certificates are made up of three parts [57]: identification data, a public key, and a digital signature, which binds the public key to the user's identification data and verifies that the binding is accepted by a trusted third-party. In an explicit certificate, the public key and digital signature are two distinct elements. In contrast, the public key and digital signature are included in implicit certificates and allow the recipient to extract and verify the public key of the other party from the signature segment. This will significantly reduce the required bandwidth since there is no need to transmit both the certificate and the verification key.

The most important advantages of using implicit certificates over conventional certificates are the smaller size and faster processing. Table 2.2 specifies the comparable key sizes for symmetric and asymmetric cryptosystems based on equivalent security strengths, *i.e.,* symmetric key size. Elliptic Curve Digital Signature Algorithm (ECDSA) is a variant of the Digital Signature Algorithm (DSA) that operates in elliptic curve groups. Elliptic Curve Qu-Vanstone (ECQV) [58] is another type of implicit certificate scheme with smaller certificate sizes, lower computational power, and very fast processing time for generating certified public keys. Accordingly, the sizes of ECQV and ECDSA-signed certificates are substantially smaller than RSA due to the reduced public key size of ECC.

Implicit certificates based on Elliptic Curve Cryptography (ECC) have been leveraged to devise lightweight and secure protocols for key establishment and authentication within resource-constrained sensor networks. In [59], the authors introduced a two-phase protocol for implicit certificate-based key establishment tailored to resource-limited sensors deployed in generic Wireless Sensor Networks (WSNs). In the initial phase, sensor nodes acquire implicit certificates from the cluster head, which serves as the Certificate Authority (CA). The certificate generation process draws inspiration from the design principles underpinning the ECQV implicit certificate scheme. The subsequent phase encompasses the key establishment module, wherein sensors utilize implicit certificates to forge pairwise keys with proximate sensor nodes. The theoretical framework supporting this protocol aims to strike a balance between computational efficiency and security in resource-constrained environments. The concepts behind this work are extended in [59] by using implicit certificates for authenticating devices and users under the umbrella of IoT. Using the schemes presented in [60] and [59], a pervasive authentication and key establishment scheme are designed for IoT networks in [40].

Given the described benefits of ECC utilization and the use of implicit certificates, both approaches are employed in the solution for group authentication introduced in Chapter 4. As evident, a majority of security solutions necessitate the involvement of a trusted third party (TTP) for tasks related to identity and key management. This centralized approach introduces a potential single point of failure in the system, *i.e.,* TTP becomes susceptible to DoS attacks or technical malfunctions. The subsequent section will delve into the discussion of Blockchain technology, which holds promise as a viable solution to address the centralization challenge in IoT.

2.5 Blockchain Technology Overview

Blockchain-based systems represent an amalgamation of cryptographic techniques, public key infrastructure, and distributed models, applied within the context of peer-to-peer(P2P) networking and decentralized consensus mechanisms in order to achieve synchronization of distributed databases. Fundamentally, the Blockchain functions as a distributed data structure and is commonly referred to as a "distributed ledger" due to its capacity to record transactions taking place within a network. While cryptocurrencies exemplify one application of the ledger functionality inherent in Blockchains, the distributed ledger holds potential for broader application in networks where various forms of data exchange occur. In a peer-to-peer Blockchain-based network, all participating peers uphold identical replicas of the ledger. Subsequent entries, encapsulating transaction-related information, are appended to the Blockchain through a process of decentralized consensus among the peers.

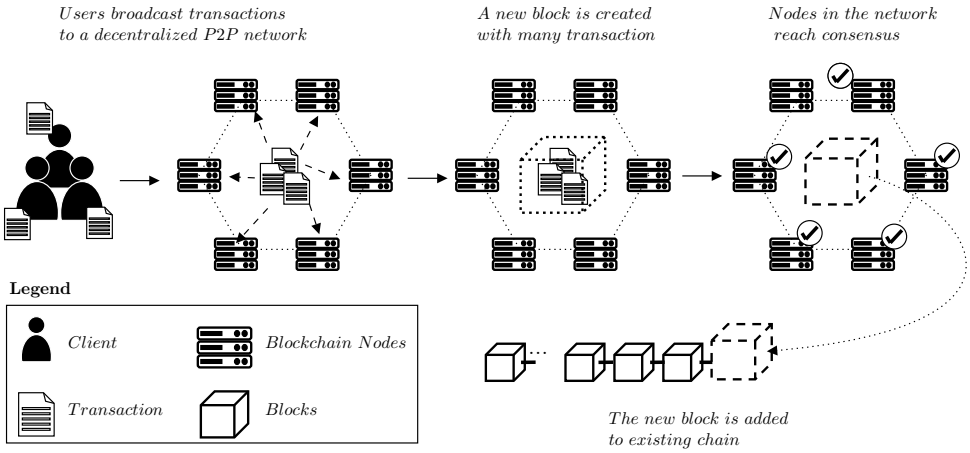

Figure 2.4: Transactions life-cycle

Figure 2.4 shows the life-cycle of data transactions to be included in a Blockchain. Blockchain operates through a decentralized P2P network where users initiate transactions. These transactions are broadcast to the network. Subsequently, Blockchain nodes collect a batch of these transactions and create a new block. This block undergoes thorough validation to ensure compliance with the network's predefined rules. Once the block is confirmed legitimate, it is broadcast to all nodes in the network. Nodes collaborate to verify the authenticity of the block, confirming the validity of transactions and adherence to network protocols. Upon reaching a consensus, the new block is appended to the existing Blockchain.

Each block is logically partitioned into two distinct components: the header and the body, as shown in Fig. 2.5. The information of the transactions is stored within the body, while the header encompasses, among other fields, the identifier of the preceding block, a timestamp, and a Merkle root computed over all the transactions. Consequently, the blocks are interlinked in a chain akin to a linked list. The initial block in this sequence is denoted as the "genesis" block.

The identifier of each block is derived from its cryptographic hash, substantiating the role of inter-block connections in preserving the immutability of the Blockchain contents. Any attempt by a hacker to modify the contents of a prior block would render its identifier invalid, triggering a cascade effect that invalidates the parent block hashes in subsequent blocks. Therefore, to successfully alter the content of a single block, an attacker would need to revise the headers in all ensuing blocks and ensure this modification is accepted by the majority of nodes in the network, thereby achieving consensus on this altered Blockchain.

In addition to the block identifier and the identifier of the preceding block, the header encompasses a timestamp indicating when the block was published, as well as the Merkle tree root representing all transactions stored within the body of the block. The Merkle tree root

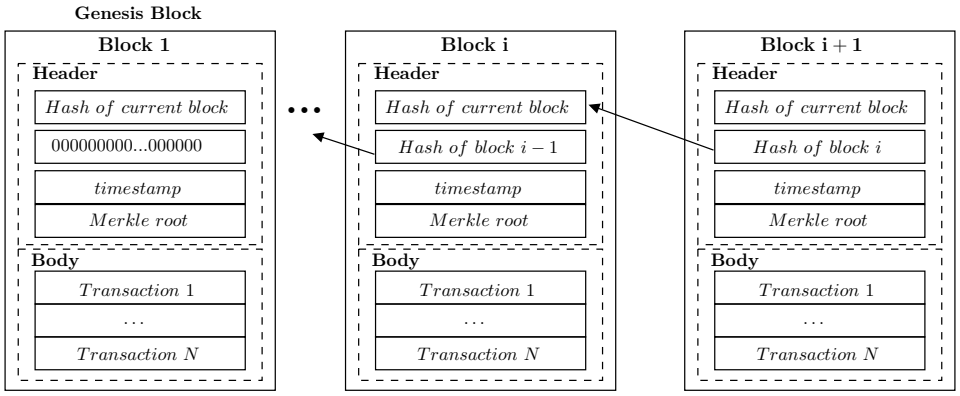

Figure 2.5: Blockchain data structure

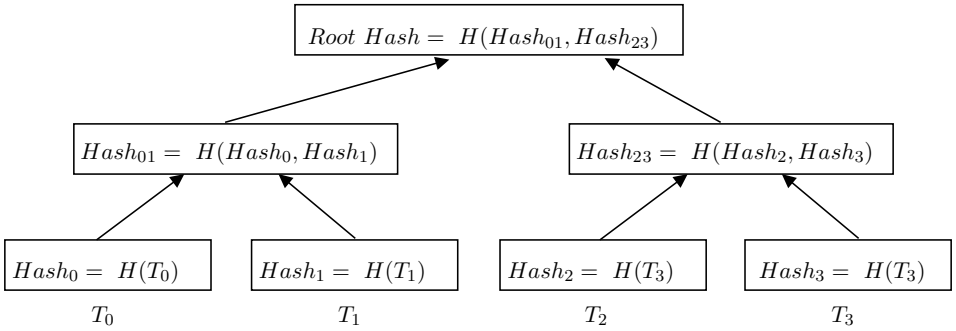

Figure 2.6: Merkle root example

significantly streamlines the process of verifying transactions within a block. To elaborate, the Blockchain constitutes a linearly expanding data structure, with heightened transactional activity inflating the sizes of newer blocks. As an integral facet of all consensus protocols, peers authenticate transactions recorded in a newly released block. Each transaction within a block possesses a unique transaction ID, derived from the cryptographic hash of the pertinent transactional information stored in the block. These transaction IDs are amalgamated in pairs, forming a hash tree within the block, as depicted in Fig. 2.6. The root of this tree is consequently stored in the block header. Consequently, to authenticate a transaction, a local repository of all transactions is unnecessary, as verification can be conducted through the utilization of the Merkle tree branch containing the relevant transaction. Any tampered transaction would yield modified hashes within its branch, swiftly detected without significant computational expenditure.

In the event of multiple nodes in the Blockchain network concurrently generating valid blocks, the Blockchain can bifurcate, this phenomenon is known as a fork, presenting challenges in maintaining a singular canonical version. Established Blockchain networks mitigate this issue by recognizing the longest fork as authoritative, with all blocks published in alternate forks being either discarded or designated as orphaned. Additional fields integrated into the block header encompass information specific to the consensus algorithm employed within the Blockchain network.

Smart Contracts

Smart contracts represent programmable applications residing within the Blockchain, designed to oversee transactions based on predefined terms and conditions. Consequently, they serve as the digital counterparts to conventional economic contracts entered into by various participating entities. In contrast to conventional contracts, which rely on centralized authorizing bodies

for enforcement, a Blockchain network operates without the need for intermediaries to verify compliance with the stipulations delineated within a Smart Contract [61], [62]. The term "Smart Contract" was introduced by N. Szabo with the aim of "establishing secure relationships on public networks" [63]. In Blockchain networks, Smart Contracts are responsible for executing transactions according to predefined rules agreed upon by participating parties. Once deployed, the code of a Smart Contract is permanently stored within the Blockchain. The functions written in the Smart Contract can be triggered by any participant at any time. A Smart Contract is often referred to as an "autonomous agent" because it maintains its own accounts on the Blockchain, complete with unique Blockchain addresses [64]. Consequently, the contract can retain custody or ownership of tokenized assets while the engaging parties work towards meeting the agreed-upon conditions.

Smart contracts play a versatile role within a Blockchain network. They facilitate "multi-signature" transactions, ensuring execution only when a majority or specified percentage of participants provide their agreement [65]. Additionally, Smart Contracts enable automated transactions triggered by predefined events, whether through fixed time intervals or in response to other transactions. This functionality supports decentralized data access and request-response transactions, with the added capability of triggering upon receipt of a message to the contract's address [66]. Moreover, Smart Contracts offer dedicated storage space for application-specific information, ranging from membership records to lists and boolean states.

2.5.1 Blockchain Properties

The architecture and data structure of Blockchain technology gives it the following inherent properties that make it attractive for numerous applications:

Decentralization: In contrast to centralized network infrastructures, wherein data exchanges, *i.e.,* transactions, undergo validation and authorization by trusted central third-party entities, Blockchain-based infrastructures facilitate transactions between two nodes without reliance on the central entity for record maintenance or authorization. This obviates the associated costs of centralized server upkeep and performance bottlenecks.

Immutability: Given that all new entries in the Blockchain undergo consensus agreement by peers in a decentralized manner, the Blockchain attains censorship resistance and becomes exceedingly resistant to tampering. Likewise, all previously recorded data in the Blockchain remains immutable. To alter any preceding records, an adversary would need to compromise a majority of nodes within the Blockchain network. Otherwise, any modifications to the Blockchain contents are readily discernible.

Auditability: Each peer maintains a duplicate of the Blockchain, enabling access to all timestamped transaction records. This transparency empowers peers to scrutinize and authenticate transactions involving specific Blockchain addresses. Notably, Blockchain addresses are not directly linked to real-life identities, affording a form of pseudo-anonymity. While the proprietorship of a Blockchain address cannot be traced, specific addresses can be held accountable, allowing inferences to be drawn regarding the transactions associated with a given address.

Fault Tolerance: All peers within the Blockchain possess identical replicas of ledger records. Any faults or data breaches occurring in the Blockchain network can be identified through decentralized consensus, and data breaches can be rectified by utilizing the replicas stored within Blockchain peers.

2.5.2 Blockchain classification

Blockchain systems fall into two categories based on participant access: Permissioned Blockchains and Permissionless Blockchains [27]. The following is a taxonomy of existing Blockchain implementations, which are compared in Table 2.3.

Table 2.3: Existing Blockchain Taxonomy. Source [27]

	Public	**Private**	**Consortium**
Consensus	All nodes	Single organization	Selected nodes in multiple organizations
Access	Public read/write	Can be restricted	Can be restricted
Identity	Pseudo-anonymus	Approved Validators	Approved Validators
Immutability	Yes	Partial	Partial
Throughput	Slow	Fast	Fast
Permissionless	Yes	No	No

- **Permissioned Blockchain:** In permissioned Blockchain deployments such as private and consortium Blockchains, only a limited number of known participants carry a copy of the entire Blockchain [67]. Maintaining consensus, therefore, is much more straightforward and doesn't require costly proofs for publishing a new block. Since participants are known, there is no risk of a Sybil attack [68], therefore voting mechanisms are used to achieve consensus. By this virtue, Permissioned Blockchains have a much higher performance than Permissionless Blockchains.

  - **Private Blockchain:** Private Blockchain is a type of Blockchain that is restricted by a single authority for any transactions or changes on nodes. There is a central authority controlling access to the functionalities. Private Blockchains are normally partially decentralized due to this reason.
  - **Hybrid Blockchain:** A hybrid Blockchain [69] combines the features of public and private Blockchains. It makes use of both the private permission-based system and the public permission-less system aspects of Blockchains. Users may manage who has access to what data is stored in the Blockchain with the help of such a hybrid network. Only a certain subset of the Blockchain's data or records may be made public, keeping the remainder secret and confidential. Users may simply combine a private Blockchain with many public Blockchains thanks to the flexibility of the hybrid Blockchain technology. A hybrid Blockchain's private network is often used to verify a transaction. However, users can also publish it on the open Blockchain in order to be confirmed. The hashing is increased and additional nodes are used for verification on public Blockchains. As a result, the Blockchain network's security and transparency are improved.
  - **Consortium Blockchain:** A consortium Blockchain is a semi-decentralized kind in which a Blockchain network is managed by more than one entity. This contrasts with what we saw in a private Blockchain, which is administered by a single entity. In this sort of Blockchain, more than one organization can operate as a node, exchanging information or mining. Consortium Blockchains are commonly utilized by banks, government agencies, and other organizations.

- **Permissionless Blockchain:** This is a type of Blockchain in which anonymous participants are termed "permissionless" and can be a member of Blockchain networks. Normally those anonymous users gain their consensus in the permissionless Blockchain using the voting technique. But there is a problem with that where an attacker can create multiple accounts to launch a Sybil attack [68] which can lead to a false representation to drive the outcome toward their favor. Therefore, in permissionless Blockchain implementations, the consensus protocols are based on a lottery-based selection of a single node that publishes a new block onto the Blockchain. To ensure security in public Blockchains where anonymous participants are required to transact in a trustless manner, block creation needs to be expensive so that the resources of one entity are insufficient to bias the consensus decisions in its favor [27].

– **Public Blockchain:** Public Blockchain is a fully decentralized Blockchain where each node or user will have equal rights for performing any functionalities like transactions or data sharing.

2.5.3 Consensus Protocols

In the context of a distributed system, the issue of maintaining the canonical Blockchain state across the P2P network can be mapped as a fault-tolerant state-machine replication problem [70]. In other words, each consensus node maintains a local replica of the Blockchain while keeping a unique and synchronized common view of the Blockchain.

According to [70], [71], a Blockchain updating protocol is said to achieve the (probabilistic) consensus (atomic broadcast [70] in a Byzantine environment if the following properties are (probabilistically) satisfied [72]:

- Validity (Correctness): If all the honest nodes activated on a common state propose to expand the Blockchain by the same block, any honest node transiting to a new local replica state adopts the Blockchain headed by that block.
- Agreement (Consistency): If an honest node confirms a new block header, then any honest node that updates its local Blockchain view will update with that block header.
- Liveness (Termination): All transactions originating from the honest nodes will be eventually confirmed. This ensures that the process does not get stuck indefinitely. Even if some nodes are unresponsive or slow, the system eventually reaches a conclusion.
- Total order: All honest nodes accept the same order of transactions as long as they are confirmed in their local Blockchain views.

Multiple consensus protocols have been proposed in the literature that possess these properties. We present existing consensus methods and discuss the possibility of applying them to a Blockchain-based IoT network. In addition, we compare all the discussed consensus protocols in Table 2.4 to indicate the most promising ones for IoT networks. The consensus protocols that are not applicable to IoT networks or are mere modifications of a general consensus method are discussed in brief. Two distinct categories of consensus protocols are used in Blockchain: Proof-based and Voting-based.

Proof-based consensus protocols

Proof-based consensus protocols, also known as Proof-of-X (PoX) protocols, are mechanisms employed in Blockchain technology to establish agreement among network participants regarding the validity of transactions and the ordering of blocks in the chain. These protocols rely on computational or economic proofs to ensure that nodes in the network agree on the state of the ledger. The fundamental idea behind proof-based consensus is to require nodes to provide evidence or demonstrate a certain level of computational effort or economic commitment, thereby discouraging malicious behavior and incentivizing honest participation.

The most widely adopted consensus algorithm in various Blockchain networks is Proof of Work (PoW) [73]. PoW is a mathematical puzzle that must be solved by a miner to validate a new block while adding it to the chain. Miners invest efforts in finding a nonce value such that the hash of the block when combined with the nonce and hashed, results in a specific number of leading zeros. PoW introduces a difficulty value to determine the required number of leading zeros. Consequently, the network generates a limited number of blocks, one every 10 minutes. PoW is a protocol that consumes a lot of computational resources and energy, given the computational complexity involved in solving the cryptographic puzzle. Although Proof of work

has proved to be an effective approach for cryptocurrencies over the years, it does not seem to be practical for IoT networks due to its high computational and bandwidth requirements. There exist some other consensus methods which are based on proof of work. These methods have tried to address some of the limitations of PoW, especially its high computational requirement. These methods are discussed next.

Proof of Stake (PoS) is the main consensus algorithm used in Ethereum [13]. PoS aims to address the challenges related to computational resources and energy consumption in PoW-based Blockchain platforms. In PoS, the mining node bets a certain amount of assets to mine blocks, and the mining power depends on the value of the bet assets. PoS relies on nodes that have invested more assets, resulting in a lower probability of success of an attack on the Blockchain. Although it has significantly eased the computational requirements of PoW, it is not yet popular for resource-constrained IoT networks. In addition, this method and its variants, Delegated Proof of Stake (DPoS), Leased Proof of Stake (LPoS), Proof of Importance (PoI), and Proof of Activity (PoA) are based on monetary concepts (stakes) that do not exist in IoT networks

Proof of Capacity (PoC) is similar to PoW but instead of depending on the computing power of the miners, it relies on their hard disk capacity. Thus, it is significantly more energy-efficient than ASICs mining used in PoW. In PoC, miners have to store huge data sets, known as plots, to get an opportunity to mine the next block. Therefore, by storing more plots, a miner will gain a higher chance of solving the next block [74]. The block creation time in PoC is 4 minutes. PermaCoin and SpaceMint are two cryptocurrencies employing PoC. Aside from the high latency, this method is not a rational choice for IoT networks where the devices have limited storage capacity.

Proof of Elapsed Time (PoET) is a consensus method proposed by Intel that works similarly to PoW but with significantly lower energy consumption. In this method, miners have to solve a hash problem similar to that of PoW. However, instead of a competition between miners to solve the next block, the winning miner is randomly chosen based on a random wait time. The winning miner is the one whose timer expires first. The verification of the correctness of timer execution is done using a trusted execution environment (TEE) like Intel's Software Guard Extension (SGX) [75].

PoET's eased computational requirements make it IoT-friendly. In addition, its low latency and high throughput make it favorable for IoT networks. The main drawback of this approach is its dependency on Intel which is in conflict with the basic philosophy of Blockchain being entirely decentralized.

These proof-based consensus protocols represent various approaches to achieving consensus in Blockchain networks. They each have distinct advantages, trade-offs, and suitability for different use cases. The choice of which protocol to implement depends on factors such as the desired level of security, energy efficiency, and decentralization.

Vote-based consensus protocols

Unlike Proof-based protocols that rely on computational or economic proofs, vote-based protocols operate on the principle of nodes reaching consensus through a voting process. In these protocols, nodes communicate and exchange messages to ascertain agreement on the validity of transactions before they are added to the Blockchain. The main problem with these mechanisms is the presence of Byzantine nodes, which leads to the Byzantine Generals Problem [76].

The Byzantine Generals' Problem is a seminal concept in distributed computing, initially formulated by Leslie Lamport, Robert Shostak, and Marshall Pease in 1982 [77]. It presents a scenario involving multiple generals, each commanding a segment of an army, positioned around a city they aim to besiege. The critical challenge is to coordinate a synchronized attack, an imperative for success. However, communication between generals is solely through messengers, and these intermediaries may be treacherous. The essence of the problem lies in finding a

strategy that guarantees unanimous agreement on the timing of the attack, even amidst the potential presence of unreliable or deceitful communication channels.

The development of consensus algorithms addresses this problem by providing mechanisms for nodes to converge on a mutually agreed course of action, even when faced with byzantine failures. Practical Byzantine Fault Tolerance (PBFT) involves the participation of all nodes in the voting process for block inclusion. Consensus is attained when more than two-thirds of all nodes concur on the selected block. PBFT mandates a minimum of $3f + 1$ replicas for proper operation, with f representing the maximal count of faulty replicas. This threshold ensures the presence of adequate non-faulty replicas to identify the malfunctioning ones, whether Byzantine or incapacitated. Byzantine Fault Tolerance (BFT) protocols present a robust mechanism for establishing highly dependable and accessible systems. PBFT can achieve consensus more expeditiously and efficiently, with fewer resource demands in comparison to PoW. Furthermore, akin to PoS, it does not necessitate ownership of assets to partake in the consensus process [76].

Delegated Byzantine Fault Tolerance (dBFT) adheres to the same principles as PBFT but dispenses with the requirement for all nodes' participation in block addition, rendering it more scalable. In dBFT, specific nodes are appointed as delegates of others and, in accordance with predetermined rules, undertake the consensus protocol akin to PBFT. In a distinct approach, AlgoRand, an algorithm based on Byzantine agreement, was proposed, enabling miners to achieve consensus within a single round. This process eschews rewards for miners, designating Validators instead. Validators are randomly chosen to validate the ensuing block, which is then disseminated across the network. Each Validator casts a vote for a block, and consensus is achieved when all Validators support the same one. The block with the highest number of votes is designated as the next block. PBFT has high throughput, low latency, and low computational overhead–all of which are desirable for IoT networks. However, its high network overhead makes it un-scalable for large networks, thus it could be applied only to small IoT networks. Other dBFT-inspired protocols have been proposed, such as Tendermint, Ripple and ByzCoin. Considering high scalability, high throughput, and low latency, this method could be applied to IoT networks if the monetary concept is replaced by some other criteria.

Raft introduces a voting-based consensus method aimed at enhancing the comprehensibility and implementability of the Paxos algorithm. While Paxos strives to resolve the consistency quandary under specific conditions in the Byzantine Generals Problem, Raft achieves analogous efficiency levels. Notwithstanding the similarities in protocol with BFT algorithms, Raft can only tolerate crash faults affecting up to 50% of nodes, unlike Byzantine algorithms, which can withstand arbitrary (including malicious) corruption. Raft exhibits elevated throughput and reduced latency; however, its efficacy hinges on the integrity of the leader node, which wields unequivocal dominance within the system. Consequently, if the leader node is subject to malicious compromise, the entire system is imperiled. Raft lacks resilience to malicious nodes and can endure up to 50% node failure due to crashes. Since it is crucial to secure the leader node, the throughput is limited by the performance of that node. Due to its low security and restricted throughput, it is not very appropriate for IoT networks [78].

Tangle is a new technology for distributed ledgers proposed by the cryptocurrency Iota. It does not require a complicated, time-consuming, and computationally intensive consensus protocol. It also does not use blocks to store transactions. Each transaction is a unique block by itself which must approve two older transactions in order to be added to the ledger. Tangle uses a directed acyclic graph (DAG) in which each transaction is linked to two older transactions that are approved by it. After a transaction approves two older transactions, it is added to the ledger through PoW.

Due to the unique design of Tangle, it is a fast, infinitely scalable framework which makes it well-suited for IoT networks. Furthermore, Tangle has no transaction fees which is desirable for an IoT network. In contrast to most Blockchain implementations, the Tangle is immune to becoming obsolete with the advent of quantum computers because of its unique design. The

main challenge with Tangle is how to choose the two older transactions for approval. No rule is imposed by Tangle on how to choose these two nodes which is very desirable for resource-constrained devices in an IoT network. However, the chosen transactions should not be the same or conflicting. To choose between conflicting transactions, Tangle runs an algorithm called the tip selection algorithm multiple times.

Unlike Blockchain frameworks, Tangle's design enables parallel transaction verification which eliminates the required wait time for mining previous blocks as in Blockchain and provides the opportunity to verify more transactions in a shorter time. Although Tangle is very promising and claims to overcome the existing barriers to decentralization of resource-constrained IoT networks [79], it confronts a lot of implementation challenges, specifically for IoT applications. The current implementation of Tangle, Iota, does not provide all the claimed goals of Tangle. One of the challenges of applying Tangle to IoT networks is the storage limitation. The resource-constrained IoT devices are unable to store the entire Tangle. Some solutions including automated snapshotting and a swarm client have been proposed to address this problem in Iota's development roadmap [80]. Another problem with Tangle is that whoever gains control over more than one-third hash power of the Tangle can make it insecure and vulnerable. As a preventive measure, Iota runs a node called 'coordinator' by amassing the hash power itself at one point. However, this can be perceived as the centralization of Tangle.

The aforementioned consensus methods have found application in various Blockchain implementations. The choice of consensus method serves as the fundamental framework for a Blockchain implementation. Consequently, many of the attributes and performance metrics of a Blockchain implementation depend on the consensus method selected. For IoT networks, the most crucial attribute is low latency and scalability [71]. In a real-world IoT network scenario, a transaction should be transmitted and finalized within a few milliseconds.

Table 2.4 presents a comprehensive comparison of classical and recently proposed consensus methods to assess their suitability for IoT networks. The most suitable consensus methods for IoT networks are marked with ●, consensus methods partially suitable are marked with ◐, and methods not applicable to IoT networks are marked with ○. As can be observed, voting-based consensus protocols are most suitable for IoT because they generally require low computational resources and have high throughput. This aligns with the needs of IoT applications where most devices are resource-constrained and require low latency and high transaction throughput.

Table 2.4: Comparison of different consensus algorithms. Source [76].

Consensus	Accessi-[1] bility	Decentra-lization	Scala-bility	Through-[2] put	Latency[3]	Adversary tolerance	Computing overhead	Network overhead	Storage overhead	IoT suitability
Proof-based Consensus Protocols										
PoW	Public (PL.)	High	Medium	Low	High	25% Comp. Power	High	Low	High	○
PoC	Public (PL.)	High	Medium	Low	High	N/A	Low	Low	Very High	○
PoS	Public (P. or PL.)	High	Medium	Low	Medium	51% Stakes	Medium	Low	High	◐
PoI	Public (PL.)	High	Medium	High	Medium	51% Importance	Low	Low	High	◐
PoET	Public (P. or PL.)	Medium	High	High	Low	N/A	Low	Low	High	●
PoA	Public (PL.)	High	Medium	Low	Medium	51% online Stakes	High	Low	High	○
PoB	Public (PL.)	High	Medium	Low	High	25% Comp. Power	Medium	Low	High	○
LPoS	Public (PL.)	High	Medium	Low	Medium	51% Stakes	Medium	Low	High	○
DPoS	Public (PL.)	Medium	Medium	High	Medium	51% Validators	Medium	N/A	High	◐
RSCoin	Private (P.)	Low	Medium	High	Low	N/A	Low	Medium	High	○
Tangle	Public (PL.)	Medium	Medium	High	Low	33% Comp. Power	Low	Low	Low	◐
Voted-based Consensus Protocols										
PBFT	Private (P.)	Medium	Low	High	Low	33% faulty Validators	Low	High	High	●
dPBFT	Private (P.)	Medium	Medium	High	Medium	33% faulty Validators	Low	High	High	◐
Stellar	Public (PL.)	High	Medium	High	Medium	Variable	Low	Medium	High	◐
Ripple	Public (PL.)	High	Medium	High	Medium	20% faulty UNL nodes	Low	Medium	High	◐
Tendermint	Private (P.)	Medium	Medium	High	Low	33% Voting power	Low	High	High	◐
OmniLedger	Public (PL.)	High	Medium	High	Medium	25% faulty Validators	Medium	Medium	Low	◐
RapidChain	Public (PL.)	High	Medium	High	Medium	33% faulty Validators	Medium	Low	Low	◐
Raft	Private (P.)	Medium	High	High	Low	50% crash fault	Low	Low	High	◐
ByzCoin	Public (PL.)	High	Medium	High	Medium	33% faulty Validators	High	Medium	High	○
Elastico	Public (PL.)	High	Medium	Low	High	25% faulty Validators	Medium	High	High	○
Casper	Public (PL.)	High	Medium	Medium	Medium	51% Validators	Medium	Low	High	○
Algorand	Public (PL.)	High	Medium	Medium	Medium	33% Weighted Users	Low	High	High	○

[1] PL. denotes Permissionless and P. denotes Permissioned.
[2] We consider that Low throughput is less than 100 TPS (Transactions per Second), Medium throughput is between 100 TPS and 1000 TPS, and High throughput is more than 1000 TPS.
[3] High latency is in the order of minutes, Medium latency is in the order of seconds, and Low latency is in the order of milliseconds.

PoET, PBFT, and Tangle were found to be the best options for Blockchain-based IoT networks because of possess the mentioned required features for IoT networks. However, even these methods cannot fully address the shortcomings of Blockchain-based IoT networks. PBFT has a high network overhead which restricts its scalability. Therefore, it is only practical for small IoT networks such as smart homes. The main drawback of PoET is its dependency on Intel which makes it significantly centralized compared to other consensus methods. While Tangle is claimed to address all the limitations of IoT networks, in practice, it also suffers from centralization [70].

2.5.4 Benefices of the Integration Blockchain-IoT

Recognizing the potential benefits of Blockchain technology and its envisaged impact, researchers and developers have endeavored to create decentralized applications tailored for the IoT. As expounded earlier, the inherent attributes of Blockchains position them as a natural fit for establishing a secure distributed framework for the IoT and distributed cloud computing in a broad sense. Grounded in these attributes, the ensuing enumeration outlines the potential advantages and driving factors for cultivating a Blockchain-based decentralized IoT framework:

Resilience: IoT applications necessitate the preservation of data integrity during transmission and analysis. Consequently, IoT frameworks must exhibit resilience against data leaks and integrity breaches. Blockchain networks maintain redundant replicas of records across peers within the Blockchain, thus aiding in the preservation of data integrity and affording resilience to IoT frameworks.

Adaptability: Currently, the heterogeneous nature of IoT devices and protocols imposes limitations on their interoperability. Given that Blockchains function as semantics-independent distributed databases, their integration as the network control mechanism for the IoT promises to augment its adaptability. Blockchains have demonstrated their efficacy across diverse hardware platforms, and a Blockchain-centric IoT framework holds the potential to flexibly accommodate varying environments and use cases to meet the evolving demands of IoT stakeholders.

Fault Tolerance: The proliferation of always-available smart devices in the Internet of Things (IoT) results in the collection of data and provision of automated functionality. Network control mechanisms for the IoT necessitate high availability, which may not always be guaranteed in architectures relying on centralized servers. Blockchains function as Byzantine fault-tolerant record-keeping mechanisms capable of identifying failures through distributed consensus protocols.

Security and Privacy: A paramount challenge faced by the IoT, as previously discussed, pertains to network security. In order to ensure confidentiality and data protection, Blockchains incorporate pseudonymity in their addressing and implement distributed consensus for record immutability. Public Blockchains thwart data modification attacks due to the absence of a singular location for the Blockchain. Additionally, the cost associated with initiating new transactions (whether monetary or computational) fortifies the network against flooding attacks and Distributed Denial-of-Service (DDoS) attacks.

Trust: Blockchains facilitate trust among transacting parties. The ”trustless” attributes of Blockchains obviate the necessity for users to repose trust in centralized entities for managing their IoT data, thereby preventing malevolent third-party entities from aggregating users' private data. Blockchains expedite swifter settlements for automated contracts, obviating the need for trusted intermediaries.

Reduced Maintenance Costs: A pivotal stride towards the global integration of the IoT involves devising efficient and economical methods to handle the prodigious volume of data generated by sensors across the IoT. Cloud-based IoT frameworks contend with a significant drawback in the form of elevated server maintenance costs, which not only impose monetary burdens but also escalate communication expenses in device-to-device interactions. Centralized cloud storage services rely on geographically dispersed data centers, which serve as substantial

single points of failure. While centralized cloud services introduced considerably lower prices for storage and computing, Blockchains harbor the potential to substantially curtail expenses associated with maintaining dedicated servers. Public Blockchain applications obviate the necessity for dedicated servers, leveraging the computational and storage capacities of participating nodes. Since participants receive incentives for their contributions, Blockchains emerge as the next stride in democratizing the IoT.

2.5.5 Integration Schemes for Blockchains and IoT

Achieving absolute decentralization in the IoT using Blockchains is challenging, considering the vastly varying devices involved in the IoT. Most devices on the IoT edge have resource constraints, and cannot host a copy of the Blockchain or engage in validating new blocks for the Blockchain. Therefore, it is important to decide upon what roles the different entities in the IoT edge (devices, gateways, etc) will play.

Table 2.5: Nodes type in Blockchain Networks

Node type	**Storage**	**Validator**
Full Node	Full Blockchain	Yes
Light Node	Block Headers	No
Transaction Issuer	None	No

Table 2.5 presents the possible roles the participants of a Blockchain network can assume. Full nodes are participants in the Blockchain network that hosts the entire copy of the Blockchain. Full nodes can issue transactions to the Blockchain and can choose to act as a Validator for adding new blocks onto the Blockchain. Light nodes can issue transactions to the Blockchain and can host a copy of the block headers from the Blockchain. Light nodes can verify the validity of transactions through the block headers, however, they do not publish new blocks to the Blockchain. Light nodes are used as an easier entry point to the Blockchain, using limited computational resources. A Transaction issuer is a participant that does not maintain a copy of the Blockchain or engage in block validation, however, it simply issues transactions to the Blockchain.

Keeping in mind the resource constraints faced by IoT devices, it becomes necessary to employ some design considerations about the extent of their involvement in a Blockchain network. Most IoT devices do not have cryptographic capabilities or meet the computational and storage requirements for engaging in Blockchain consensus protocols. To account for these limitations, IoT edge devices only take on the role of simple transaction issuers. Even in the case of light nodes, most IoT edge devices do not carry sufficient storage capabilities to host the ”headers only” version of the Blockchain. IoT edge devices or gateways running as simple transaction issuers have verifiable Blockchain identities without the need to host an entire copy of the Blockchain. Therefore, such edge devices are more manageable within Blockchain networks and can continue making contributions to the Blockchain, while other full nodes in the Blockchain network can carry out decentralized consensus and block validation [27], [81].

Fog computing [82] has also revolutionized the IoT with the inclusion of a new layer between cloud computing and IoT devices and could also facilitate this integration. fog computing refers to a decentralized computing architecture that positions computational resources closer to IoT devices, mitigating latency and reducing dependence on centralized cloud servers. By processing data locally or at the network edge, fog computing enhances efficiency, privacy, and security, while also allowing for dynamic decision-making and resilience to network failures.

Recent works [27], [83] have conducted an extensive survey of diverse integration methodologies designed to accommodate the limitations of IoT edge devices within a Blockchain-based IoT framework. Another aspect to take into account is related to the IoT interactions, *i.e.,* the communication between the underlying IoT infrastructure. When integrating Blockchain, it needs to be decided where these interactions will take place: inside the IoT, a hybrid design involving IoT and Blockchain, or through Blockchain. Below, these alternatives (shown in Fig. 2.7) are described.

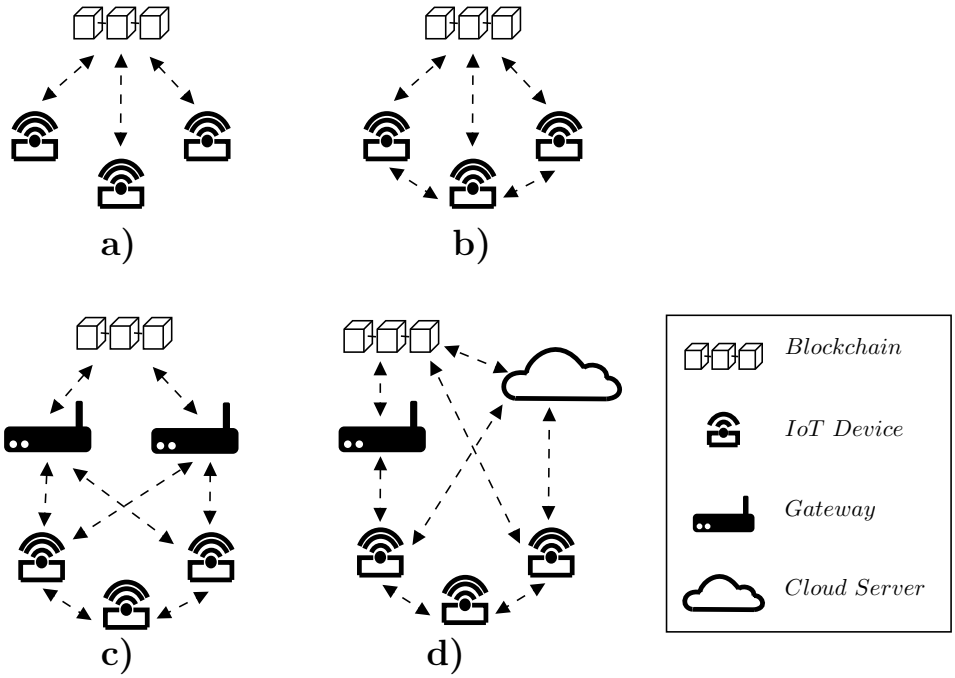

Figure 2.7: Blockchain integration schemes for the IoT. All arrows indicate interactions. **a)** IoT edge devices as transaction issuers to the Blockchain, **b)** Interconnected edge devices as end-points to the Blockchain, **c)** Gateway devices as end-point to the Blockchain and **d)** Hybrid Cloud/Blockchain approach. Adapted from [83] and [27]

- Devices as transaction-issuers to the Blockchain: This integration scheme is seen in [83], however, in our discussion, we are assuming that the IoT devices are not in fact carrying a copy of the Blockchain but are simply issuing transactions to the Blockchain, as shown in Fig. 2.7 a). Similar to the previous approach, all IoT interaction events are logged onto the Blockchain for secure accountability. In this approach, IoT devices can be provided with cryptographic functionality. The trade-off here is a higher degree of autonomy of IoT devices and applications, versus increased computational complexity of IoT hardware.

- Interconnected edge devices as end-points to the Blockchain: In this approach [83], IoT gateways and devices issue transactions to the Blockchain and can communicate with each other off-chain, as seen in Fig. 2.7. While introducing the need for routing and discovery protocols, this approach ensures low latency between the IoT devices and the choice to log specific interactions on the Blockchain. This integration scheme would be more suited to scenarios where interactions are much more frequent and high throughput, low latency, and reliable IoT data are required.

- Gateway devices as end-points to the Blockchain: In this integration scheme, all communications go through the Blockchain, while the IoT gateways act as end-points to the Blockchain network. In this case, the IoT devices will be registered to the gateway device,

and the gateway issues transactions to the Blockchain. This approach enables traceability of all communications involving a specific IoT gateway and IoT service. This integration scheme can also be used to authenticate communications between devices connected to separate Blockchain-enabled gateways [84]. In this approach, not all of the data transferred needs to be stored on the Blockchain. The Blockchain itself can be used as a control mechanism, with Smart Contracts acting as programmable logic, while data transfer can occur over peer-to-peer technologies like BitTorrent and IPFS [85]. However, recording all IoT interaction events on the Blockchain will increase bandwidth and storage requirements, and currently, scalability is a well-known research challenge in the integration of Blockchains and IoT. Figure 2.7 c) is an illustration of this approach. The degree of decentralization achieved through this approach is not as fine-grained as in the case where devices issue transactions directly to the Blockchain.

- Cloud-Blockchain hybrid with the IoT edge: This approach is an extension to the previous integration scheme, whereby IoT users have a choice to use the Blockchain for certain IoT interaction events, and the remaining events occur directly between IoT devices [83]. This approach leverages the benefits of decentralized record-keeping through Blockchains as well as real-time IoT communication. Figure 2.7 d) is an illustration of this hybrid integration scheme. The challenge posed by this approach is to optimize the split between the interactions that occur in real time and the ones that go through the Blockchain. Hybrid approaches can utilize fog computing [86] to overcome the limitations of Blockchain-based IoT networks related to latency and processing capabilities.

The choice of which integration scheme to implement is contingent upon the specific requirements of the IoT application. For example, in scenarios where maintaining an immutable record is paramount and there are relatively fewer interactions occurring, the first two interaction schemes are more appropriate. However, in cases demanding higher performance, relying solely on a Blockchain may prove insufficient, warranting the adoption of a hybrid integration scheme. It is important to note that in IoT applications, neither IoT devices nor gateways should be employed as full nodes. This is because the associated storage and computational burdens may outweigh the potential benefits.

One clear alternative to the integration of Blockchain with the IoT is the integration between the IoT and cloud computing [87]. This integration has been used in the last few years to overcome the IoT limitations of processing, storage, and access. However, cloud computing usually provides a centralized architecture, which in contrast to Blockchain, complicates reliable sharing with many participants. The integration between Blockchain and the IoT is intended to address previous limitations in addition to maintaining reliable data. Fog computing aims to distribute and bring the computing closer to end devices, following a distributed approach like Blockchain. This can incorporate more powerful devices than the IoT such as gateways and edge nodes, which can then be reused as Blockchain components. Therefore, fog computing could ease the integration of the IoT with Blockchain.

2.6 Fairness on Blockchain

Fairness has been gaining interest in the past few decades. The decision-maker (DM) the objective is to maximize efficiency, but if fairness is not considered the service receivers are not satisfied and claim that the distribution of resources is unfair or unjust and could be demotivated to use the service. If the objective is to distribute resources fairly, the DM is not satisfied if the resources are not utilized. A balanced solution between fairness and efficiency is the goal of fair resource allocation. Fairness in its early development was applied in the microeconomics of social welfare. Every individual or demand is assigned a utility function based on the preference of the individual assuming that the DM is aware of the preferences of each individual.

Conventional optimization models typically strive for efficiency by maximizing total benefit or minimizing total cost. Benefit can be measured in many different ways, such as profit, revenue, output, or health outcomes obtained, and cost can be measured by labor, materials, and resources invested or undesirable outputs generated. The common thread in these models is that the benefits or costs are dispersed across stakeholders. However, by pursuing an efficiency goal, a conventional optimization model may lead to an unfair distribution of benefits and costs among the stakeholders. Some may receive less than they should, and some more than they should, relative to the others.

The term fairness has found numerous definitions in Blockchain literature. For many consensus protocols, fairness does not come naturally. The approach of Helix [88] provides fairness by allowing pending transactions to be selected with the same probability as a block through random block selection [89]. Sokolik et al. suggested reducing the tail latency of the time it takes a transaction to be included in a block by giving priority in the block selection to transactions observing high latency [90]. The notion of fairness among transactions is defined differently in [91], as each node gets a fair share of the ledger. Namely, each block contains the same number of transactions from each node assuming that they have infinite streams of transactions. Another related definition is due to Receive-order-fairness [92] which enforces transaction selection such that if many nodes learned about a transaction before some other transactions, such an order should be reflected in the ledger. Weaker potential definitions refer to the unfairness of the order of two transactions only if they were received sufficiently apart in time. Another option is to ignore the internal order of transactions within blocks and only refer to the order of transactions in different blocks.

Wendy [93] handles the relative order fairness and claims for fairness requirements only for subsets of transactions, e.g., those belonging to each of several existing markets. This approach is different than that of Helix, where fairness is kept between the different applications and it is assumed that a transaction has no priority regarding the internal order of its transactions. Table 2.6 overviews these different fairness aspects. Beyond Blockchain, aspects of fairness have been studied in other networking and computer system settings where a restricted resource is shared among multiple entities. Examples include a queue with a bounded service rate or a link with limited capacity [94]. A well-known notion is that of max-min fairness, which suggests how to determine a resource partition based on the demands of multiple users, summing up to more than the resource availability. Intuitively, such a fair allocation tries to maximize the share of users with small demands. Generalizations of the definition have also been suggested [6].

Table 2.6: Consensus protocol for Permissioned Blockchain

	Fairness aspect
Helix [88]	Similar probability for a transaction to be selected for a block
Age-aware fairness [90]	Prioritizing transactions with a large observed latency in block selection
Fair share [91]	Similar block parts among nodes
Receive-order-fairness [92]	Transaction order is based on time nodes learn on each transaction
Relative order fairness (Wendy) [93]	Internal fairness within subsets of transactions
Cryptocurrency Payments [95]	Payment-for-receipt solution: optimistic fair payments.

In addition, existing consensus protocols for Blockchain do not consider fairness among users during the processes, such as the Validator selection in a consensus mechanism based on voting, we drive this problem in this work.

2.6.1 Social Welfare and Fairness Metrics

Welfare economics has long used social welfare functions (SWFs) as a tool to measure the desirability of a given distribution of benefits and harms. A SWF is a function of the utility levels allocated to affected parties, where utility reflects a party's gain or loss as a consequence of the decisions of interest. Using a SWF motivates explicit consideration of the downstream

outcomes of fairness and equity criteria. In contrast to leading notions of fairness that focus on eliminating disparity between groups, SWFs allow a broader perspective that emphasizes fairness in the welfare impacts of decisions.

Table 2.7: Fairness Measures. Source [96]

Value of beta	**Fairness measure**	**Known Name**
$\beta \to \infty$	$-max_i \left\{ \frac{\sum_i x_i}{x_i} \right\}$	Max ratio
$\beta \in (1, \infty)$	$-\left[(1-\beta) U_{\alpha=\beta} \left(\frac{x}{\sum_i x_i} \right) \right]^{\frac{1}{\beta}}$	α fair utility
$\beta \in (0, 1)$	$\left[(1-\beta) U_{\alpha=\beta} \left(\frac{x}{\sum_i x_i} \right) \right]^{\frac{1}{\beta}}$	α fair utility
$\beta \to 0$	$e^{H\left(\frac{x}{\sum_i x_i} \right)}$	Entropy
$\beta \in (0, -1)$	$\left[\sum_{i=1}^{n} \left(\frac{x_i}{\sum_i x_i} \right)^{1-\beta r} \right]^{\frac{1}{\beta}}$	No name
$\beta = -1$	$\frac{(\sum_i x_i)^2}{\sum_i x_i^2} = n \cdot J(x)$	Jain's index
$\beta \in (-1, -\infty)$	$\left[(1-\beta) U_{\alpha=\beta} \left(\frac{x}{\sum_i x_i} \right) \right]^{\frac{1}{\beta}}$	No name
$\beta \to -\infty$	$min \left\{ \frac{\sum_i x_i}{x_i} \right\}$	Min ratio

We briefly review a collection of SWFs to illustrate how they can embody various conceptions of equity. For each, we indicate the type of optimization model. it yields, and whether it is appropriate for our running example of mortgage loan processing. In [97] the SWFs are classified as pure fairness metrics, functions that combine fairness and efficiency, and statistical fairness metrics. In [96], a unified representation of the constructed fairness measures.

The set of resources X is to be distributed among individuals. Given that $x \in X$ is a feasible allocation of resources among the individuals chosen by the DM, then $f_i(x)$ is the utility of individual i for every $i = 1, \ldots, n$. This leads to the utility set U for all individuals:

$$U = \{u_i = f_i(x), \forall i = 1, \ldots, n)\} \tag{2.2}$$

From a user perspective, its perception of maximum fairness is independent of the population size of the system. The fairness measures are defined as follows:

$$f_\beta(x) = sign(1-\beta) \left[\sum_{i=1}^{n} \left(\frac{x_i}{\sum_j x_j} \right)^{1-\beta} \right]^{\frac{1}{\beta}} \tag{2.3}$$

We summarize the special cases in Table 2.7, where β sweeps from $-\infty$ to ∞, and $H(\cdot)$ denotes the entropy function. For some values of β, the corresponding mean function h has a standard name, and for others, known approaches to measure fairness are recovered. Additionally, for $\beta \in (0, -1)$ and $\beta \in (-1, -\infty)$, new fairness measures are discovered. In this work, we employ the α-Fairness as an SWF approach given the combination of fairness and efficiency it provides.

2.7 Conclusion of the Chapter

The Internet of Things (IoT) demonstrates a wide range of applications that significantly impact various aspects of daily life. Its potential reaches into numerous domains, offering transformative benefits to individuals and industries alike. However, the sensitivity of the information involved in these applications has made them susceptible to a range of attacks across different layers of the IoT architectures. This underscores the critical importance of robust security measures to safeguard the integrity and confidentiality of data.

It has been observed that classical cryptographic solutions are not directly applicable to IoT devices due to their resource constraints. This necessitates the adoption of lightweight cryptographic techniques, such as those based on elliptic curves, to ensure efficient and effective encryption methods within the IoT ecosystem. Traditional solutions often feature a centralized architecture, presenting several challenges in terms of scalability and security, particularly within the dynamic and distributed environment of IoT. As such, a move towards decentralized architectures, enabled by technologies like Blockchain, offers a compelling solution to mitigate these concerns.

Blockchain technology, characterized by its decentralized and immutable ledger, stands out as a promising solution to address issues of centralization and security within IoT applications. By distributing authority and control, Blockchain provides a foundation for increased trust and transparency. Nonetheless, the integration of Blockchain and IoT is not without its hurdles. One of the foremost challenges lies in selecting an appropriate consensus mechanism tailored to each application, given its profound impact on the overall system's performance. Striking the right balance between security, efficiency, and scalability remains a critical consideration. Current consensus mechanisms predominantly focus on optimizing transaction processing efficiency and often neglect considerations of fairness among users. This oversight can potentially lead to user disengagement and dissatisfaction. Therefore, efforts towards achieving fairness in Blockchain processes are essential for sustaining a healthy and participative ecosystem.

In summary, this chapter underscores the imperative of implementing robust security and privacy measures within IoT and highlights Blockchain as an attractive solution. Additionally, it emphasizes the necessity of lightweight security solutions that consider the resource constraints of IoT devices. Furthermore, active endeavors toward achieving fairness in Blockchain processes are crucial for fostering an equitable environment for Blockchain users.

Chapter 3

μTesla-based Authentication for Reliable and Secure Broadcast Communications in IoD using Blockchain

3.1 Introduction

Drones allow access to hard-to-reach places with low energy, time, and manpower consumption. As a result, drone applications have expanded rapidly in various branches of human development, ranging from supply chain and agriculture applications, and rescue operations to military applications. The Internet of Drones (IoD) paradigm employs a layered network architecture to facilitate communication and coordination among drones. As leakage of sensitive information in IoD applications could result in significant economic and social losses, ensuring the security and privacy of exchanged data is imperative. Particularly, the authentication of legitimate IoD devices plays a critical role [98], [99].

Drones have limited power, computational, and storage resources. The battery is used simultaneously for flight functions, communications, and onboard processing systems. These limitations, combined with the high mobility of drones, are the reasons why ensuring authentication is a significant challenge. Therefore, optimizing the power and time consumption of authentication protocols maximizes flight time and the level of operability in the missions.

Authentication is generally addressed through a centralized solution, where digital certificates are issued by a Certificate Authority (CA). Conventional centralized solutions have scalability issues as the network expands. In addition, they often have a single point of failure and are susceptible to disruption in the event of Denial of Service attacks (DOS) or technical failures [100], [101].

Due to the limitations of centralized solutions, several authentication protocols incorporating Blockchain technology have been proposed in the literature to address centralization and security concerns in conventional solutions [102]. Blockchain is a type of Distributed Ledger Technology (DLT) that uses cryptographic techniques to create an immutable record of data. The data is stored in blocks, which are replicated on all nodes in a peer-to-peer network. Peers in the network do not need to trust each other and maintain a local copy of the ledger. The consensus algorithm, developed by the peers, is responsible for adding new blocks of data to the Blockchain in a distributed manner across the network.

3.1.1 Research Motivation and Contribution

Most communications in IoD networks are conducted over a public channel in a broadcast fashion. As a consequence, it is essential that all source devices and corresponding communication messages are properly authenticated in the network. In addition, given the high mobility of drones, it is possible to lose some packets. In the case where any of the lost packets contain authentication handshake information, the authentication will fail and must be restarted. Moreover, drones can change from one fly zone (domain) to another, *i.e.,* handover, which results in

some cases in the need for re-authentication. Therefore, given the high cost of the authentication process for the drones, the authentication protocol must also consider handover operations in the authentication and support packet losses.

Taking into account the limitations of drones and the fact that many IoD applications are sensitive to delays, it is essential that the authentication protocol is efficient in terms of energy consumption, authentication time, and computational complexity. Several existing works have proposed authentication protocols for IoD networks [103]–[109]. However, these proposals are not resilient to packet loss during communication, do not integrate handover solutions, and in some cases are vulnerable to certain attacks.

μTesla [110] is a well-known lightweight authentication protocol designed especially for resource-constrained devices that are both computationally lightweight and robust to packet loss. In this Chapter, we propose an authentication scheme inspired by μTesla, using Blockchain technology to solve the limitations of the original protocol. The main contributions of the present protocol are summarized below:

- Our approach uses Blockchain to support drone authentication information. Additionally, we use lightweight cryptographic operations, *e.g.,* like hash function and eXclusive OR (XOR), making our protocol computationally lightweight. Our solution is resistant to different security attacks and robust to packet loss and handover events.
- We provide a security analysis of the proposed protocol against the main attacks to which IoD networks are vulnerable and compare it with some existing works in the literature.
- We propose a storage optimization algorithm to address the storage requirements in μTesla.
- In addition to the security analysis, we evaluate by simulation the performance of the protocol in terms of storage, communication overhead, and consumption of energy. The proposed protocol shows a very good trade-off between security and efficiency.

3.2 Related Work

3.2.1 Broadcast Authentication

Broadcast communications are critical in many applications because multiple receivers can be reached with the same packet, enabling rapid and efficient information exchange. Unfortunately, packet injection attacks and eavesdropping are easy to implement in a wireless environment, hence source authentication is necessary to avoid these security issues.

In the special context of IoD, an authentication scheme should provide resilience against various attacks, including eavesdropping, replay attacks, impersonation attacks, and man-in-the-middle attacks. Most point-to-point solutions are not secure against these attacks in broadcast transmissions and in some cases are not efficient enough. In IoD networks, authentication schemes must consider their dynamic and heterogeneous nature, as well as the resource constraints of drones. Therefore, it is necessary to authenticate the source of broadcast packets efficiently. A broadcast authentication protocol in IoD networks must meet the following performance and security requirements [111]:

- Secure and attack-resistant
- Low computational cost for generation and verification of authentication information
- Low communication overhead, and robust to packet loss and handover
- Scalable for a large number of receivers

- Decentralized architecture

Different authentication protocols have been proposed in the literature to address the requirements of IoD networks with broadcast communications. In the following section, we survey several of these existing solutions.

3.2.2 Authentication protocols for IoD

Depending on the architecture of key or certificate management, existing solutions for authentication can be classified as centralized or decentralized. Several protocols have been proposed in the literature for authentication in IoT and IoD networks that present a centralized architecture [103], [104], [112], [113]. In [114], the Timed Efficient Stream Loss-tolerant Authentication (Tesla) protocol is introduced. Tesla allows all receivers to check the integrity and authenticate the source of each packet in broadcast transmission. The protocol does not require trust between receivers, uses low-cost computational operations at both the sender and receiver and can tolerate the loss of packets without the need for retransmissions. The Tesla protocol achieves asymmetric properties by delaying the disclosure of secret keys while relying on symmetric Message Authentication Codes (MAC).

Perrig *et al.* proposed the Tesla-inspired protocol μTesla in [110], designed for resource-constrained networks. This protocol communicates the initial key in the key chain to all receivers, reducing the size of transmitted packets compared to the Tesla protocol, and saving time and energy. In addition, it restricts the number of authenticated senders by not storing the one-way key chain in all the nodes. Unlike the original Tesla, where a digital signature is used for initial packet authentication, it instead sends the initial key commitment to all receivers by unicasting. In our solution, we utilize μTesla to ensure the authentication of the drones. The μTesla protocol will be discussed in-depth in Section 3.3.3.

Centralized services possess the problem of a single point of failure, which makes them vulnerable to DoS attacks or technical failures that could disable the network. In addition, centralized solutions present scalability problems due to the deterioration in performance when the number of users (drones) that the server has to serve simultaneously increases.

To overcome the centralization drawbacks, Blockchain has been applied to distribute services. It replaces trusted entities with a publicly verifiable, tamper-proof, peer-to-peer distributed data storage that maintains its integrity. Several decentralized solutions for authentication employ a Blockchain to store and verify the validity of the identity and public key of devices. Using Blockchain alleviates Public Key Infrastructure (PKI) management without a third party while ensuring the security and privacy of the system [115].

Multiple Blockchain-based solutions for authentication have been proposed in different applications including Wireless Sensor Networks (WSN) [105], [115]–[117], Smart Home [118], Industrial IoT (IIoT) [106], [107], [119], [120] and IoV [121]. These protocols proposed for IoT are generally lightweight and could be adapted to IoD networks. However, recently, novel authentication protocols for IoD have been proposed. In [122], a secure and low latency authentication of drones using Blockchain-based security is addressed. The proposed architecture provides a transparent and efficient mechanism for data security as well as to ensure the secure migration of drones between different zones.

In addition, a cross-domain authentication scheme for 5G-enabled UAVs based on Blockchain is proposed [123]. The identity of each drone is dynamically managed by applying a multi-signature smart contract. Entities from different domains can authenticate each other without knowing their true identities. The Blockchain enables security auditing and the establishment of an accountability mechanism for the involved entities.

In [108], a Blockchain-assisted authentication service for industrial UAVs is designed. The distributed peer nodes in the Blockchain keep together the ledger that stores authentication

information. Industrial drones can call Smart Contract APIs to access the ledger to facilitate their authentication process, which is performed on the basis of ECC to ensure security.

The solution presented in [109] involves the implementation of a secure and efficient distributed authentication mechanism. The approach utilizes a multi-signature smart contract to facilitate mutual authentication between terminals operating in a distributed environment. This is achieved through the utilization of consortium Blockchain technology.

Based on the above literature review, we observe that many practical authentication mechanisms have been designed for IoT and specifically for IoD networks. Most of them consider the privacy preservation of the participants and successfully resist different attacks caused by internal or external attackers, while they are suitable for authentication for resource-constrained devices due to their computational efficiency. However, not all of these solutions offer resistance to the packet losses that occur in wireless networks and to the possibility of handover mechanisms, either due to technical problems or mobility.

Table 3.1 gives an overview of some existing solutions with demonstrated effectiveness in many contexts, highlighting their differences in terms of security requirements provided (or not) by these existing protocols. The last column shows the proposed protocol, which is based on μTesla to guarantee the authentication of broadcast packets in an efficient and packet loss-resistant way while providing decentralized and secure management of the key via the Blockchain. In contrast to other existing works, our approach provides all the analyzed security requirements which will be verified in Section 3.5.

Table 3.1: Comparison of authentication schemes

Req.	[103]	[104]	[105]	[106]	[107]	[108]	[109]	[122]	Ours
EA	•	•	•	•	•	•	•	•	•
DII	•	•	•	•	•	•	•	•	•
DIF	•	•	•	•	•	•	•	•	•
ESL	•	•	•	•	•	•	•	•	•
MITM	•	•	•	•	•	•	•	•	•
DoS	•	•	•	•	•	•	•	•	•
Dec	○	•	•	•	•	•	•	•	•
SKB	○	•	○	○	○	○	○	○	•
HA	○	○	○	○	○	○	•	•	•
MLT	○	○	○	○	○	○	○	○	•

• Provides ○ Does not provide

Requirements: **EA:** Eavesdropping Attack, **DII:** Drone Identity Impersonation, **DIF:** Drone Identity Forgery, **ESL:** Ephemeral Secret Leakage Attack, **MITM:** Man-in-the-Middle Attack, **DoS:** Denial of Service Attack, **Dec:** Decentralization, **SKB:** Symmetric Key Based, **HA:** Handover Authentication, **MLT:** Message Loss-Tolerant.

3.3 Preliminaries

In this section, some background is given on the different building blocks of the proposed solution, being the Blockchain, the cryptographic hash operation, and the μTesla protocol.

3.3.1 Blockchain Considerations

A Blockchain is essentially a distributed ledger on a peer-to-peer network that allows transactions to be securely stored and verified without the need for any centralized authority. Transaction serves as the fundamental unit of information exchange between entities. Blockchain allows

the transfer of data of any type directly from an externally owned account to another account on the Blockchain platform [124].

A typical transaction comprises several fields including *from*, *to*, *value*, *data*, and other fields related to mining. The *from* field contains the address of the sender, the *to* field contains the destination address to which the information is transmitted, and the *value* field represents the digital currency transmitted value. The *data* field is used to store attachment information for the transaction and is usually left empty. In our proposed protocol, we utilize the *data* field to store the necessary information required for the authentication of the drones and set to zero the field *value* due to the Blockchain model being used only to transmit and store the information required for authentication.

In the Blockchain structure, the information is stored in blocks and each block is linked to the immediately preceding block through a hash pointer. Thus, it is not possible to modify a block without being detected, since the hash value of the modified block is significantly different from that of the same block without modifications. Moreover, since the Blockchain is distributed among all peers in the network, any local change made by a dishonest node to the data in a block can be easily discovered by other nodes in the network. The process of adding a new block of information to the existing Blockchain involves a consensus protocol that is developed by all the peers in the network. This protocol enables the validation of the reliability and authenticity of the block within a decentralized and untrusted peer-to-peer environment, without the need for a trusted third party.

The proposed scheme incorporates the distinctive chain structure of a Blockchain for information flow, following the typical Blockchain structure but block contents differ slightly. In general, each block contains several fields including *Version number*, *Timestamp*, *Previous hash*, and *Merkle root*. In our scheme, the *Version number* field is utilized to document the identifier of the flying zone of the drone, and the *Timestamp* field records the block's generation time.

Smart contracts, as an added functionality to the Blockchain, are executable programs whose instances and states are stored in the Blockchain. Smart contracts allow for the automation of code execution without intermediaries and are executed in a decentralized way by the peers in the network and the results are validated via the consensus protocol.

In our scheme, we use two Smart Contracts *RegisterUAV* and *RevokeUAV*. When *RegisterUAV* is invoked, it automatically generates a special transaction adding the authentication information of a drone to the list of Registered drones (White List) allowing the drone to authenticate with the other elements of the network. On the other hand, the invocation of *RevokeUAV* rewrites the White List without including the drone's authentication information, thus disabling the authentication of the drone.

3.3.2 One-Way cryptographic Hash Functions

Hash functions are an important cryptographic primitive and are widely used in security protocols to guarantee the integrity of information. They compute a digest of a message, which represents a short, fixed-length string of bits. For a particular message, the message digest, or hash value, can be thought of as the fingerprint of a message, *i.e.,* a unique representation of a message. A small change in the input string results in a completely different output string. One-way cryptographic Hash Functions are defined in [103] as follows.

Definition 1. *A cryptographic One-Way hash function* $H : \{0,1\}^* \rightarrow \{0,1\}^n$ *is a deterministic function that takes input data of arbitrary size and produces a fixed length output string.*

A cryptographic One-Way hash function must have some properties such as:

1. The hash value is easy and fast to compute and has a low hardware implementation cost.
2. Preimage resistance or unidirectionality: for a given y no polynomial time algorithm exists for finding a value x such that $H(x) = y$.

3. Second pre-image resistance or weak collision resistance: for a given x no polynomial time algorithm exists for finding a value $x' \neq x$ such that $H(x') = H(x)$.

4. Collision resistance or Strong collision resistance: no polynomial time algorithm exists for finding two distinct values $x' \neq x$ such that $H(x') = H(x)$.

3.3.3 μTesla Authentication

μTesla improves the performance of the Tesla protocol, making it possible to implement it in devices with limited resources. μTesla is lighter because of the elimination of digital signatures, which are computationally expensive. The main idea of μTesla is to broadcast an authenticated packet through a MAC protocol and a short period of time later publish the key used to compute the MAC. In this way, it is impossible to forge the broadcast packets before the key is published.

Figure 3.1 shows an example of μTesla. First, the sender generates a sequence of secret keys (or key chain), for which it chooses the last key K_N randomly, and generates the remaining values by successively applying a one-way function H, that satisfies Definition 1. Hence, the key in the interval i can be obtained by applying $N - i$ times the function H to K_N, *i.e.,* $K_i = H^{N-i}(K_N)$. The one-way function gives the key chain the characteristic that anyone can compute in one direction, *i.e.,* $K_{j-1}, K_{j-2}, \ldots, K_0$ for a given K_j, while it is impossible to compute in the other direction, *i.e.,* $K_{j+1}, K_{j+2}, \ldots, K_N$.

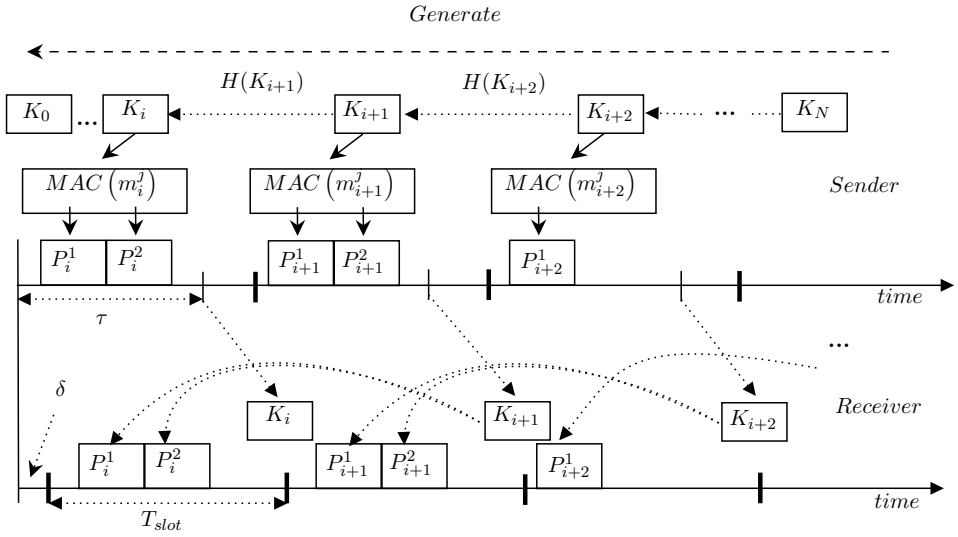

Figure 3.1: μTesla Authentication Protocol

The time is divided into intervals of equal length (T_{slot}) and the sender associates each key of the one-way key chain with a time interval. In this way, the sender uses the key K_i in the interval i to calculate the MAC code of all the packets in this interval. However, in order for the receivers to verify the MAC code of each received packet q in the interval i (P_i^q), they must eventually know the key K_i. The sender publishes K_i during the next interval after some time (τ) has elapsed. From that moment, the receivers can verify the packet P_i^q without the risk of an impersonation attack since the Sender uses in this interval the key K_{i-1} to calculate the new packet.

In order for μTesla to operate effectively, time synchronization between the sender and receivers, as well as knowledge of the key distribution schedule, are necessary prerequisites. However, μTesla does not need the strong time synchronization properties that sophisticated time synchronization protocols provide [125], but only requires loose time synchronization, and the receiver knows an upper bound on the local time of the sender. The time delay interval (τ)

utilized in the key revelation must exceed any round-trip time between the network's sender and receivers, as well as any potential synchronization error (δ).

Time synchronization is established in μTesla by a mechanism that provides strong freshness and point-to-point authentication [110]. For this purpose when joining the network each receiver sends a random nonce (N_A) in the request packet to the sender (D_{req}). The sender responds with the message ($T_S|K_i|T_i|T_{slot}|\tau$) and its corresponding MAC, which contains its actual time (T_S) (allowing synchronization), the corresponding key (K_i) of the one-way key chain for interval i, and the start time (T_i) of the interval i, the duration of a time interval (T_{slot}), and the disclosure delay (τ). Note that those last three values are sufficient to unambiguously determine the timing of the key disclosure.

In broadcast communication, the sender node may not have a pre-shared key with each receiver. In that case, the sender must send a unicast packet to each receiver with the authentication information, which brings network overhead problems. In addition, the number of keys is finite and consequently the number of packets, so a mechanism for refreshing the keys is required.

3.4 Proposed Authentication protocol

This section presents the proposed scheme, which adapts μTesla to be used in an IoD scenario. We present the considered network architecture and the processes involved in the proposed protocol, which include configuration, drone registration, communication authentication, and authentication revocation. Finally, we address the problem of limited storage in drones.

3.4.1 Network Architecture

We consider a network architecture as shown in Fig. 3.2, in which drones coexist in different flight zones. These zones may represent different domains, zones in a smart city, or sectors in a disaster zone. We assume that drones can communicate with each other and with a Ground Station (GS) through broadcast messages, as long as they are in the same zone.

Each GS is responsible for controlling and managing the drones, including the processes of registration, authentication revocation, handover management, and drone communication with the Blockchain. We assume that in each flight zone, there is at least one GS with which the drones flying over that zone can communicate, and in turn, the GSs communicate with each other through the P2P network provided by the Blockchain.

We assume drones to be limited in terms of energy, processing, and storage capacity. On the other hand, we assume that GSs are provided with much more computational and energy resources than drones and they will be the entities in charge of Blockchain storage. In this model, the Blockchain stores the information necessary for the authentication of each entity in the network, we consider a Permissioned Blockchain that anyone can access but only a registered GS can add new blocks.

3.4.2 Setup and Registration

In this phase, the public parameters for the subsequent authentication process are generated and all drones must be registered to obtain Blockchain-assisted authentication services in the respective flight zones.

System setup: In this phase each drone is prepared for a specific mission. It happens offline and without the risk that any attacker can have access to the data generated during the course of the mission. Before each mission, the user who owns a drone must generate a hashchain of length n in a similar way as in μTesla. The following procedure is used to perform this calculation.

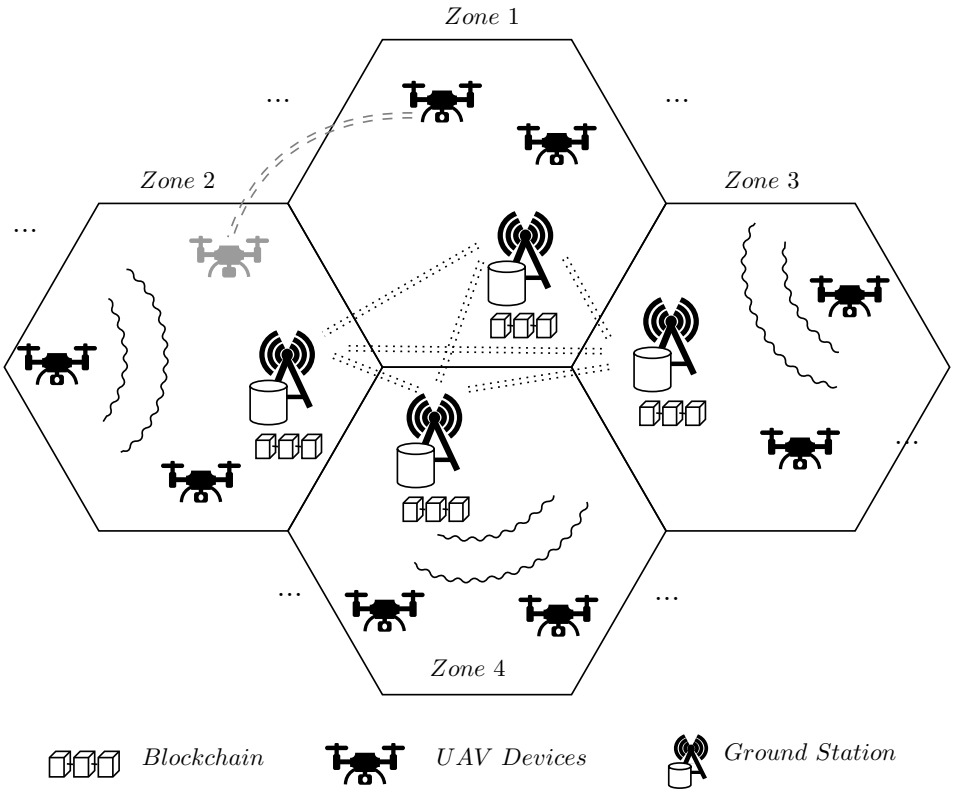

Figure 3.2: Blockchain enabled IoD Architecture

1. Choose a random number K_n of 128 bits from the set $\{1, 2, \ldots 2^{128} - 1\}$ as private key.

2. Choose a cryptographic one-Way hash function H (see Definition 1), which meets the properties discussed in Section 3.3.2.

3. Compute and store $K_i = H^{n-i}(K_n), i \in (1, n)$, where K_0 will be stored in the Blockchain doing public key functions.

Here $H^i(x)$ denotes applying i times the hash function H to the message x. Note that this process does not consume power from the drone because it can be executed on another computing medium, *e.g.,* laptop. It can also be done with the drone on the ground, where the battery charge completes again before take-off. Once the key chain has been stored in the drone and the mission has been programmed, it proceeds to register it in the Blockchain through one of the GSs.

Registration: The drone owner must receive through a secure channel a unique identifier, *i.e.,* pseudonym, from the GS and send the information generated in the Setup phase, *i.e.,* K_0 and the hash function H used to compute the hashchain. When the GS receives and validates this information, it invokes the smart contract *RegisterUAV*. As a result, the information K_0, H, and the pseudonym of the drone is sent in the *data* field of a transaction and added to the Blockchain, and the public key K_0 of the drone is added to the White List.

Once validated and included in the Blockchain of all GS, the drone can start the mission in any area. The keys of each drone can be accessed by a simple query to any of the GSs in the different zones, which is responsible for searching the local copy of the Blockchain and responding to the query.

3.4.3 Broadcast Authentication

Once the mission of each drone starts, all sent packets will be authenticated by a MAC code added to each message. As in μTesla, in each time slot i a different key K_i will be used to calculate the MAC code of all messages transmitted by the drone during this time slot. Figure 3.3 shows an example of the authentication protocol of a packet P_i sent by the drone A.

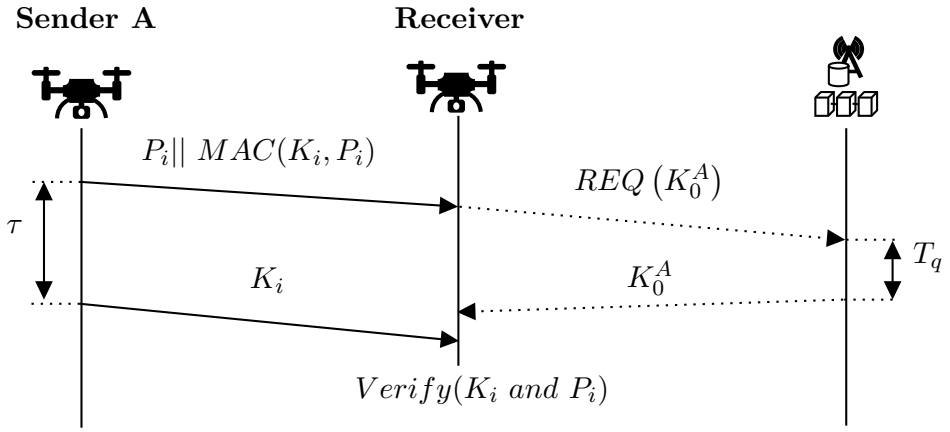

Figure 3.3: Blockchain-assisted μTesla Authentication Protocol

When a receiver R receives the packet (P_i), coming from A, with its corresponding MAC, it needs to make sure that the packet could not have been spoofed by an adversary. The threat is that the adversary already knows the key revealed for this time slot and could therefore forge the packet since it knows the key used to calculate the MAC. Therefore, the receiver needs to be sure that the sender has not yet revealed the key that corresponds to an incoming packet, which implies that no adversary could have spoofed the content. This is called the security condition, in which receivers check for all incoming packets.

In addition, each receiver must verify which key corresponds to the current time slot l, which is calculated from the last key of the drone A stored in drone R, denoted as K_l^A. The verification process is performed using Algorithm 1. In the case that some receiver has never had communication with the drone A, it must request the Blockchain through the GS of its zone, which will respond with the value of the key of A stored in the Blockchain during the Registration phase.

Algorithm 1 Authentication Algorithm

Input: t, i, K_i, K_l, P_i, MAC(K_i, P_i)
Output: **bool** A

if: $t < i * \tau$ *% Check security condition*
  Verify MAC(K_i, P_i)
  if: $K_l == F^{(i-l)}(K_i)$ *% Key verification*
    $A = True$ *% Successfully authentication*
    Store K_i *% $K_l = K_i$*
  else: $A = False$
return A

3.4.4 Drone Revocation

In case of expiration of the information provided for drones in the Registration phase or the detection of malicious behaviors of some drones, the smart contract *RevokeUAV* is invoked. Consequently, the authentication information of the corresponding drone is disabled by rewriting

the While List without including the public key of the drone. As a result, the drone will not be able to compute valid MAC codes for the messages and will lose the authentication with the others.

The White List can be publicly accessed with a simple query to the Blockchain via the GSs. In addition, when malicious behaviors have been detected, the GS proposes disabling the authentication of the drone immediately by invoking the *RevokeUAV* smart contract. Note that the GSs will record the White List (based on the consensus mechanism) into the Blockchain for complete auditing. Many existing mechanisms for intrusion detection could be applied for the automatic detection of malicious drone behaviors, some of them are studied in [126].

3.4.5 Handover and loss of packets Analysis

In our scheme, where a drone requires a handover, it does not require to be re-authenticated. The legitimate drones know the corresponding time slot number and will use the appropriate key. Since the public key is stored in the Blockchain, the GS of the new zone can respond to the queries of the receivers, and the authentication of the drone is successful.

No need for re-authentication makes the protocol efficient in case of handover events. Without loss of generality, if there is a node handover in time slot number i, it sends the next packet in the new zone after the handover in time slot number $i + j$. So, the legitimate drone will be able to use in this case the correct key K_{i+j} to calculate the MAC of the message sent, and the receivers will be able to validate the key via the Blockchain following Algorithm 1.

By the same token, thanks to the properties of the μTesla hashchain, the protocol is resistant to packet loss. If a receiver receives a packet from a drone in slot number i and then several packets containing the key to validate the MAC code of previous packets are lost, eventually, with the key of the next received packet in slot $i+j$ and thanks to the properties of the hashchain, it is possible to validate all the keys from K_i to K_{i+j} (via Algorithm 1) thus guaranteeing the authentication of all previous packets. By combining μTesla and public key storage in the Blockchain, the proposed protocol is efficient during handover events and resilient to packet loss.

3.4.6 Limited storage considerations

Since the number of packets to be transmitted in a mission and thus the length of the hashchain can be considerably large, a large amount of data needs to be stored in each drone. Even when the keys have a length of 32 bytes for a 128-bit security level, some devices may not be able to store the whole hashchain needed for a mission. We next consider the problem that the drone does not have the memory capacity required to store all the values of a hashchain of length N and can store only a number S of the values. From these stored values it must be able to reconstruct the hashchain completely and unambiguously, by successively applying the hash function. Note that applying the hash function consumes device time and power, so it is desired to minimize the number of times the hash function is applied.

Let x_u^t be the index of the key stored in memory location u at time slot number t. During time slot number i the key K_i is used. If it is not stored, the hash must be calculated from the last stored key with a subindex less than i. The value K_1 must always be stored, *i.e.,* $x_1^t = 1; \forall t$, or it would be impossible to calculate it. In turn, once the stored value has been used in the corresponding time slot, this memory space can be used. The main goal is to minimize the number of times the hash function is applied, which leads to the following optimization problem:

$$\begin{aligned} \min_{x_u^t} \quad & \sum_{u=2}^{S}\sum_{t=N}^{2}(x_S^t - t) + (x_u^t - x_{u-1}^{t-1}) \\ \text{s.t.} \quad & x_u^{t+1} \leq x_u^t \\ & 1 \leq x_u^t \leq N \\ & 1 \leq u \leq S; 2 \leq t \leq N \end{aligned} \tag{3.1}$$

Note that the first term of the objective function is how many times the function should be applied in slot t given that the nearest value is x_S^t and the second term is how much it costs to store a more convenient value in the already used, *i.e.,* available, memory slots.

The problem can be solved using an integer linear programming algorithm, but it would be computationally expensive and therefore impractical for drones. We propose an algorithm (shown in Algorithm 2) for dynamic memory management, which allows for improving memory utilization and thus minimizes the computational cost of the proposed scheme.

The idea of the algorithm is to start from the storage of S keys evenly spaced among the N keys of the hashchain and update the memory location with the largest index of the stored keys, *i.e.,* x_S^t. The value to be stored will be equidistant between x_S^t and x_{S-1}^t (as shown in Algorithm 2) according to Theorem 1. Let us define the distance function $\mathbb{D}$ between two stored values as the number of times the hash function must be applied to compute the other value, for example between $\mathbb{D}(K_{10}, K_4) = 6$ because $K_4 = H^6(K_{10})$.

Algorithm 2 Key storage Algorithm

Input: $N, S, t, [x_1^{t-1}, x_2^{t-1} \dots x_S^{t-1}]$
Output: x_S^t

if $(x_S^{t-1} == t)$ #Update storage
  if $(\mathbb{D}(x_{S-1}^{t-1}, x_S^{t-1}) > 1)$ #not at distance one
    $x_S^t = \left\lfloor \frac{x_S^{t-1} - x_{S-1}^{t-1}}{2} \right\rfloor$
  else
    $x_S^t = x_{S-1}^t$
    $x_{S-1}^t = \left\lfloor \frac{x_{S-1}^{t-1} - x_{S-2}^{t-1}}{2} \right\rfloor$
else
  $x_S^t = x_S^{t-1}$ #Keep the same value
return x_S^t

A toy example could be when a hashchain has a length of seven ($N = 7$) and only two values can be stored ($S = 2$). Following the proposed algorithm, in the beginning, K_1 and K_4 are stored. In the first slot, key K_7 is required but is not stored and must be calculated from K_4, via applying three times the function H, $\mathbb{D}(K_4, K_7) = 3$. The next two slots require K_6 and K_5 respectively, which are not in memory and are calculated from K_4 by applying twice and once the function H, respectively. Subsequently, K_4 is used, which is already in memory (zero cost). In the next slot, K_3 is used, which is not in memory and should be calculated from K_1, for which it is necessary to calculate $K_2 = H(K_1)$. At this moment the value of K_2 is stored in the memory space where K_4 was stored. Finally, in the following slots, K_2 and K_1 are required, which are in memory and have zero cost. Therefore, a total of 8 times the calculation of function H is required. This means that one less time is required than if the allocations were static, *i.e.,* without reusing the memory spaces. As the values of N and S increase, the gain is higher as verified later in Section 3.6.8.

Theorem 1. *If the distance between two consecutive stored values x_i^t and x_j^t, with respective*

*indexes i and j, is equal to d at time t, (*i.e., $\mathbb{D}(K_i, K_j) = d$*), then the value x_y^{t+1} to be stored, which minimizes the number of operations to reconstruct the hashchain, is when $y = \lfloor \frac{d}{2} \rfloor$.*

Proof. It is impossible to use the value x_j^t to calculate x_i^t because the problem of calculating the inverse of the hash function is intractable. Therefore the hashchain is reconstructed from the stored value with a lower index x_i^t. Then the total number of operations T required to reconstruct the hashchain is given by:

$$T = \sum_{t=i}^{y} \mathbb{D}(x_i^t, x_y^t)) + \sum_{t=y}^{j} \mathbb{D}(x_y^t, x_j^t)) \tag{3.2}$$

For simplicity and without loss of generality we use $i = 1$ and $j = d$, as $\mathbb{D}(x_t^i, x_t^x) = y - i$ then:

$$T = (1 + 2 + \cdots + y) + [(d - y) + (d - y - 1) + \cdots + y] \tag{3.3}$$

$$T = \frac{y(y+1)}{2} + \frac{(d-y)(d-y+1)}{2} \tag{3.4}$$

Solving $\frac{\partial T}{\partial y} = 0$, *i.e.,* $2y - d = 0$, results in $y = \frac{d}{2}$ and with $\frac{\partial^2 T}{\partial y^2} = 2$ (positive), it is the minimum value.

□

3.5 Security Analysis

This section provides a security analysis of the proposed scheme, analyzing the resilience to potential attacks and providing a formal security analysis. It also develops the analysis of compliance with the requirements for an authentication protocol addressed in Section 3.2.1. A comparison with existing authentication solutions for IoD networks is also provided.

3.5.1 Attack Model

In the proposed Blockchain-based authentication scheme, the following assumptions are applied to analyze the security against existing attacks.

1. The private Blockchain is jointly maintained by the authorized GSs in the whole network so that the transaction and smart contract data are transparent to all participating GSs.

2. Attackers can intercept communication data and compromise system security by impersonating drones or GS. In the worst case, for example, the drone could be taken over and its data acquired.

3. We consider the Dolev Yao (DY) model, which involves communications over an insecure channel and an untrusted nature between the parties. Thus, an attacker can eavesdrop, modify, and replay messages exchanged over the public channel. With the illicitly acquired information, the attacker can also obtain some ephemeral secrets including secret keys.

Based on the attack model described above the possible attacks that the protocol must resist are listed and analyzed below.

Eavesdropping Attack: An adversary can record all the messages exchanged during the communication in any flight zone and coming from both drones and GSs. This means that the attacker can obtain any message with its respective MAC code and the messages containing the keys used in the previous slots. Suppose that the packet P_i is captured with its respective MAC ($MAC(K_i, P_i)$) and then the keys K_i, which are broadcasted in another packet τ seconds

later. Note that if the message is modified using the K_i keys, no receiver will accept the spoofed packet because the temporary key security condition is not met. On the other hand, it could try to calculate the key K_{i+1} that should be fresh. For this, besides having a short time, it would have to calculate the inverse operation to the hash function H since $K_i = H(K_{i+1})$. Note that the probability of success for a brute-force attack to try to compute the inverse hash function is very low. For example, for the SHA-256 hash function, due to birthday attacks, 128-bit security level is guaranteed and eavesdropping attacks can be resisted.

Drone Identity impersonation: If an attacker wants to impersonate a drone, it must generate valid MAC codes for messages with the correct key corresponding to the current time slot. However, due to the intractability of calculating the inverse of the hash function, the impersonation attack can be prevented with this scheme.

Drone Identity forgery: An attacker who wishes to generate an illegal MAC code in a packet to fool the verifying party, must use the key corresponding to the current time instant, which results in solving the intractable problem of calculating the inverse of the hash function.

Drone Cloning Attack: An attacker could attempt to clone a drone if he knows the physical address and IP of the drone, but he would not be able to access the hashchain generated in the Setup and Registration process and therefore would not be able to impersonate the identity of the drone. As a consequence, resistance against the Drone Cloning attack is obtained.

Ephemeral Secret Leakage Attack: The attacker can guess one of the keys that will be used in one of the time slots. With this, he could generate valid MAC codes during this time interval. In the next time interval, to generate valid MAC codes he must calculate the next key, for which he would need to solve the intractable problem of calculating the inverse of the hash function. However, the probability of guessing the key in an instant of time is very low given the large size of the search space.

Man-in-the-Middle Attack (MITM): An attacker could secretly relay and possibly alter communication between drones and GS or other drones. However, as discussed above, it would be impossible for an adversary who does not know the valid key to be used in the current slot to generate valid MAC codes. Thus the scheme is not susceptible to MITM attacks.

Denial of Service Attack: Because authentication in μTesla is delayed for some time, receivers appear vulnerable to flooding attacks that can cause excessive packets in the buffer, even if they are eventually unauthenticated. In [127] some requirements that guarantee security against DoS attacks are shown. In the proposed scheme we employ the same mechanisms described in [127] to eliminate the risk of a DoS attack which includes not reusing keys and checking that when a packet arrives if the key index is incorrect, it should not be buffered.

3.5.2 Formal Security Analysis

We use a formal method to prove the security of the proposed protocol. The message exchange is modeled by ProVerif [128], which is designed for the analysis of secrecy and authentication properties as well as additional properties, such as privacy, traceability, and verifiability. It has been used for the validation of a large number of protocols [129], [130].

Figure 3.4 shows the ProVerif simulation code for the proposed scheme. The *ver_key* function verifies that the security condition holds and that the key corresponds to the current time slot number. Figure 3.5 shows the results of the simulations in Proverif. As can be observed, the private key of every slot (*ki* in the code) will not be obtained by the adversary, which formally evidences the security of the proposed scheme.

3.6 Performance Analysis

In this section, we present an analysis of the main performance parameters on which the proposed scheme has a direct influence and which are desired to be optimized in an IoD authen-

```
type key .
type timeslot .
free c1: channel.
free c2: channel[private].
free n,ki:bitstring[private].
free slot:timeslot.
free pk:key.

(*---Cryptographic functions---*)
fun H(bitstring):bitstring. (*One-way Hash function*)
fun MAC(bitstring,key):bitstring.
reduc forall m:bitstring, k:key ; getMessage(MAC(m,k)) = m.

(* functions for generating and verifying keys *)
fun gen_keys(bitstring,key) : bitstring.
fun get_key(bitstring, timeslot): key.
fun ver_key(key,timeslot,key): bitstring.

(*---Queries---*)
query attacker (ki).

(* Drone A*)
let UAV_Tx(sk:key, Hashchain:bitstring)=

    let ki = get_key(Hashchain,slot) in
    new m:bitstring;
    let HMAC = MAC(m,ki) in
    out ( c1 ,(HMAC,m) );
    out ( c2 ,ki ).

(* Drone B*)
let UAV_Rx(pk:key) =
    in (c1,(ki:key, HMAC:bitstring)) ;
    let Valid = ver_key(ki,slot,pk) in
    let M = getMessage(HMAC) in
    0.

process
new sk : key ;
let Hashchain = gen_keys(n,sk) in
((!UAV_Tx(sk,Hashchain))|(!UAV_Rx(pk)))
```

Figure 3.4: Proverif code for the proposed scheme.

```
Verification summary:
Query not attacker(ki[]) is true.
```

Figure 3.5: Proverif simulation results of the proposed scheme.

tication protocol. First, the computational cost and network overhead caused by the proposed scheme are analyzed. In addition, the authentication delay is estimated via simulation when the number of drones and the number of flight zones are increased.

3.6.1 Experimental settings

To evaluate the computation and communication costs of the drones, the computation time of the cryptographic operations involved in our scheme is measured on a Raspberry Pi microcomputer, which has hardware properties similar to a drone. A Raspberry Pi 3B configuration consists of a Broadcom BCM2711, Quad-core Cortex-A72 (ARM v8) 64-bit SoC @ 1.5 GHz, 8G RAM, and 64G ROM. For encryption, we use the *pycrypto* library (version 2.6.1) written in Python(version 3.8) language. We use AES128 as the symmetric encryption algorithm, SHA256 as the general hash function, and the code defined in [131] as MAC code, which is computed using Equation 3.5.

$$MAC(K, m) = H((K' \oplus opad) \parallel H(K' \oplus ipad) \parallel m)) \tag{3.5}$$

Here *opad* is the block-sized outer padding, consisting of repeated bytes valued $0x5C$. The variable *ipad* is the block-sized inner padding, consisting of repeated bytes valued at $0x36$. K' is a block-sized key derived from the secret key, K, either by padding to the right with zeros up to the block size or by hashing down to less than or equal to the block size first and then padding to the right with zeros. Like in [106], we use 64 bytes and 384 bytes for the operations on the additive cyclic groups, $\mathbb{G}_1$ and $\mathbb{G}_T$ respectively. Both groups have order q, where q is a large prime number of length 128 bits. The time cost of the average of 1000 executions for each of the cryptographic operations analyzed in this Chapter is shown in Table 3.2. For a confidence level of 95%, this number of runs ensures that the maximum error in the estimation of the metrics is less than 3%.

Table 3.2: Time cost of different cryptographic operations

Notation	Description	$Time$(ms)
T_{hm}	Hyper-elliptic curve multiplication	9.899
T_H	SHA256 Hash function	0.026
T_S	AES128 encryption and decryption	1.975
T_m	Scalar Multiplication in $\mathbb{G}_1$	0.031
T_e	Exponentiation in $\mathbb{G}_T$	7.682
T_{bp}	Bilinear pairing in $\mathbb{G}_T$	8.128
T_{MAC}	MAC Code	0.053

3.6.2 Computational Overhead

To estimate the computational overhead of the proposed scheme, a theoretical analysis of the most time-consuming operations is performed, neglecting lighter operations such as string concatenation and XOR. Table 3.3 shows the results of the proposed scheme for packet authentication and packet verification at one of the receivers. A comparison with existing authentication solutions is also shown, the works used in the comparison were selected based on their good performance in this type of scenario. In our scheme, to authenticate a message, which may or may not have been previously encrypted, it is required to calculate the MAC code of the message (T_{MAC}) by the transmitting drone. To verify the key, it is required to calculate the hash of the previous key (T_H). So, in total ($T_H + T_{MAC}$) is required to authenticate the message. Note

that the previous encryption of the message is not essential for authentication, which could be achieved even if the original message is in plaintext. Thus, if we consider the time to encrypt and decrypt (T_S) in the proposed scheme, a total of ($T_H + T_{MAC} + T_S$) is required. In the third column of Table 3.3, the computation time required to complete authentication, for a security level of 128 bits, is provided.

Table 3.3: Computation and communication cost

Schemes	Time Complexity	$Time$(ms)	Comm.($bits$)
[103]	$12T_{hm}$	118.788	1280
[104]	$23T_H + 2T_S$	2.548	2304
[106]	*	12799.200	16912
[109]	$2T_H + 5T_S$	4.927	7168
Ours	$T_H + T_{MAC} + T_S$	1.054	1024

* $14T_H + 14T_m + T_e + 3T_{bp} + 3T_S$

3.6.3 Communication overhead

Since the message length varies depending on the task and application, only the payload related to authentication is calculated. The communication overhead is calculated based on the total number and size (in bits) of messages used for authentication. Note that the MAC code to be added to each message is 256 bits long if SHA256 is used as the hash function. In addition, 256 bits are needed to send the key, thus 512 bits are needed to authenticate a packet. The last column of Table 3.3 (Comm.) shows the total number of bits required by some existing solutions. Our protocol requires little information to achieve authentication due to the lightness of the HMAC code.

3.6.4 Blockchain latencies

The latencies of the different smart contracts involved in the protocol are estimated. For this purpose, a Blockchain network is implemented using Hyperledger Fabric (v2.2) in a Docker environment (v20.10.6). The network consists of five organizations that function as validators and three peers in each organization. The validators process, store, validate transactions, and add new blocks to the Blockchain using the consensus mechanism, as well as handle queries from peers. Hyperledger Caliper(v0.4.2) is installed as a benchmark to obtain the average delays for query and writing of 300 independent invocations to the smart contracts (chaincodes in Hyperledger Fabric).

Write latency is the duration from the time the write operation code is invoked to the time the data is uploaded into the Blockchain. The query latency is the duration from the time the read operation code is invoked to the time the result is returned. Note that the query operation does not require transaction validation and consensus, and the data can be queried from a local copy of the Blockchain. Table 3.4 shows the result of the minimum, maximum, and average time of the main operations of the smart contracts and the queries to the Blockchain. The results are consistent with the average block generation time of $2s$ level established for the implementation. For this, 10^4 runs of each operation were performed through the API included in the Hyperledger Fabric Python SDK[1].

We have estimated these parameters using the Practical Byzantine Fault Tolerance (PBFT) consensus mechanism with a transaction size of $64kB$, which is sufficient to transmit key in-

[1]available: https://github.com/hyperledger/fabric-sdk-py

Table 3.4: Time cost of Blockchain operations (s)

Operation	Min	Max	Average
Invoke RegisterUAV	1.342	1.721	1.418
Invoke RevokeUAV	1.541	1.726	1.692
Query for White List	0.364	0.621	0.481
Query for UAV	0.127	0.182	0.150

formation and update the flight zone of the handover drones. Both the MATLAB(R2018) simulations and the previous experiments with Hyperledger Fabric were run on a laptop with Intel Core $i7-7700$ CPU 3.60GHz×8, and 16GB of RAM.

3.6.5 Implementation and Performance Evaluation

In this section, we present the results of the simulation of the proposed authentication scheme for a varying number of zones and drones in the network traffic, as well as for different levels of instability of the wireless links. For this purpose, several scenarios are simulated in which the influence of the packet arrival rate of each drone, the key disclosure time, and the probability of packet loss on the average authentication time of each packet and the network throughput are varied. The performance metrics are defined as follows:

- Average message authentication delay (A_d): This corresponds with the average time it takes to authenticate a message, from the time a message is received (T_r^i) until the information to verify the MAC code and the key used to calculate it is obtained at the receiver (at time T_a^i). Denote by n_p the total number of packets, then the average authentication delay is calculated as:

$$\bar{A_d} = \sum_{i=1}^{n_p} (T_r^i - T_a^i)/n_p \tag{3.6}$$

- Average Authenticated Throughput (T_p): is defined as the average amount of information per unit of time exchanged in the authenticated packets and is calculated as follows:

$$T_p = n_p * \left|p^i\right| / T_S \tag{3.7}$$

For the analysis of the performance of the proposed scheme, a discrete event simulator is developed in MATLAB(R2018), using Table 3.3 values for the processing times on the drones. Table 3.5 shows the simulation setup of the experiments. Cross-domain device authentication is achieved using smart contracts. To measure the efficiency of key management, the latency of the smart contract operation is recorded.

3.6.6 Average Authentication delay

Figure 3.6 shows the results of the simulations for different scenarios, in which the number of zones (z) and the density (ρ) of drones per zone, *i.e.,* the ratio between the total number of drones and the number of zones, are modified. We here consider four scenarios, corresponding to the values 5 and 10 for ρ and z. For each of these scenarios, we modified the time in which the drone discloses the key (τ), taking in each case a value between 20 ms and 100 ms with an increment of 20 ms.

Table 3.5: Simulation setup

Parameter	Value
Simulation time	30 minutes
Zones (GS)	$[5, 10]$
Drones/Zone (ρ)	$[5, 10]$
Mobility model	Random
Average Block time	2 seconds
Transaction size	64kB
Packet size	512 Bytes
Data Rate (802.11b)	11 MBps

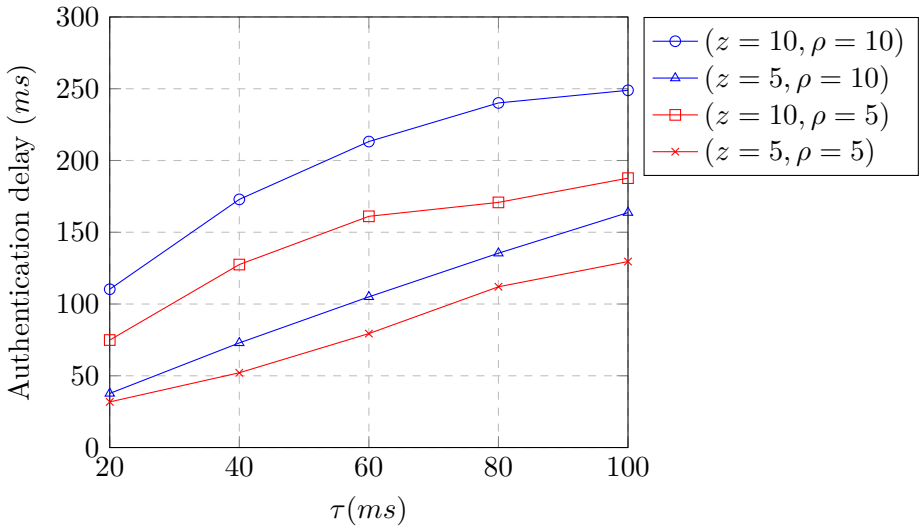

Figure 3.6: Average Authentication time for different key disclosure times.

As shown in Fig. 3.6, in all scenarios, the average time required to complete the authentication is higher with an increase of τ. This is due to the fact that the receivers must wait longer to receive the key with which the MAC code of each message was calculated.

In addition, for scenarios with equal node density, when the number of zones increases, the authentication time increases because the number of drones will be higher and a higher number of queries to the Blockchain will be required when a previous key is not stored.

Similarly for scenarios with an equal number of total drones, for example, $(z = 10, \rho = 5)$ and $(z = 5, \rho = 10)$, both having 50 drones in total, the scenario where there are fewer zones, *i.e.,* $(z = 5, \rho = 10)$, presents better performance in terms of authentication time. This is because with more zones there is a higher probability that the drones did not have previous communication and do not have a previous key. As a consequence, a query to the Blockchain is necessary and thus requires additional time. To conclude, for τ values below 100 ms, the authentication time does not exceed 250 ms for each of the four scenarios, which is useful for most drone applications.

Impact of packet loss

Since μTesla is resistant to packet loss due to the hashchain characteristics of the keys, the impact of packet loss on the average authentication time is studied. To achieve this, the channel stability is varied by modifying the packet loss probability (p_L), taking values between 0 and

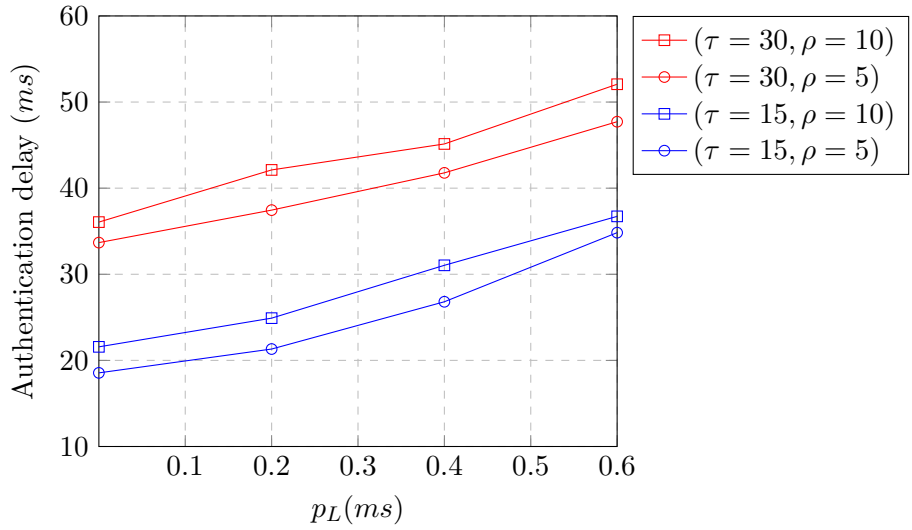

Figure 3.7: Average Authentication time for different packet loss probabilities (p_L) in a network with 10 zones.

0.6 with increments of 0.1 in scenarios with 10 zones and different densities (5 and 10), as well as modifying the value of τ, with values 15 (blue lines in the Fig. 3.7) and 30 (red lines in the Fig. 3.7), respectively.

Figure 3.7 shows that as P_L increases the average authentication time increases, which is due to the loss of packets. It takes longer for the receivers to receive the respective keys which allows verifying the authenticity of the received packets. On the other hand, for a given value of τ, the higher the density of the network, the higher the number of drones and therefore the higher the authentication time. Although the increase in the probability of packet loss increases the authentication time, the protocol is able to authenticate messages with keys received subsequently, which makes it resilient to the loss of messages containing the keys to verify authentication.

Impact of network traffic

An important parameter to consider in the performance analysis of every protocol is network traffic. Figure 3.8 shows the results of several simulated scenarios in which the packet sending rate varies. We consider again four scenarios with the number of z zones and density ρ equal to $\{5, 10\}$. The packet generation of each drone follows a Poisson distribution with rate parameter λ and the probability of observing k events in a time period is calculated using Equation 3.8. In the simulated scenarios, the value for λ ranges between 10 and 50 in increments of 10 packets per second.

$$p(X = k) = \frac{\lambda^k e^{-\lambda}}{k!} \tag{3.8}$$

As shown in Fig. 3.8, as the network traffic increases, the average authentication time increases because the drones must process more packets for authentication. Note also that for two scenarios with the same number of total drones, for example, scenarios $(z = 10, \rho = 5)$ and $(z = 5, \rho = 10)$, the scenario where there are fewer zones present a worse performance in the authentication time for values of 40 packets/s or less, but from this value, the scenario with more zones present worse performance. This behavior is due to the fact that with more areas it is more likely that queries are made to the Blockchain because the receivers are less likely to store a previous key from a given transmitter.

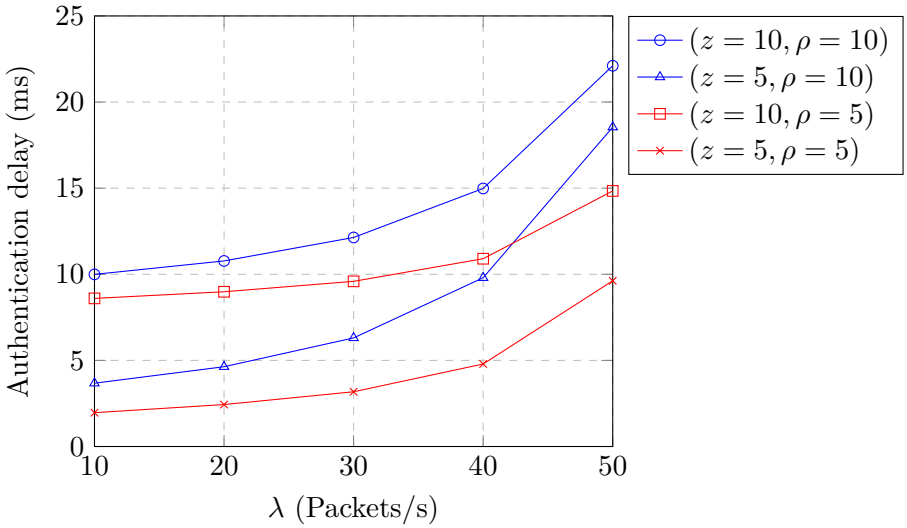

Figure 3.8: Average Authentication time for different arrival rates.

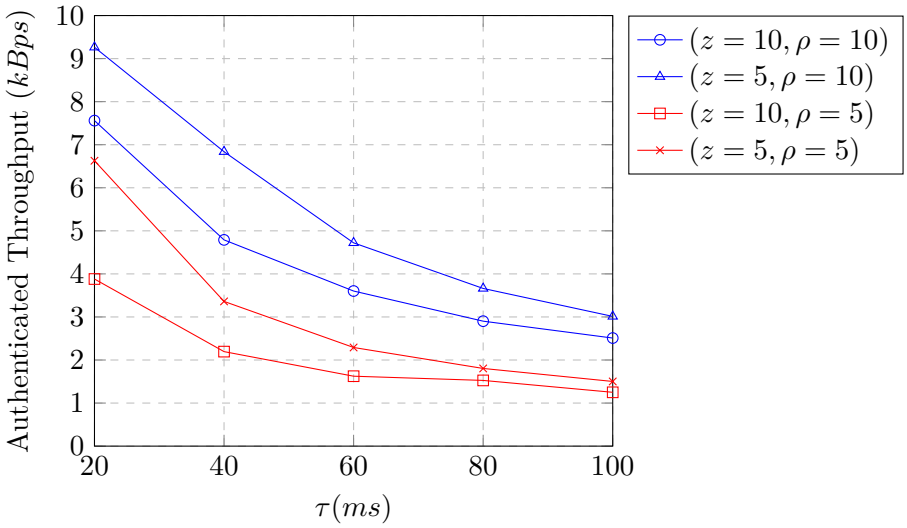

Figure 3.9: Average Authenticated Throughput for different key disclosure times.

3.6.7 Average Authenticated Throughput

Finally, we analyze the behavior of the Average Authenticated Throughput (T_p) in several scenarios in which the time to discover the key (τ) is varied. Again, we consider the same four scenarios with the number of zones $z \in \{5, 10\}$ and the network density $\rho \in \{5, 10\}$. The value of τ changes in the range of 20 to 100 with increments of 20 ms. As seen in Fig. 3.9, as τ increases the T_p of the network decreases in all scenarios.

This is because more time is required before authentication of the message becomes possible, affecting the denominator in the calculation of the T_p. Note also that for the same density in the network, for example, scenarios $(z = 10, \rho = 5)$ and $(z = 10, \rho = 5)$, the scenario with a higher number of zones, *i.e.,* $(z = 10, \rho = 5)$, has worse performance in terms of authenticated Throughput. This follows from the fact that as shown above, the increase in the number of zones worsens the authentication time and therefore the amount of information authenticated per unit of time decreases. These results allow the choice of a value of τ that guarantees that a secure condition is maintained with the maximum possible T_p in the network.

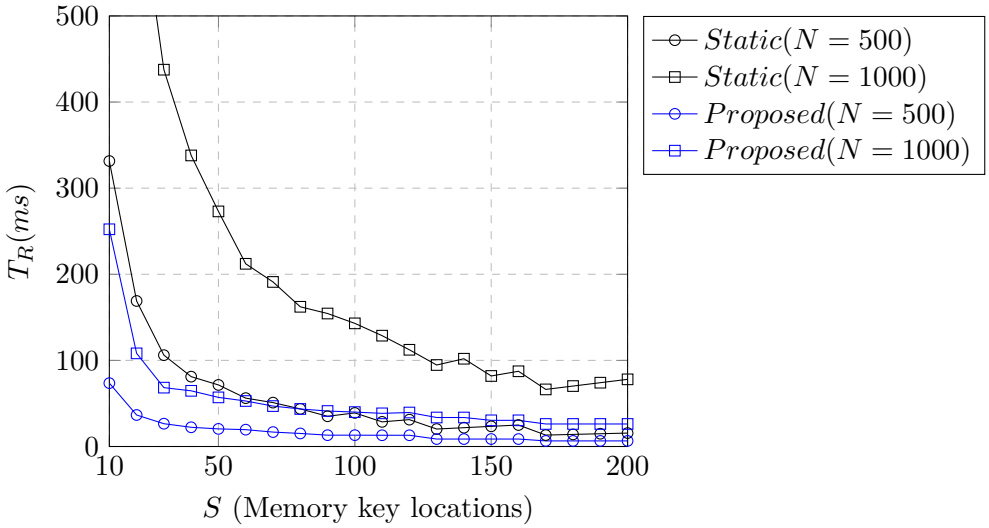

Figure 3.10: Number of operations to reconstruct the hashchain for different memory availabilities.

3.6.8 Storage limited scenario

Although μTesla requires little storage (2Kb) of the program and 32 byte keys for 128 bits security levels, in our proposal it is necessary to store multiple keys considering the total number of packets sent in a mission. In some cases, this may result in insufficient memory available in the drone to store the entire hashchain. In this section, results are presented for scenarios where drones cannot store the entire hashchain and can only store S keys of the total length N of the hashchain. To evaluate the performance of the proposed memory management algorithm, the time required to reconstruct the hashchain from the S values stored in memory is determined. For this purpose, the number of times the hash function is applied when reconstructing the hashchain is calculated and multiplied by the time required to complete a hash operation. To this end, we consider the computed time (T_H) for a Raspberry Pi 3B, shown in Table 3.2.

Figure 3.10 shows the results in terms of the Key Reconstruction delay, denoted as T_R, which is applied in order to reconstruct the entire hashchain of length 500 and 1000, which is realistic for a flight of 40 minutes, allowing sending more than 2 and 4 packets per second respectively. In each case the amount of available memory, *i.e.,* S, is varied, taking values between 10 and 200 with increments of 10. We include the comparison with scenarios where static values are stored (without reusing the memory locations). The results of the static and proposed algorithm are shown in the graph with black and blue curves respectively.

As shown in Fig. 3.10, as the amount of available memory space increases, the time required to rebuild the hashchain decreases due to the decrease in total number of required hash operations. At the same time, in all scenarios, as the total length of the hashchain increases (from 500 to 1000) the number of operations required to reconstruct the hashchain increases for an identical amount of available memory.

On the other hand, in all simulated scenarios the proposed solution matches the obtained optimal solution and is better than the solution with static memory utilization, with a minor improvement as the amount of available memory increases. For example, for values of S equal to 50, the proposed algorithm requires 2.5 times fewer operations than if a static allocation is used, but for $S = 200$ the improvement is 23%. These results show that the proposed algorithm allows efficient use of the available memory in the drone in case the storage capacity is very limited.

3.7 Conclusions of the Chapter

Ensuring security and privacy in IoD networks is critical to protect data from cyber-attacks. However, establishing secure authentication is challenging because drones are power-limited devices with high mobility. Typical authentication solutions present several drawbacks due to their centralized architecture. We use Blockchain technology to address the centralization problem and contribute to improving the security level. We provide a decentralized, secure, and very lightweight authentication protocol based on μTesla and supported by Blockchain to store, manage, and control the information needed to authenticate drone-to-drone communication. We address the packet loss problem in broadcast authentication, which is a novelty considering the existing works. Simulation and security analysis results show that the proposed solution outperforms other recent authentication solutions while ensuring security.

Chapter 4

Blockchain-Based Group Key Management Scheme for IoT with Anonymity of Group Members

4.1 Introduction

The Internet of Things (IoT) is a swiftly evolving technology that interconnects an extensive array of physical devices. These devices are capable of exchanging data for diverse applications spanning from smart homes/cities, industry, and agriculture to public safety applications. However, for the sustainable development of these applications, it is imperative to guarantee the security and privacy of the information being exchanged, along with ensuring adequate quality of service (QoS) levels [132].

Group communication enables multiple devices to exchange information efficiently through broadcast transmission over insecure channels. Nonetheless, the security and privacy of group communication in IoT environments are major challenges due to the open nature of wireless communications, aggravated by the resource limitations of most IoT devices in terms of energy, and processing and memory capabilities [133]. Basic security services such as data privacy, integrity, and authentication are required in group communications. A secure, robust, and efficient group key management is the base for guaranteeing these services.

A key challenge in group communication is the presence of a malicious node within the group. Upon detection of such a node, it becomes necessary to immediately destroy and reestablish all group keys in which this node is involved. This process demands high efficiency, considering the significant computational and energy costs associated with the creation and establishment of group keys.

In this context, if the members of the group need to be anonymous, the challenge is more significant. Anonymity is often required to offer additional privacy, for instance in healthcare, where a group of nodes is used to monitor patients. A leakage of the identity of the device could be detrimental, as it could potentially reveal the disease of the patient, severely compromising their privacy.

4.1.1 Research Motivation and Contribution

Most existing key management solutions are based on a centralized architecture, where a central entity is responsible for generating, distributing, and updating group keys. However, centralization poses challenges such as scalability limitations and the single point of failure problem. These problems have sparked interest in decentralized solutions in recent years. Blockchain technology, with its tamper-proof ledger, transparent transactions, and secure operations within a decentralized network, offers an attractive solution to address the challenges of group communication [134].

This chapter introduces a group key management scheme enabled by blockchain technology while ensuring device anonymity inside the group. The proposed scheme utilizes blockchain to facilitate secure group formation and key establishment processes between a Service provider (SP) and IoT devices. Additionally, the public keys of nodes and groups are stored in the

blockchain without reference to the real identity of the nodes. The use of Smart Contracts in the Blockchain allows the proposed scheme for the immediate and automatic revocation of group keys in case malicious nodes are detected.

The main contributions of the present protocol are summarized below:

- Our approach utilizes Blockchain technology to facilitate group key management. Specifically, we establish an asymmetric group key among the nodes within the group, enabling secure communication between outsiders and the group via the public key of the group while keeping the anonymity of nodes inside the groups. Furthermore, we employ lightweight cryptographic operations, resulting in a computationally lightweight protocol.
- We conduct a security analysis of the proposed scheme and show its resistance to attacks that commonly occur in group communications. Moreover, we compare our proposed scheme with other existing works in the literature.
- In addition to performing security analysis, we evaluate the computation and communication overhead of the protocol via simulation. We also studied the impact of Blockchain technology integration on protocol performance.

4.2 Related Works

4.2.1 Group communication requirements

We study system requirements for group communication in IoT. According to [135]–[137], a scheme for secure group communication should meet the following security and performance requirements.

Security requirements: The system performs encryption for data security in group communication. In addition to classic security services including the privacy, authentication, and integrity of the information, group key management must satisfy forward and backward secrecy. Forward secrecy ensures that when a node leaves a group, it will not have any access to the future key. Backward secrecy implies that when a new node joins a group, it will not have access to the previous information and keys exchanged in the group.

In addition to these requirements, we consider the anonymity of the group nodes, the possibility of communication between an outsider and the group [138], the non-repudiation property for the group and the individual members, and the possibility for a member to belong to many groups simultaneously.

Performance requirements: In group communication, when group members join and/or leave the group, the group key must be changed. If this process frequently occurs, the computation and communication overhead increases. Therefore, the scheme should minimize the overall costs by reducing the frequency of updating the group key and the computational and energy cost of updating the keys to improve efficiency, especially in IoT networks given the limitations of most devices.

4.2.2 Group Key Management existing solutions

Numerous solutions for group key management have been proposed to meet the group communication requirements. Existing solutions, given their architecture, fall into one of the following classifications: Centralized and Decentralized.

In centralized solutions, there is usually a single entity (Key Distribution Center (KDC)) to control the entire group [139]. The KDC is responsible for key generation, distribution, and management of the keys. Conventional centralized solutions have scalability issues as the network scales up. In addition, they often have a single point of failure, being susceptible to disruption in the event of denial of service attacks (DOS) or technical failures [140].

Given the drawbacks of centralized solutions, several decentralized GKM solutions have been proposed in the literature for IoT networks [135], [136], [141], mostly using blockchain to achieve decentralization. Recently, blockchain-based solutions for key management for group communication have been proposed for many applications, including UAV [142], [143], VANET [144], [145] and data co-auditory [146].

In [141] authors propose a blockchain-based authenticated group key agreement protocol for IoT. In that scheme, an asymmetric key system is used to build a symmetric group key. In [136] an asymmetric group key agreement protocol supported by blockchain is proposed for the Industrial Internet of Things (IIoT). This scheme realizes fine-grained access control while ensuring that user identity information is not leaked, and reduces resource consumption through the attribute-based multi-signature.

A Diffie-Hellman Elliptic Curve group key agreement supported by blockchain is introduced in [135]. In this scheme, a smart contract acts as a group controller in a two-round protocol. In the first round, the smart contract generates two-party shared keys with each group member. The smart contract calculates partial group keys in the second round and sends them to the respective group members. Once the partial group keys have been received, each group member generates a symmetric group key by multiplying the product received by its shared key.

In the method proposed in [147], each group member only needs to authenticate the left neighbor once to complete authentication and only the neighbor at the left needs to be updated when some node joins or leaves the group. Blockchain stores group identities, a list of group members, and some parameters related to group members, which can solve the problem of single-node failure.

An asymmetric group key management scheme based on elliptic curve cryptography is introduced in [137]. The solution uses anonymous authentication to prevent information disclosure and protect privacy. This scheme includes a Key Generation Center allowing the verification of the identity of the terminal entity for tracking the identity of anonymous members in the blockchain.

In [148] a blockchain-based solution is proposed for constrained devices. This scheme succeeds in constructing a symmetric group key while migrating the communication and verification overhead of participant key management to a Hyperledger Fabric blockchain platform. In fact, the solutions proposed in [148], [137], and [147] could be adapted for being applied in an IoT group communication scenario.

Based on the above literature review, we note that some group key management mechanisms have been designed for IoT networks. Most of them take into account the privacy of the participants and successfully resist different attacks caused by internal or external attackers while pursuing to be computationally efficient and suitable for resource-constrained devices. However, not all of these solutions meet all the security requirements mentioned in Section II.A.

Table 4.1 presents a summary of the blockchain-based approaches recently proposed for group key management described earlier. Our scheme considers multiple users and groups, where users can create groups of nodes, and a node can belong to multiple groups. The use of unlinkable pseudonyms guarantees the anonymity of nodes within groups while enabling secure communication with nodes that do not belong to the group. Non-repudiation of group and individual members is ensured through the presence of individual and group public keys. Additionally, efficient group communication is maintained through the use of elliptic curve cryptography (ECC) and a symmetric key. Our approach leverages Smart Contracts to facilitate cascading revocation and automatic notification of broken group keys. In contrast to existing work, our approach satisfies all the analyzed requirements, which is verified in Section 4.5.

Table 4.1: Comparison of key group management approaches

	[141]	[136]	[135]	[147]	[137]	[148]	Ours
R1	●	●	●	●	●	●	●
R2	●	●	●	●	●	●	●
R3	●	●	●	●	●	●	●
R4	○	○	●	●	○	○	●
R5	○	●	○	●	○	○	●
R6	○	○	○	○	○	○	●
R7	○	○	○	○	○	○	●
R8	○	○	○	○	○	○	●

● Provides ○ Does not provide

Requirements: **R1:** Blockchain-based, **R2:** Forward Secrecy, **R3:** Backward Secrecy, **R4:** ECC-based, **R5:** Asymmetric Group Key, **R6:** Anonymity in the group, **R7:** Automatic notification of broken group keys **R8:** Multiple groups.

4.3 Preliminaries

4.3.1 Public key based operations

Our scheme is based on Elliptic Curve Cryptography (ECC), which is currently the most efficient public key-based solution. Security in ECC is based on the Elliptic Curve Discrete Logarithm Problem (ECDLP), which is defined in Def.2. It is assumed that it is computationally hard for any algorithm to solve the ECDLP in polynomial time.

Definition 2. *Let E be an elliptic curve defined over a finite field $\mathbb{Z}_p$, and let $P \in E(\mathbb{Z}_p)$ be a generator of the cyclic group $\mathbb{C}$ of order n. Given $Q \in \mathbb{C}$, the Elliptic Curve Discrete Logarithm problem is to find the integer $l, 0 \leq l \leq n-1$, such that $Q = l\,P$.*

As a consequence, the pair (d, Q_d) represents the private and public keys respectively of a given entity, which meets the condition $Q_d = d\,G$ with G the generator point of the curve. Even knowing Q_d and G, it is computationally very hard to find d, given the impossibility of solving the ECDLP.

4.3.2 Elliptic Curve Qu Vanstone (ECQV) certificates

We use the Elliptic Curve Qu Vanstone (ECQV) certificates [149] for the derivation and certification of a key pair by an entity with the Certificate Authority (CA). Fig. 4.1 presents the cryptographic operations and message exchange in ECQV, without including the signature of the messages. Denote the private-public key pair the CA by (d_c, Q_c). Each entity possesses an ID and the public key Q_C of the CA.

ECQV represents a two-round protocol, where the user Alice starts with sending the identity and EC point R_i, where r_i is randomly chosen in the interval $(1, p)$. Upon receiving M_1, the CA also chooses a random number r_c and computes the certificate $(Cert_A)$. In addition, the CA determines auxiliary information a and sends it to Alice. Based on this information, Alice can derive its private key d and verify if the key is legitimate checking that $d_i\ G = H(Cert_A \,||\, ID_A)\ Cert_A \ + \ Q_c$. Here, $H(.)$ represents a collision-resistant hash function, e.g. SHA2 or SHA3.

The ECQV mechanism is very attractive for IoT applications due to the low computational and communication cost [150], [151]. Only the identity and certificate are required for deriving the corresponding public key. Moreover, only the participant can construct the private key,

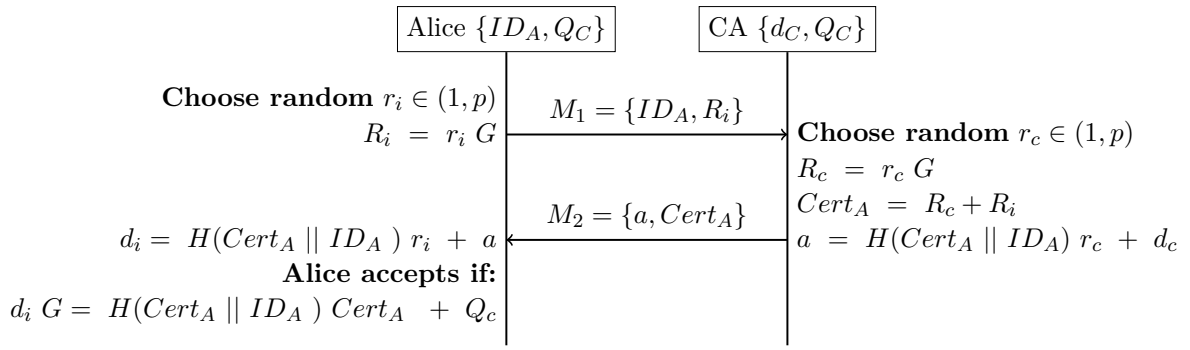

Figure 4.1: Elliptic Curve Qu Vastone (ECQV) Certificates protocol.

since it is the only one who knows r_i. Our scheme is inspired by ECQV for the assignment of a pseudonym to every node and the formation of group keys.

4.3.3 Blockchain overview

A blockchain is a decentralized and distributed ledger system operating on a peer-to-peer network. It enables secure storage and verification of transactions without relying on a central authority. Transactions are the fundamental units of information exchange between entities. The blockchain platform facilitates the direct transfer of various types of data between different entities that do not trust each other.

A typical transaction consists of several fields, including *from*, *to*, *value*, *data*, and additional fields related to mining. The *from* field contains the address of the sender, the *to* field contains the destination address for transmitting the information, and the *value* field represents the transmitted digital currency value. The *data* field is usually empty but can store attachment information for the transaction. In our proposed protocol, we utilize the *data* field to store necessary information for conforming the group key, while setting the *value* field to zero since the blockchain model is used for storing the group key material information.

In the blockchain structure, information is stored in blocks, and each block is connected to the preceding block through a hash pointer. This ensures that modifying a block without detection is virtually impossible since the hash value of a modified block significantly differs from the original block. Furthermore, as the blockchain is distributed among all network peers, any unauthorized alteration made by a dishonest node to the data in a block can be easily identified by other nodes. Adding a new block to the existing blockchain involves a consensus protocol developed collectively by all network peers. This protocol validates the reliability and authenticity of the block within a decentralized and untrusted peer-to-peer environment, eliminating the need for a trusted third party.

The proposed scheme incorporates the distinctive chain structure of a blockchain for information flow, following the typical blockchain structure with slight variations in block contents. Generally, each block includes fields such as *Version number*, *Timestamp*, *Previous hash*, and *Merkle root*. In our scheme, the Version number field documents the Group Identifier, and the *Timestamp* field records the time when the block was generated.

Consensus protocol

The blockchain consensus protocol is the mechanism that ensures that all nodes in the network are synchronized with respect to the content of the blockchain. It is the process by which distributed nodes reach a common agreement on the state of the blockchain, including the order and validation of transactions. The consensus protocol ensures that the blockchain is secure, tamper-proof, and resistant to attacks, such as double-spending or denial-of-service

attacks [152]. In our scheme, different consensus mechanisms could be used as long as they guarantee those properties. Different consensus mechanisms that are suitable for IoT can be found in [5].

Smart Contract

Smart contracts, as an additional functionality of the blockchain, are executable programs whose instances and states are stored within the blockchain. They enable automated code execution without intermediaries and are executed in a decentralized manner by network peers. All results resulting from the invocation of the Smart contract are validated by the consensus protocol [153].

4.3.4 Attack Model

In the proposed scheme, the following assumptions are applied to analyze the security against existing attacks.

The Dolev Yao (DY) model [154] is employed, which involves communications over an insecure channel and an untrusted nature between the parties. According to this model, the adversary $\mathcal{A}$ can read, modify, delete, forge, replay, or insert false information through insecure public channels between two communication parties.

In addition, we consider that $\mathcal{A}$ cannot only perform all functions mentioned in the DY model, but also disclose the secret credentials, session state, and session keys during a session. Therefore, a group key management scheme should ensure that even if secret credentials (such as session temporary secrets and session keys) are disclosed to $\mathcal{A}$, it should have minimal impact on the confidentiality of other members of that group and to the other groups. Furthermore, the following assumptions are made:

1. We consider the existence of malicious nodes within the groups, which aim to impersonate legitimate nodes. The proposed scheme ensures that nodes can verify the validity of the other members of the groups they belong to by participating in the group key construction process and through the verification information stored in the blockchain by the SP.

2. The SP is fully trusted, it is responsible for the construction of individual security material (private-public key pair and pseudonyms) of the nodes and the group. The SP cannot derive the individual private key of each group member but is able to construct the private key of the resulting group key.

3. The SP has high computational, storage, and power capacity. In addition, there is a secure authenticated channel between the users and the SP, which as neither of them is limited, the channel can be established by traditional methods.

4. The Blockchain network could contain dishonest participants (i.e., byzantine members), even though the consensus protocol ensures that data written in the blockchain is immutable and verifiable.

4.4 Proposed scheme

This section presents the considered network model and the processes in each phase involved in the proposed scheme.

4.4.1 Network model

We consider a network architecture as shown in Fig. 4.2, which includes the following four entities:

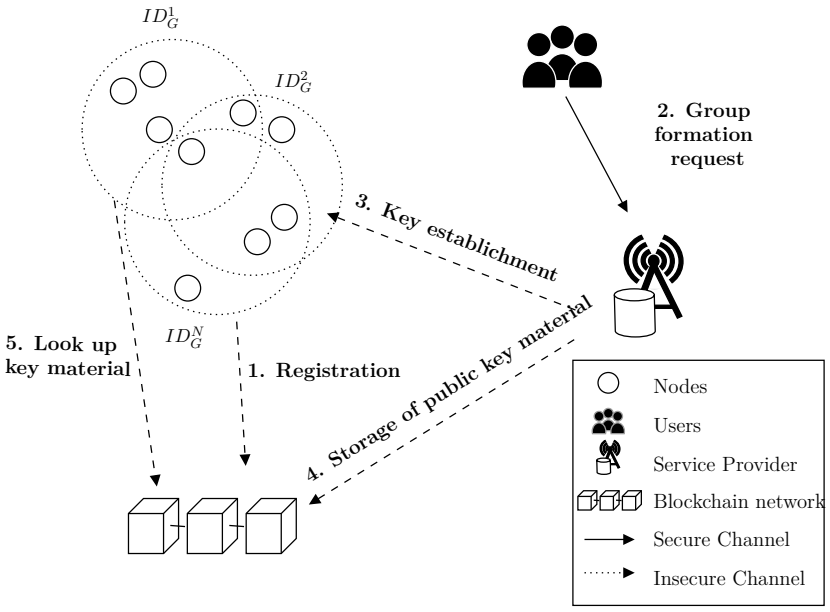

Figure 4.2: The network model consists of nodes, users, Service Provider, Certificate Authority, and blockchain network

Service Provider (SP): The service provider is the entity in charge of managing and maintaining the services that use the data collected by IoT devices. The SP should ensure that the communication between the devices and between devices and the users takes place in a secure manner and without leakage of data. In this scheme, the SP is responsible for forming communication groups according to user requests. For this purpose, the SP provides the nodes with the key material required to form the group keys. We assume that the SP is fully trusted and is responsible for deriving the public keys and also knows the identities of the nodes in the group. In addition, it derives the common shared symmetric key of the group.

Nodes: The nodes represent constrained IoT devices that collect data to be used by the Service Provider and for users. The nodes form a Wireless Sensor Network to monitor parameters such as temperature, humidity, air quality, and others. Each node can belong to several communication groups and communicate with other nodes even if they do not belong to any common group.

Users: Individuals or Companies operate with the IoT data services of the SP. According to their needs, users want to create communication groups among the nodes. There could be different criteria for the formation of groups including geographic location, type of device, or data provided or used in the same application. When a user wants to create a group, a request is sent to the SP, which manages the creation of the group.

Blockchain Network: The Blockchain network facilitates the group formation processes and key establishment between the SP and the nodes. The Blockchain stores the public keys of the nodes and groups, without reference to the real identity of the nodes. Smart contracts enable the proposed mechanism for the detection of malicious nodes and the automatic revocation of group keys.

4.4.2 Scheme process

The proposed blockchain-based group key management scheme involves five phases, as shown in Fig. 4.2. In the first phase, Node Registration (1), all nodes are registered in the Blockchain, storing their identity and public key. Then in the group formation phase (2), users send a request to the SP to create a group of nodes and the SP delivers the necessary key material to the identified group members to allow them to create the group key. Consequently, the SP

sends the individual key material and a pseudonym to each participating group member. The SP stores the necessary information to verify the keys on the Blockchain in the Key material storage stage (4), once this information has been stored on the blockchain, it is possible for a third party to verify the keys formed, this process occurs in the Verification phase (5). The cryptographic operations involved in the different phases are explained in the following sections. For simplicity, Table 4.2 lists the notations used in the scheme.

Table 4.2: Notation

Symbol	**Definition**
ID_i	Real identity of node i
ID_s	Identity of SP
ID_G^k	Identity of group k
d_i, Q_i	Private and public keys of node i
d_s, Q_s	Private and public keys of SP
ID_i^k	Pseudo-identity of the node i in group k
$C = Enc(K, M)$	Symmetric encryption of message M with key K
$M = Dec(K, C)$	Symmetric decryption of message M with key K
$Cert_i^k$	Digital Certificate delivered for pseudonym ID_i^k
$M_1 \| \| M_2$	Concatenation of two messages M_1 and M_2
$H(M)$	Hash of message M

We assume that the parameters to be used in the ECC are published in the blockchain by the SP in an initial process before the development of our scheme. For this purpose, the SP chooses two big enough prime numbers, p, and q, used to initialize the elliptic curve $E : y^2 \equiv x^3 + ax + b \ mod \ p$, where $(a, b) \in Z_q^*$ and holding the non-singularity condition $4a^3 + 27b^2 \not\equiv 0 \ mod \ p$. The SP chooses the generator G of length q and creates the cyclic additive group G that combines all points on E along with the infinity point $\mathcal{O}$.

Subsequently, the SP randomly chooses a secret key $d_s \in Z_q^*$, then computes its related public key $Q_s = d_s G$. In addition, the SP chooses a cryptographic one-way hash function $H : \{0,1\}^* \rightarrow \{0,1\}^n$, which is a deterministic function that produces a fixed length output string of n bits against a variable length input string and is resistant to collision, preimage, and second-image attacks. Finally, the public parameters $\{a, b, p, q, G, Q_s, H\}$, are published in the blockchain by the SP.

Registration

Once the public parameters have been included in the blockchain, the SP can register each node. For this purpose, every node i randomly chooses a private key d_i and derives the respective public key $Q_i = d_i G$. Subsequently, each node i is assigned a unique identifier ID_i, and the identifier and the public key are registered in the Blockchain. This pair of keys of every node will not be used in the group communications, but it will secure the communication between the node and the SP. The public key of the SP, Q_s, will be stored in the memory of the node. This process will be followed every time a new node is added to the system.

Group formation request

Once the nodes have been registered on the Blockchain by the SP, users can request the creation of groups. This is done through an authenticated channel and taking into account the

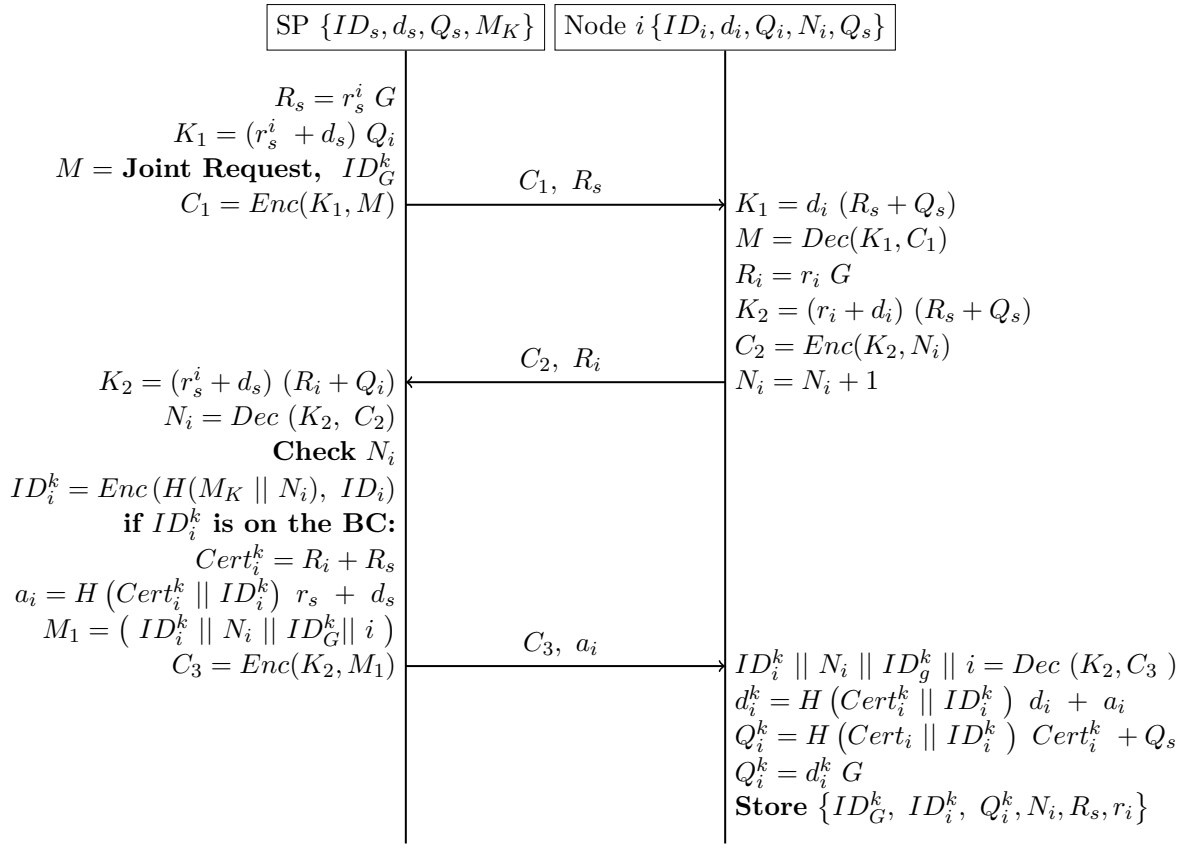

Figure 4.3: Individual Key initialization phase.

permissions set by the SP for each user. Without loss of generality, let's suppose that a user u requires the SP to create a group of n nodes. If the request process by the user u is validated, the SP generates the key material necessary to create a group key among these n identified nodes.

Key Establishment

The SP starts establishing a key following the protocol shown in Fig. 4.3. As a result, each node $i, i \in (1, n)$, receives a pseudonym ID_i^k to be used in the group k, and a certificate is issued for this pseudonym.

As shown in Fig. 4.3, the SP randomly selects r_s^i for each node i, and computes $R_s = r_s^i\, G$ and $K_1 = (r_s^i + d_s)\, Q_i$. Then, the SP encrypts the message request M to conform the group using K_1, $C_1 = Enc(K_1, M)$, as the key and sends the message $C_1||R_s$ to node i. With R_s, node i can also compute K_1, as $K_1 = d_i\,(R_s + Q_s)$. Then, using K_1 as a key, it is possible to decrypt C_1 and the node i knows to which group it will join and increases the number N_i of groups in which it participates. This parameter is used to detect the leakage of a key automatically.

The node then also derives a new random value r_i and computes $R_i = r_i\, G$ and a new common shared key $K_2 = (r_i + d_i)\,(R_s + Q_s)$. This key is used to encrypt the number N_i. The SP can now derive N_i and verifies its validity by checking if a node with pseudo-identity $ID_{i-1}^k = Enc(H(M_n||N_{i-1}), ID_i)$ is present at the blockchain. In the positive case, it continues the process and computes the new pseudo-identity $ID_i^k = Enc(H(M_n||N_i), ID_i)$, together with the auxiliary data for deriving the private and public key pair following the ECQV process (see Section 4.3.2). In the proposed scheme the SP takes the role of the Certificate Authority (CA). Note that a parameter i is communicated, which represents the i-th position in the group and is important to be used in the later phase of the group construction.

Once all the nodes of the new group obtain their respective pseudonyms and certificates, the group key can be formed. The Algorithm 3 shows the procedure to be followed by the SP to deliver the key material to the nodes. First the SP calculates G_K as the sum of all K_i keys, where $K_i = a_i(r_s^i + d_s)(Q_i^k + R_i)$ with $a_i = H(d_s Q_i^k || ID_G^k)$ for each node and $\bar{K_i} = G_K - K_i$. Note that the coefficient a_i is used to facilitate the group key construction in case a (limited amount of node(s) leave(s) or enter(s) the group.

Afterward, the SP computes the private key of the group as $d_G^k = H(G_K || ID_G^k)$ and the respective public key $Q_G^k = d_G^k G$. Finally it publishes in the Blockchain the information $(\bar{K_1}, \bar{K_2}, \ldots, \bar{K_n}, \{(ID_1^k, Q_1^k), \ldots, (ID_n^k, Q_n^k)\}, Q_G^k)$.

Algorithm 3 Group key Formation on SP

Input: ID_s, Q_s, Q_i^k, ID_i^k, r_s^i, d_i, R_i, i

1: **For each node** i:
2: $\quad a_i = H(d_s\, Q_i^k || ID_G^k)$
3: $\quad K_i = a_i \left(r_s^i + d_s\right)\left(Q_i^k + R_i\right)$
4: $\quad \bar{K_i} = \sum_{j \neq i} K_j$ #end for
5: $G_K = \sum K_i$
6: $d_G^k = H(G_K \,||\, ID_G^k)$ #Private group key
7: $Q_G^k = d_G^k\, G$ #Public group key
8: **Publish on the Blockchain:**
$\{\bar{K_1}, \ldots, \bar{K_n}\}, \{(ID_1^k, Q_1^k), \ldots, (ID_n^k, Q_n^k), ID_G^k\},\ Q_G^k$

When this information is public, the nodes can access it and locally compute and verify the generated key. Algorithm. 4 shows the process followed by every node to compute and validate the group key. After obtaining the key material information from the Blockchain, every node i computes a_i, followed by $K_i = a_i \left(d_i^k + r_i\right)(R_s + Q_s)$ and with $\bar{K_i}$ obtained from the BC it can calculate $G_K = K_i + \bar{K_i}$, the private key $d_G^k = H(G_K || ID_G^k)$ and the respective public key $Q_G^k = d_G^k\, G$. The node i accepts the key if it is verified that $Q_G^k = H(G_K || ID_G)\, G$. At the end of this process, a key confirmation stage can be added, in which each node proves to the SP that it has successfully formed the group key. For this purpose, we propose that each node sends to the SP an acknowledge *Ack* signed by means of its private key and group key $(c_i^k = d_i^k + d_G^k)$. Once the SP receives the n messages from the group, it condenses them using a Merkle tree and publishes them in the Blockchain for future audits.

Algorithm 4 Group key Formation on Node i

Input: ID_i, d_i, Q_i, Q_s, Q_i^k, ID_i^k, N_i, R_s

1: **Get from the Blockchain:** $\{\bar{K_1}, \ldots, \bar{K_n}\},\ Q_G^k$
2: $a_i = H(d_i^k\, Q_s || ID_G^k)$
3: $K_i = a_i \left(d_i^k + r_i\right)(R_s + Q_s)$
4: $G_K = K_i + \bar{K_i}$
5: $d_G^k = H(G_K || ID_G^k)$
6: $Q_G^{k\star} = d_G^k\, G$
7: **Accept the key if:** $Q_G^k = Q_G^{k\star} = H(G_K || ID_G)\, G$
8: Sent to the SP $Sign(c_i^k, Ack)$

As a result of this process, an asymmetric key for each group and each node within every group has been established, any of the outsiders can communicate securely with the whole group

using the group public key, on the other hand, there is also the possibility of communicating with a node of a specific group through the pseudonym and the certificate received. Note that only the SP can identify the nodes and know which groups they belong to. Even the group members themselves cannot identify each other, thus maintaining anonymity within the group. In the next section, we provide the security analysis of the proposed scheme.

Automatic Key Revocation

For the automatic revocation of certificates, our scheme uses a Smart Contract, named *Revoke-Cert*. When a node is identified as malicious, the SP invokes the Smart Contract *RevokeCert* that automatically revokes the certificates of that node in all the groups to which it belongs. Without loss of generality let's suppose that node i of group k with pseudo-identity ID_k^i is detected as malicious, and the SP invokes the Smart Contract passing as a parameter this identity. From this parameter, the Smart Contract calculates for all values of N_i, increasing with one every time, $D(H(M_k||N_i), ID_i^k)$ until ID_i exists in the Blockchain. The Certificate associated with this ID_i and all successive ID_i^k, together with the corresponding group keys in which this identity is involved, are revoked, i.e. it is marked as invalid in the blockchain. In addition to the revocation of the certificates, the group keys of the groups to which the node belongs must be updated.

Group key update

The update of the group keys needs to be executed due to one of the four factors, i) a malicious node is detected, ii) a node leaves the group, iii) a node is added to the group, or iv) a periodic update of the keys is required. When a given node leaves a group because of cases i) or ii), the certificate issued for the pseudo-identity will be revoked via the Smart Contract *RevokeCert* and to keep the forward secrecy the key of the group should be updated within the members of the group without the left node. In any of the four cases mentioned where updating the group keys is required, the SP should compute new key material.

We here propose three different mechanisms, each corresponding with a different trade-off in performance and security.

Full update In the full update, the whole process, including the individual key initialization, restarts again and the security remains the same as in the original.

Partial update In the partial update, the nodes keep their individual key pair of the group and only submit a new random point R_i^n, signed by its private key d_i^k. In addition, the SP generates a new random value and corresponding point R_s^n, signed by its private key d_s. This information is used in the group key formation process, similar to before. As a consequence, the individual key initialization phase can be shortened to two communication phases, instead of the original four. However, this comes with a cost of loss in unlinkability among the nodes of the two groups.

Small update In order to make this type of update possible, the SP needs to store the group key G_K. Here, the individual key initialization phase is skipped, and an update of the parameters a_i is defined by $a_i^{'} = a_i^{-1} H(d_s Q_i^k || ID_{G'}^{k'})$, with $ID_{G'}^{k'}$ the identity of the new group defined by SP. As a consequence, SP needs to update the K_i of the participating nodes by $K_i^{'} = a_i^{'} K_i$ and compute the resulting information from these values. Note that the SP can easily retrieve the different values of K_i given the fact that it has G_K stored in its memory and the stored information of $\{\bar{K_1}, \bar{K_2}, \ldots, \bar{K_n}\}$ on the blockchain.

Also, each individual node can easily do the update as it can retrieve K_i from the share $\bar{K_i}$ on the blockchain and the previously used group key G_K.

It is evident that this update process is much less secure, compared to the previous two versions. First, it also lacks protection against the unlinkability of the identities, similar to the partial update process. In addition, it can not offer protection in case the SP or an individual node is hacked in the future and its private key is revealed. In that case, the attacker is able to reveal the group key and break the confidentiality of all messages, encrypted by means of the corresponding group key.

4.5 Security analysis

This section provides a security analysis of the proposed scheme, considering the resistance to possible attacks based on the assumed attack model and the requirements for a secure group communication scheme presented in Section II.A. Note that we only consider in this analysis the full update and thus the complete process. As mentioned before, the partial or small update offers slightly less security.

4.5.1 Formal Security Analysis

We use a formal method to prove the security of the proposed protocol using the symbolic model [155], often called Dolev-Yao. The message exchange is modeled by ProVerif [128] which is an automated symbolic protocol verification tool that supports a diverse set of cryptographic primitives, specified through either rewrite rules or equations. It can prove various security properties: secrecy, authentication, and process equivalences, for an unbounded message space and an unbounded number of sessions. The tool requires as input a description of the protocol to verify in a dialect of the applied pi calculus, an extension of the pi calculus with cryptography. Subsequently, ProVerif autonomously transcribes this protocol specification into Horn clauses and subsequently employs resolution on these clauses to ascertain the validity of the targeted security properties. It has been used for the validation of a large number of protocols [130], [155]–[157].

Since we do not need to verify the performance of our protocol in this section, we assume that there are only four nodes that need to form a group key. Fig. 4.4 shows the ProVerif simulation code for the proposed scheme. Fig 4.5 presents the results of the simulations in Proverif. The secret parameters rs, ri, for the Certificate and pseudonym derivation, as well as the individuals and group private keys dik and dkg or any secret message s encrypted using one of those keys, will not be obtained by the adversary, which formally evidences the security of the proposed scheme under the symbolic model.

4.5.2 Informal Security Analysis

Based on the attack model and the security requirements described above, we now informally discuss that our protocol offers the required security strength.

Lemma 2. *The proposed protocol offers authentication, both at individual and group levels. As a consequence, protection against impersonation and man-in-the-middle attacks is obtained.*

Proof. In the individual key initialization phase, thanks to the publication of the public key material on the blockchain, the SP can send an invitation, which can only be read by the intended receiver (key K_1). Since every receiver has the public key of the SP installed in its memory during registration, it can construct a shared key with only the SP (K_2). As a consequence, the rest of the ECQV mechanism continues with the required guarantee of authentication.

The authentication of the group key is guaranteed by the fact that the construction relies on the key pair established in the initialization phase. Only the entity with the corresponding private key is able to generate the share and thus also to construct the group key.

```
type skey .
type point .
free c:channel.
free s,rs,ri,Ni,i,M,IDgk,IDs,IDi,dik,dkg:bitstring[private].
free Mk:skey[private].
free G,Z1,Z2,Z3:point.
(*---Cryptographic functions---*)
fun EC_Add(point,point):point.
fun EC_Mul(bitstring,point):point.
fun add(bitstring,bitstring):bitstring.
fun mul(bitstring,bitstring):bitstring.
fun concat(bitstring,bitstring ):bitstring.
fun deconcat(bitstring):bitstring.
fun H(bitstring):bitstring.
fun Enc(bitstring,skey):bitstring.
reduc forall m:bitstring,k:skey;Dec(Enc(m,k),k)=m.
fun point2str(point):bitstring[typeConverter].
fun str2point(bitstring):point[typeConverter].
fun str2key(bitstring):skey[typeConverter].
fun key2str(skey):bitstring[typeConverter].
fun point2key(point):skey[typeConverter].
(*---Queries---*)
query attacker ( s ).
query attacker (dik).
query attacker ( ri ).
query attacker (rs).
query attacker (dkg) .

let SP(IDs:bitstring,ds:skey,Qs:point,Mk:skey,Qi:point)=
(* Individual *)
let Rc = EC_Mul(rs , G) in
let K1 = EC_Mul(rs , Qi) in
let C1 = Enc(M , point2key(K1)) in
out ( c ,(C1,Rc) );
in (c,(Ri:point, C2:bitstring)) ;
let K2 = EC_Mul(add(rs,key2str(ds)),EC_Add(Ri,Qi)) in
let Nii = Dec( C2 , point2key(K2)) in
let Aux = concat(IDi,H(concat(key2str(Mk),Nii))) in
let IDik = Enc(Aux,point2key(K2)) in
let Certi = EC_Add(Ri,Rc) in
let H1 = H(concat(point2str(Certi),IDik)) in
let ai = add(key2str(ds),mul(H1,rs)) in
let M1 = concat(concat(IDik,Ni),concat(IDgk,i)) in
let C3 = Enc(point2str(K2),str2key(M1)) in
out (c,(C3,ai));
(* Group *)
in ( c , (Qik:point) ) ;
let Ai = H(concat(point2str(EC_Mul(key2str(ds),Qik)),IDgk)) in
let Ki = EC_Mul(mul(Ai,add(rs,key2str(ds))),EC_Add(Qik,Ri)) in
let Gk = EC_Add(EC_Add(Z1,Z2),Ki) in
let dkg = H(concat(point2str(Gk),IDgk)) in
let Qkg = EC_Mul(dkg,G) in
out (c,(Qkg,ai,EC_Add(Ki,Z1),EC_Add(Ki,Z2),EC_Add(Z1,Z2))).
```

```
let Nodei(di:skey, Qi:point,Ni:bitstring,Qs:point) =
(* Individual *)
in ( c , (Rc:point,C1: bitstring) ) ;
let K1 = EC_Mul(key2str(di),Rc) in
let M1 = Dec(C1,point2key(K1)) in
let Ri = EC_Mul(ri,G) in
let K2 = EC_Mul(add(ri,key2str(di)),EC_Add(Rc,Qs)) in
let C2 = Enc( Ni , point2key(K2)) in
out ( c ,(C2,Ri) );
in ( c , (C3:bitstring, ai: bitstring) ) ;
let Mi1 = Dec(C3,point2key(K2)) in
let IDik = deconcat(Mi1) in
let Certi = EC_Add(Ri,Rc) in
let A1 = concat(IDik,point2str(Rc)) in
let A2 = H(concat(point2str(Certi),A1)) in
let dik = add(mul(A2,key2str(di)),ai) in
let Qik = EC_Add(EC_Mul(A2,Certi),Qs) in
out(c,(Qik,Enc(s,point2key(Qik))));
(* Group *)
in ( c , (Ki_n:point) ) ; (* Ki_n = Gk-Ki =Z1 + Z2 *)
let Ai1 = H(concat(point2str(EC_Mul(dik,Qs)),IDgk)) in
let Ki = EC_Mul(mul(Ai1,add(dik,ri)),EC_Add(Qs,Rc)) in
let Gk1 = EC_Add(Ki_n,Ki) in
let dkg = H(concat(point2str(Gk1),IDgk)) in
let Qkg = EC_Add(EC_Mul(A2,Certi),Qs) in
out(c,(Qkg,Enc(s,point2key(Qkg))));
0.

process
new di : skey ;
new ds : skey ;
let Qs = EC_Mul(key2str(ds),G) in
let Qi = EC_Mul(key2str(di),G) in
((!SP(IDs,ds,Qs,Mk,Qi))|(!Nodei(di,Qi,Ni,Qs)))
```

Figure 4.4: Proverif code for the proposed scheme.

```
Verification summary:
Query not attacker(s[]) is true.
Query not attacker(dik[]) is true.
Query not attacker(ri[]) is true.
Query not attacker(rc[]) is true.
Query not attacker(dkg[]) is true.
```

Figure 4.5: Proverif simulation results of the proposed scheme.

For an impersonation or a man-in-the-middle attack, the attacker should be in possession of the private key. Thanks to authentication protection, this is impossible. □

Lemma 3. *The proposed protocol offers integrity, both at the individual and group levels.*

Proof. In the individual key initialization phase, the node can verify the correctness of the received material thanks to the ECQV mechanism. In the group key formation phase, an acknowledgment is sent by each of the members and published on the blockchain, using the newly derived group key. This proves the integrity of the obtained result. □

Lemma 4. *The proposed protocol offers confidentiality, both at the individual and group levels.*

Proof. First, the individual key initialization phase is based on the ECQV protocol, which is known to offer confidentiality. Second, for the group key construction, the hardness of the ECDLP ensures the confidentiality of the individual share and the security of the secret sharing mechanism ensures the impossibility of retrieving the group key by any outsider. Since the nodes join the group in an authenticated way, only group members composed of authenticated nodes can communicate with the use of the symmetric group key. Therefore, our proposal ensures confidentiality. □

Lemma 5. *The proposed protocol offers protection against replay attacks.*

Proof. In the proposed scheme, at each step during individual key establishment and group key formation, the SP first checks the freshness of the message by seeking an authentication record with the same values to avoid duplicates. If the incoming request is determined to be a fresh message, the SP will continue with the protocol. Otherwise, the request is rejected and the transaction is rolled back. The fact that each message involves the selection of random values and the probability of selecting the same values is negligible, guarantees the prevention of replay attacks originating from malicious nodes.

□

Lemma 6. *The proposed protocol offers anonymity and unlinkability of the nodes in the group.*

Proof. First of all, in the individual key initialization phase, the original identity ID_i is protected by means of the master key M_K of the SP and the new identity ID_i^k of the node cannot be retrieved without knowledge of either the private key of the node or the SP (construction of K_2). As a consequence, no link can be made between ID_i and ID_i^k in the later group formation process.

Since the construction of ID_i^j, for different values of j are independent of each other and the relation cannot be revealed without knowledge of the master key M_k, the identities of the nodes in the different groups are also unlinkable. □

Lemma 7. *The proposed protocol offers perfect forward secrecy.*

Proof. Perfect forward secrecy implies that if the long-term key material is leaked, protection of the previously sent messages is still guaranteed. We here need to consider three situations.

- Leakage of the private key of a node (d_i): If this key is leaked, only a request to join message can be leaked from the individual key initialization phase. For the rest of this phase, a temporary random value r_i is used, and thus nothing else, e.g. ID_i^k, ID_G^k, N_i, i, can be derived. Without this information, also from the group key formation, no further relevant data can be revealed.

- Leakage of the private key of SP (d_s): Both, the protection in the individual key initialization phase and the group formation phase, rely on the combination of the private key and a temporary random value r_i^c. Without both, no relevant data can be derived.

- Leakage of master key of SP (M_k): This will not lead to any vulnerability with respect to confidentiality and authentication. However, if this key is leaked, the unlinkability of the identity for the individual nodes is broken as this key is used to construct the different pseudonyms of the node, which is exploited in the revocation process.

□

Lemma 8. *The proposed protocol offers protection against session-specific temporary leakage attacks.*

Proof. In this type of attack, we assume that session-specific variables like random values are leaked (for instance because of vulnerabilities in the random generator). Since in both the individual key initialization phase and group formation phase, a combination of the private key and random value is used for the derivation of the keys K_1, K_2, K_i, G_K, protection against this type of attack is automatically obtained. □

Lemma 9. *The proposed protocol offers both backward and forward secrecy.*

Proof. Forward secrecy ensures that the group member who has left the group session will have no ability to get the further session key. Backward secrecy guarantees that after a node leaves the group, it will not be able to retrieve the content of messages sent in the group. These features are ensured thanks to the group key update phase, which is automatically started by means of a smart contract. As explained before, the full update, partial update, and small update enable this feature but differ in the security features obtained for the resulting group key. □

4.6 Performance evaluation

In this section, an analysis of the main performance parameters influenced by the proposed scheme is presented and compared with the most relevant, related, and recently published works. First, the computational cost and network overheads caused by using the proposed scheme are analyzed and presented in Table 4.4. In addition, the delay caused by the blockchain on the scheme is estimated.

Table 4.3: Time Cost of Different Cryptographic Operation

Notation	Description	Time(ms)
T_H	Hash operation	0.0007
T_{mul}	Scalar multiplication	0.0082
T_{inv}	Modular inverse operation	0.0005
T_{exp}	Scalar exponentiation	0.0001
T_{Ad}	EC Point Addition	3.3772
T_{Sm}	EC Scalar Multiplication	3.6916
T_{Enc}	AES128 encryption/decryption	0.1125
T_{bp}	Bilinear pairing	22.5412
T_{Abp}	Point Addition on BP	11.4675

For evaluation of the computation time, the heaviest cryptographic operations involved in our scheme are measured on a Raspberry Pi 3B Broadcom BCM2837, equipped with a microprocessor Quad-core Cortex-A72 (ARM v8) 64-bit SoC @ 1.5 GHz, 1GB RAM, and 16G ROM. We use the *pycryptodome* library (version 3.17.0) from Python for cryptographic operations.

For the ECC, we use the secp256k1 curve, AES128 as the symmetric encryption algorithm and SHA256 as the general hash function. In all experiments, the input message and the random numbers used in operations were randomly generated in every iteration with a length of 256 and 128 bits respectively. In the cases where the modular inverse of the generated number does not exist, the time consumed is not considered in the calculation of the average time. The average time of 1000 executions of different cryptographic operations is shown in Table 4.3. For a confidence level of 95%, this number of runs ensures that the maximum error in the estimation of the metrics is less than 3%.

4.6.1 Computational cost Analysis

To estimate the computational overhead of the proposed scheme, a theoretical analysis of the most time-consuming operations is performed, neglecting lighter operations such as string concatenation and XOR. Table 4.4 shows the operation that n nodes should perform to compute the group key, including the individual phase of the scheme. A comparison with some existing solutions is also included in the Table. As in [135]–[137], [147], the blockchain operations are not included in this analysis as they depend on the underlying blockchain mechanism used for the solution. The analysis of the impact of the blockchain is discussed further in this section.

Table 4.4: Comparison of Computation cost

Scheme	**Complexity**
[136]	$(n+3)\,T_{bp} + (8n-2)\,T_{mul} + (6n+6)\,T_{exp}$
[135]	$2\,T_{Sm} + 2n\,T_{mul} + n\,T_{inv}$
[147]	$(n+7)T_{Sm} + 4T_{bp} + (n+3)T_H + 2T_{Enc} + (3n+1)T_{Ad}$
[137]	$9n\,T_{bp} + 3n\,T_{mul} + n\,T_{inv} + n(n+1)\,T_{Abp}$
Ours	$5n\,T_{Sm} + 3n\,T_{Ad} + n\,T_H + 2n\,T_{Enc} + n\,T_{mul}$

The results presented in Table 4.4 allow the estimation of the computational cost required by the proposed scheme. Fig. 4.6 shows the results of the comparison with the different proposed protocols under the same conditions. With this objective, the number of nodes is varied from 100 to 1000 with steps of 100. For all protocols, the time required to complete the process grows linearly with the increase in the number of nodes. The proposed scheme presents a performance similar to the solution proposed in [135] and outperforms the other solutions included in the comparison. This low computational cost is due to the lightness of the operations used in the protocol, mainly the ECC and hash functions, and the fact that publishing the key material in the blockchain reduces the computational cost for IoT devices. Even in the case of 1000 nodes, the time required is less than 8 seconds, which is considered acceptable for many IoT applications.

4.6.2 Communication Overhead Analysis

Since the message length varies depending on the task and application, only the payload related to authentication is calculated. The communication overhead is calculated based on the information contained in the messages used for authentication. Table 4.5 shows the information received and transmitted for conforming a group key between n nodes for the proposed scheme and some recently proposed schemes. In Table 4.5, $|G|$ and $|q|$ denote the length of the cyclic group points and the length of the prime number respectively.

Using the results of Table 4.5, it is possible to estimate the communication cost required by the proposed scheme. For this purpose, we consider the radio interface CC2420. The

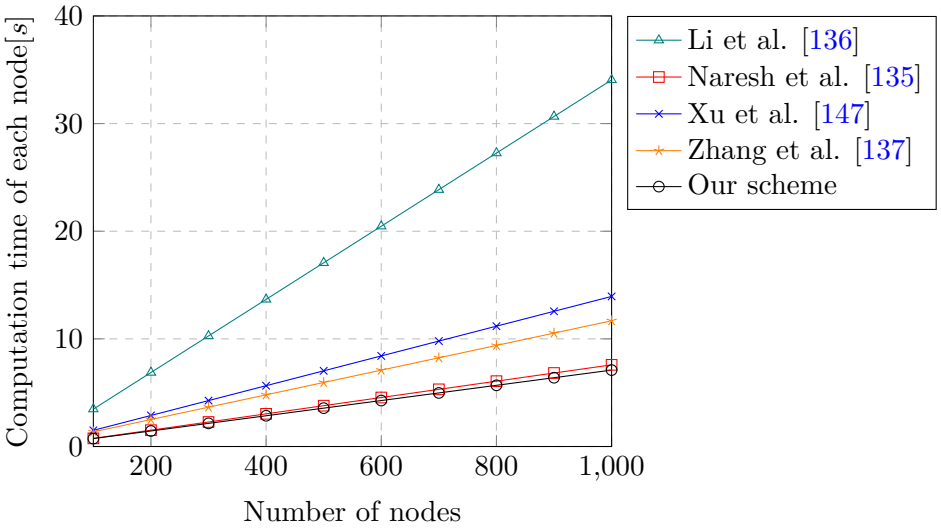

Figure 4.6: Computation time of each node.

Table 4.5: Comparison of Communication Overhead

Scheme	Message length sent	Message length received
[136]	$(n+2)\|\mathbb{G}\|$	$3(n-1)\|\mathbb{G}\|$
[135]	$2(n+1)\|\mathbb{G}\|$	$2(n+1)\|\mathbb{G}\|$
[147]	$(5C+2T)$	$(3n-1)C+nT$
[137]	$(n+4)\|\mathbb{G}\|+\|q\|$	$(2n)\|\mathbb{G}\|+\|q\|$
Ours	$3n\|\mathbb{G}\|+\|q\|$	$(n+2)\|\mathbb{G}\|+\|q\|$

C and T are parameters defined in [147] with length 256 and 64 bits.

CC2420 radio transmitting at 0 dBm power and using O-QPSK modulation with Direct-sequence spread spectrum (DSSS) on IEEE 802.15.4 (2.405GHz) channel 11, has a power consumption of $57.42mW$ $(P = 17.4mA * 3.3V)$. The reception power consumption is $65.01mW$ $(19.7mA * 3.3V)$. With these conditions the nominal bit rate is 250 kbps, to transmit and receive one bit of information, $0.224uJ$ and $0.254uJ$ are consumed respectively. Figure 4.6 shows the results of the comparison with different protocols proposed under the same conditions. For these experiments we use $|G| = 160$ bits and $|q| = 128$bits.

In Fig. 4.6, the number of nodes is varied between 100 and 1000, with steps of 100, and the energy consumed in sending and receiving the information required by each protocol to form a group key is calculated. The black curve shows the performance of the proposed scheme, as it can be seen that as the number of nodes increases, the energy required to form the key grows linearly. Similar behavior occurs in all the protocols. Our protocol consumes more energy than those proposed in [137] and [136]. Still, there is no significant difference, and our protocol provides security properties that the other protocols do not guarantee. In all cases, our protocol consumes less than 300 mJ of energy in each node even when 1000 nodes construct the group keys, which is suitable for IoT applications.

4.6.3 Blockchain Performance Analysis

The timing analysis for group key formation presented above does not consider the delay in the blockchain because the timing of operations in the blockchain is highly dependent on the type

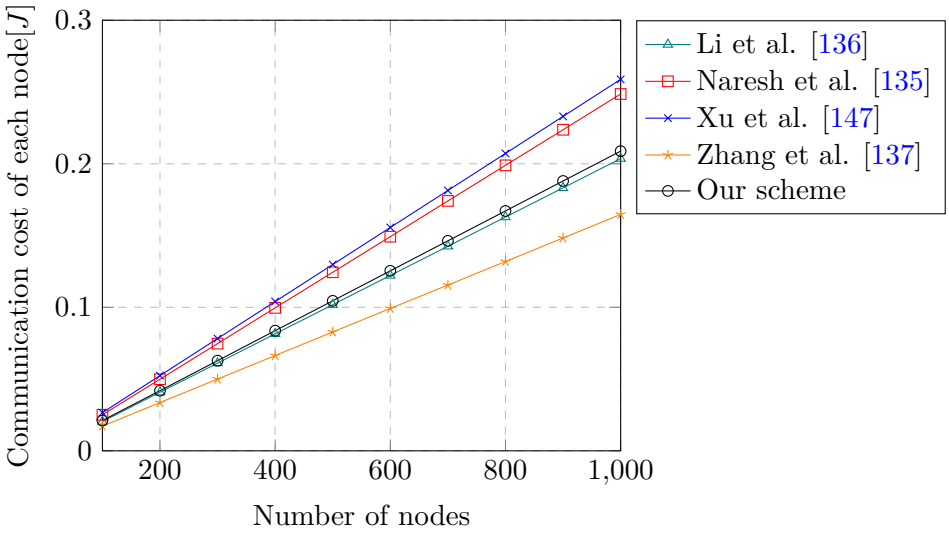

Figure 4.7: Communication cost of each node.

of blockchain used and for the proposed scheme several types of blockchains could be used. This section analyzes the latency and storage capacity required to store the blockchain.

Blockchain Latencies

In the proposed scheme, during the different phases, the blockchain is used to store information necessary to authenticate the nodes, generate the pseudonyms and the respective certificates as well as to validate the group keys, and revoke the certificates. During these processes, three operations involving the blockchain are required: writing, querying, and Smart Contract invocation. Write latency (T_W) and Smart Contract invocation latency (T_{SC}) are the time between executing the write operation code or invoking the Smart Contract and validating and publishing the data on the blockchain as a result of that execution or invocation. In both cases, this process ends with the execution of a consensus protocol, which validates a new block with the new information. Similarly, query latency (T_Q) is the time that elapses between the execution of the read operation code and the return of the result. No transaction validation or consensus is required for this, and the data can be queried from a local copy of the blockchain.

During the registration phase of the nodes, the SP uses a Write operation for each node, which records its real identity and its public key. Subsequently, the SP writes the key material of each node to form the group key, and finally another write operation with the group key, the group identity, and the necessary information to validate the key creation process. On the other hand, if any malicious node is detected or any node enters or leaves the group, the Smart Contract *RevokeCert* is invoked. All these operations add extra delay to the protocol. This delay depends on the consensus protocol used in the cases of Write and Invoke and on the latency of the Blockchain network servers in the case of Queries.

To numerically estimate these delays in blockchain operations, a blockchain network using Practical Byzantine Fault Tolerance (PBFT) as a consensus protocol is implemented. This protocol is suitable for IoT environments because dishonest peers can exist without affecting transparency and integrity, and it is also not computationally expensive compared, for instance, to other protocols based on Proof of Work.

To obtain the latency parameters of the Smart Contract in the blockchain, a blockchain network is deployed using Hyperledger Fabric (v2.2) in a dockerized virtual environment (Docker v20.10.6). The network consists of three validators and nine peers. The validators process, store, validate transactions and add new blocks to the blockchain using the consensus mechanism, as

well as handle queries from peers. Hyperledger Caliper (v0.4.2) is installed as a benchmark to obtain the average query and writing delays and invocation to the Smart Contracts (chaincodes in Hyperledger Fabric).

Table 4.6: Time Cost of Blockchain Operations

Operation	Min (s)	Max(s)	Average(s)
RevokeCert	1.541	1.792	1.626
Query	0.297	0.453	0.376
Write	1.127	1.582	1.350

Table 4.6 shows the result of the minimum, maximum, and average time of the main operations of the Smart Contracts and the queries to the blockchain. The Hyperledger Fabric Python SDK [1] was used to perform 10^4 runs of each operation. We estimate these parameters using a transaction size of $64kB$, which is sufficient to transmit the key information. The Hyperledger Fabric experiments were conducted on a laptop with Intel Core i7-7700 CPU 3.60GHz×8, and 16GB of RAM.

From the results of Table 4.6, it can be observed that with a dedicated blockchain, each node perceives write and Smart contract invocation delays of less than 2 seconds, which do not significantly affect the performance of the protocol, in addition to the advantages in terms of computation time and increased levels of security and transparency that the use of the Blockchain in a group key management protocol provides.

4.7 Conclusion of the Chapter

Since most Internet of Things (IoT) devices have limited power, there is a growing preference for developing IoT applications based on group communication. Group communication transmits messages to all group members more efficiently compared to sending them one by one. Traditional group key management protocols feature centralized architectures resulting in scalability issues and the single point of failure problem. Blockchain technology addresses the centralization problem and helps to improve the security level of a group key management protocol.

In this chapter, we propose a group key management system in which anonymity is maintained within the group while constructing an asymmetric key, which allows non-group outsiders to communicate with group nodes securely. We provide an efficient scheme based on elliptic curve cryptography and prove its security. We also analyze the computational and communication cost of the protocol and the impact of using blockchain in it. These results prove that the protocol is secure and efficient compared to other existing protocols.

[1] available: https://github.com/hyperledger/fabric-sdk-py

Maximizing Social Welfare through Fair Validators Selection in Permissioned Blockchains

5.1 Introduction

Blockchain is a distributed ledger technology that uses cryptographic methods and consensus protocols to improve the degrees of trust, transparency, and security of the exchange and distributed storage of data among parties that do not trust each other [158]. In recent years, government, research institutions, and industry have become interested in Blockchain, with applications in almost every industry segment, including e-voting [159], [160], supply chain [161], [162], Internet of Things (IoT) [163], [164], healthcare [165], [166] and others [167].

In the blockchain structure, information is organized into blocks that are connected through hash pointers, ensuring the integrity of the data. Modifying a block without detection is nearly impossible because any changes would result in a significantly different hash value. Additionally, the distributed nature of the blockchain network allows unauthorized alterations made at one node to be easily identified by other nodes. Verified blocks are added to the blockchain using a consensus algorithm which ensures reliability, authenticity, and synchronization in a decentralized, untrusted peer-to-peer environment, eliminating the need for a trusted third party [168].

Blockchain systems fall into two main categories based on participant access: Permissioned Blockchains and Permissionless Blockchains [168]. Permissionless blockchains, often referred to as public blockchains, offer open access, allowing any user to join the network and participate in the consensus. In contrast, Permissioned Blockchains present a clear separation of roles, where the nodes in the consensus (Validators) are known and determined, and users are known and identified in advance. This allows the use of more efficient consensus protocols, improving transaction processing rate and proper access control, which make Permissioned Blockchains more attractive to applications with high-performance requirements like IoT.

In existing Permissioned Blockchains, the nodes responsible for verifying and validating the current block must be selected at the initial stage. In most cases, the Validators are predetermined and belong to a predefined set [169], [170]. However, this approach does not ensure that users will consistently experience the highest quality of service or equitable treatment regarding the cost of utilizing blockchain services or the speed of validating transactions. In certain scenarios where users may not encounter uniform latencies from different servers, significant disparities in Transaction Confirmation Times (TCT) may arise [171], leading to a reduced level of fairness. For instance, in a scenario where a node encounters prolonged TCT from a specific set of servers if the group of Validators is mostly selected among such set, this user is likely to perceive a very unequal TCT to the one possibly perceived by other users. This unfair service could diminish the enthusiasm and participation of users and ultimately cause the blockchain system to lose its advantages and attractiveness.

5.1.1 Research Motivation and Contribution

Social welfare, in the context of resource allocation, refers to the overall utility derived by individuals within a group [172]. It encompasses the collective satisfaction, benefits, and improvements in the quality of service by individuals as a result of the allocation and distribution of resources. Social welfare could take into account various factors such as fairness, equality, efficiency, and the satisfaction of requirements. The goal is to optimize the allocation of resources in a way that maximizes the overall welfare of the group, ensuring that resources are distributed in a manner that benefits the greatest number of individuals and minimizes any potential disparities or inequalities [173].

This Chapter presents FairVSP, a novel algorithm for the selection of Validators, aimed at maximizing the Social Welfare of Permissioned blockchain users. The idea is that blockchain nodes collaboratively determine the set of nodes that will act as Validators for a given Block while maximizing Social Welfare. Since solving this problem leads to a combinatorial optimization problem with no known polynomial solution, *i.e.,* an NP problem, we compare the performance of several Evolutionary Algorithms based on heuristic solutions and propose a strategy to enhance their performance and solve them in a distributed way. The improvement strategy is based on decomposing the main problem into sub-problems with smaller search spaces and every node solves a sub-problem. Followed by a voting process where a ranking for each node is computed based on the number of times it appears in the solution of the sub-problems. Our approach integrates with many existing Permissioned Blockchains without significant change or non-standard assumptions on the blockchain implementation.

The main contributions of the present protocol are summarized below:

- FairVSP focuses on maximizing Social Welfare among users of the blockchain. Specifically, we address the distributed solution of the problem of selection of the set of Validators nodes in voted-based consensus algorithms. We provide a mathematical model for the problem, prove the hardness, and provide a method to solve it in a distributed manner.
- We conduct a security analysis of the proposed protocol and demonstrate its resilience against common attacks on the blockchain.
- We evaluate the quality of the solution, the convergence time, and the Price of Fairness of the solution found by FairVSP with Evolutionary algorithms. Moreover, the computational and communication overhead of the proposed algorithm is analyzed.

5.2 Related Works

5.2.1 Consensus protocol for Permissioned Blockchain

Consensus protocol enables a group of nodes to collectively establish and support a preferred option by reaching an agreement. It involves making decisions where nodes prioritize the majority choice over their individual preferences. Two distinct categories of consensus protocol are used in Blockchain: Proof-based and Voting-based [168]. Proof-based algorithms like Proof of Work (PoW) [174], require nodes to solve a cryptographic problem to append a new block. Those algorithms have been extended and used in Permissionless Blockchain platforms including Bitcoin [175] and Ethereum. Conversely, Voting-based algorithms involve exchanging results in the network before appending a block to the blockchain, making it suitable for Permissioned Blockchain where participants are known but do not trust each other or could be crashed or victims of attacks. Hyperledger [176] and Corda[177] are emerging platforms designed for enterprise and business applications based on Permissioned Blockchain technology.

For a consensus protocol to be applicable in a Permissioned Blockchain, it must meet specific requirements that align with the characteristics and goals of permissioned networks and

applications. A consensus protocol for a Permissioned Blockchain should possess the following properties:

Identity and access control A Permissioned Blockchain often requires a mechanism for identity management and access control to verify the authenticity and permissions of participants. The consensus protocol should integrate with these mechanisms to ensure that only authorized participants can propose and validate transactions and blocks.

Privacy and confidentiality Many Permissioned Blockchain applications deal with sensitive or proprietary data. The consensus protocol should provide mechanisms to ensure the privacy and confidentiality of transactions and sensitive information shared among participants, such as through encryption or selective disclosure.

Permissioned participation The consensus protocol should support a predefined set of known and trusted Validators who are authorized to participate in the consensus process. This differs from Permissionless Blockchains where anyone can join and participate in consensus.

Byzantine fault tolerance Byzantine fault tolerance ensures that the system can tolerate and recover from malicious or faulty behavior by nodes. Byzantine fault tolerance provides a consensus protocol that can handle arbitrary failures and maintain the integrity and security of the blockchain. That can be encapsulated in three properties: Liveness, Safety, and Validity. Liveness refers to the property that all honest parties (nodes or participants in the blockchain network) eventually commit to a decision. Safety ensures that all honest parties agree on the same values. This guarantees that there is a single consistent version of the truth (block). Finally, Validity guarantees that honest participants follow the protocol correctly and when a block is sent by an honest party, it will be included in the blockchain as long as it follows the rules of the protocol [178].

Efficiency and scalability Permissioned Blockchains often deal with higher transaction volumes compared to permissionless networks. Therefore, the consensus protocol should be designed to handle a significant number of transactions per second efficiently, with faster confirmation times than permissionless networks, and scale as the network grows.

Fairness among users Fairness in confirmation time prevents any particular user or group from receiving preferential treatment, promoting equality and avoiding potential biases. Additionally, fairness in confirmation time enhances transparency and trust among users by ensuring that no participant is disadvantaged or subjected to excessive delays, reinforcing the overall integrity and credibility of the Permissioned Blockchain network.

Table 5.1 provides a comparison of a non-exhaustive group of the most common consensus protocols in Permissioned Blockchains. The criteria for comparison are the properties required by a consensus protocol in Permissioned Blockchains, in clauses a) to f) of Section 5.2.1. In Table 5.1, all consensus algorithms exhibit robust identity and access control, ensure privacy and confidentiality while presenting a permissioned participation of the blockchain node, and in most cases are resistant to Byzantine attacks. However, not all of them guarantee efficiency and scalability and do not provide low latency in transaction validation times and fairness among these delays for different users.

5.2.2 Validators Selection process

To reduce communication complexity in consensus protocols based on voting, a set of nodes is typically chosen to act as Validators and participate in the voting process. There are various

Table 5.1: Consensus protocol for Permissioned Blockchain

	a)	b)	c)	d)	e)	f)
PBFT [169]	●	●	●	●	◐	○
PoA [179]	●	●	●	◐	●	○
FBA [180]	●	●	●	●	◐	○
Ripple [181]	●	●	●	●	●	○
Stelar [182]	●	●	●	●	●	○
Tendermint [170]	●	●	●	●	●	○
PoET [183]	●	●	●	●	●	○
DPoS [184]	●	●	●	●	●	○

● Strong ◐ Moderate ○ Low

PBFT: Practical Byzantine Fault Tolerance, **PoA:** Proof of Authority, **FBA:** Federated Byzantine Agreement, **PoET:** Proof of Elapsed Time, **DPoS:** Delegated Proof of Stake.

approaches to perform this selection. The most common is for Validators to be a predetermined set established by the network administrators, as is the case with Practical Byzantine Fault Tolerance (PBFT) [169]. On the other hand, some protocols, like Tendermint [170] and Proof of Elapsed Time (PoET), involve a random lottery system where the probability of being selected as a Validator is proportional to the amount of resources, reputation, or other criteria. Finally, another existing approach consists of sequentially rotating the set of Validators, following specific sequences, for instance in a round-robin fashion. Ripple [181], and Stellar[182] are examples of the use of this approach. Based on this analysis of existing approaches used in the existing consensus protocols (See Table 5.1), this chapter introduces a novel algorithm for the selection of Validators that maximizes the Social welfare based on Fairness among users of the Permissioned Blockchain.

5.3 Preliminaries

This section aims to provide a brief summary of the building blocks employed in the proposed algorithm.

5.3.1 Social Welfare and α-Fairness

Social welfare functions (SWF) are a measure of the total utility obtained by all users in the system [97]. For a system with N users, where user i has utility function $U_i(x_i)$ that depends on its allocation x_i, the Social welfare (W) can be computed as the sum of individual utilities:

$$W = \sum_{i=1}^{N} U_i(x_i) \tag{5.1}$$

Here, $U_i(x_i) : \mathbb{R}_+ \to \mathbb{R}_+$, measures the satisfaction of user i from receiving x_i resources The goal is to optimize the allocation vector x_i to maximize social welfare while adhering to constraints. Several SWFs combine equity and efficiency, sometimes with a parameter that regulates the relative importance of each. α-Fairness is a well-known SWF, where the parameter α governs the trade-off between maximizing the welfare of the worst-off user and achieving a more equal distribution of utilities. The α-Fairness utility function $U_i(x_i)$ for agent i is given by:

$$U_i(x_i) = \begin{cases} \frac{x_i^{1-\alpha}}{1-\alpha} & \text{if } \alpha \neq 1,\ x_i \geq 0 \\ \ln(x_i) & \text{if } \alpha = 1,\ x_i > 0 \\ -\infty & \text{if } x_i \leq 0 \end{cases} \tag{5.2}$$

The range of values for α within the interval $\alpha \in [0, \infty)$ reflects the trade-off between individual fairness and efficiency. Lower values of α are associated with utilitarian welfare, emphasizing societal well-being (efficiency). Conversely, higher values of α indicate a preference for egalitarian allocation of resources, placing greater emphasis on individual equality (fairness). For instance, when $\alpha = 1$, the conventional log-based proportional-fair utility imposes notable penalties on prioritizing high utility in some applications it could potentially disadvantage some users [185]. When the allocation vector x_i is a binary vector, maximizing W leads to solving a combinatorial optimization problem. In our approach, Evolutionary algorithms are used to deal with this problem.

5.3.2 Evolutionary Algorithms for Optimization

Evolutionary algorithms are stochastic search methods inspired by natural processes that have been applied successfully in many search, optimization, and machine learning problems [186]. Among the diverse spectrum of evolutionary algorithms, this chapter focuses on Simulated Annealing (SA) [187], Ant Colony Optimization (ACO) [188], and Genetic Algorithms (GAs) [189] because these algorithms have demonstrated their efficacy in solving combinatorial optimization problems characterized by non-linearity, multimodality, and high-dimensionality.

Simulated Annealing Simulated Annealing (SA) is a popular optimization technique inspired by metallurgical annealing. It mimics gradual cooling to achieve a low-energy state. Algorithm 5 shows an SA pseudocode for minimizing the function f. The algorithm begins with an initial solution and iteratively explores neighboring solutions, through the function *GenerateNeighbor*, by allowing moves to higher-cost solutions with a decreasing probability (See line 7 in Alg. 5). This stochastic acceptance of poorer solutions aids in avoiding local optima, eventually reaching a global optimum. The temperature parameter, gradually decreasing during optimization, via the strictly decreasing *CoolingSchedule* function, balances exploration and exploitation and leads to convergence.

Algorithm 5 Simulated Annealing

1: Initialize: $T_{\text{initial}}, T_{\text{final}}$, current_solution
2: $T \leftarrow T_{\text{initial}}$
3: **while** $T > T_{\text{final}}$ **do**
4: **for** $i \leftarrow 1$ **to** $n_{\text{iterations}}$ **do**
5: *neighbor* $\leftarrow$ GenerateNeighbor(current_solution)
6: $\Delta E = f(\text{neighbor}) - f(\text{current\_solution})$
7: **if** $\Delta E < 0$ **or** $\text{random}(0,1) < e^{-\Delta E/T}$ **then**
8: current_solution $\leftarrow$ neighbor
9: **end if**
10: **end for**
11: $T \leftarrow \text{CoolingSchedule}(T)$
12: **end while**
13: **return** current_solution

Ant Colony Optimization (ACO) This metaheuristic algorithm is inspired by real ants behavior, replicating their pathfinding to food sources. Algorithm 6 shows an ACO pseudocode for

minimizing function f, it simulates artificial ant populations traversing solution spaces, leaving pheromone trails on solution components. For each ant, a path is generated (*ConstructAntPath*), in which paths with a higher concentration of pheromones are more likely to be chosen. Subsequently, the pheromone matrix ($\wp$) is updated using the functions *UpdatePheromones* and *EvaporatePheromones*, which increase the pheromone level on the paths of the current solution and decrease on the paths not chosen, respectively. ACO utilizes pheromone reinforcement for positive feedback and diversifies by exploring alternative paths, granting resilience against local optima.

Algorithm 6 Ant Colony Optimization

```
1: Initialize: n_ants, n_iterations, pheromone_matrix
2: best_distance ← 0
3: best_path ← {}
4: for i ← 1 to n_iterations do
5:    for j ← 1 to n_ants do
6:       ant_path ← ConstructAntPath()
7:       ant_distance ← f(ant_path)
8:       if ant_distance < best_distance then
9:          best_distance ← ant_distance
10:         best_path ← ant_path
11:      end if
12:      UpdatePheromones(ant_path, ant_distance)
13:   end for
14:   EvaporatePheromones()
15: end for
16: return best_path
```

For the *ConstructAntPath* function, the next node to be included in the solution is randomly selected based on the probability of each path. The probability of moving from node i to j is calculated as follows:

$$p_{i,j} = (\wp[i,j])^{\alpha} \left(\frac{1}{\text{path\_cost}[i,j]} \right)^{\beta} \tag{5.3}$$

Here, α and β are parameters that play a crucial role in guiding the behavior of the ants during their search for optimal solutions. The parameter α represents the pheromone importance or exploration factor. A higher value of α means that ants give more weight to the pheromone trail left by previous ants. This makes them more likely to choose paths with higher pheromone concentration, favoring exploration of the solution space. On the other hand, β represents the heuristic information importance or exploitation factor. A higher β gives more weight to problem-specific knowledge, influencing ants to prefer paths that are considered better according to the heuristic information. With a probability defined by the parameter Elitist probability, the best solution path will be selected. Finding an appropriate balance between α and β is crucial for achieving effective solutions in different problem domains, as it determines the trade-off between exploration and exploitation strategies employed by the algorithm.

Genetic Algorithm (GA) This is a population-based optimization method inspired by natural selection and genetics, which operates on a population of candidate solutions rather than a single solution. Algorithm 7 presents pseudocode for a GA aimed at minimizing the fitness function. The creation of a new generation of individuals involves three fundamental phases: Selection, Crossover, and Mutation. In the Selection phase (line 7 in Alg. 7), two individual parents are randomly chosen from the population for mating. The probability of selecting a

particular individual is often proportional to its fitness, giving preference to individuals with higher fitness values. During the Crossover (line 8 in Alg. 7), genetic material from the selected parents is used to generate offspring that will form the basis of the next generation. Finally, Mutation (line 15 in Algorithm 7) involves random alterations to one or more genes within an individual. This process further diversifies the population and explores different regions of the solution space.

A new generation is produced by iteratively applying the Selection, Recombination, and Mutation phases until the entire population of individuals in the new generation replaces those in the old one. An effective genetic algorithm strikes a balance between maintaining genetic quality (favoring individuals with high fitness) and preserving genetic diversity to support an efficient and thorough search.

Algorithm 7 Genetic Algorithm (GA)

1: **Input:** Population size $n_{\text{population}}$, Number of generations $n_{\text{generations}}$, Crossover probability $p_{\text{crossover}}$, Mutation probability p_{mutation}
2: **Output:** Best solution found
3: Initialize population with random solutions
4: **for** $i = 1$ **to** $n_{\text{generations}}$ **do**
5: Calculate fitness for each solution in the population
6: *best* $\leftarrow$ null
7: Randomly select k individuals from the population, with replacement
8: **for** selected individual in the population **do**
9: Select two parents based on fitness for crossover
10: **if** random number $< p_{\text{crossover}}$ **then**
11: Perform crossover to create offspring
12: **else**
13: Copy parents to create offspring
14: **end if**
15: **if** random$(0, 1) < p_{\text{mutation}}$ **then**
16: Apply mutation to the offspring
17: **end if**
18: Evaluate the fitness of the offspring
19: **if** fitness of offspring is better than fitness of *best* **then**
20: *best* $\leftarrow$ the offspring
21: **end if**
22: **end for**
23: Replace the current population with the new generation
24: **end for**
25: **return** Best solution found in the final population

5.4 System Model

We assume a partially synchronous communication network. A peer-to-peer distributed system of authenticated nodes communicates over the network and keeps a common state update. This state is essentially a replicated data structure shared in memory between replicas, *i.e.,* blockchain. Users of the blockchain send transactions to these nodes, which are then grouped together to form a block. To validate each block, the nodes must execute a voted-based consensus algorithm, performed by the nodes selected as Validators.

We consider that among the nodes, there is a maximum of f faulty nodes, which can either be Byzantine or fail due to technical issues, be victims of attacks, or for some other reason.

Additionally, we consider the presence of a node responsible for directing the voting and validation of the block, referred to as the Primary, similar to the concept in PBFT [169]. This Primary node proposes the initialization of a new block and rotates periodically. Once the Validator nodes are selected, the consensus algorithm based on voting begins. In the following, we describe the optimization problem involved in selecting the Validators in a way that maximizes the Social Welfare of the users.

5.4.1 Problem Statement

Let $U = \{u_1, u_2, \ldots, u_k\}$ be a set of k users with transactions in the block B, and $N = \{n_1, n_2, \ldots, n_S\}$ a set of all the S blockchain nodes. The problem $\mathcal{P}$ consists of finding a set V of m nodes to be Validator, where $V = \{v_1, v_2, \ldots, v_m\}$, $V \subseteq N$, which maximizes the Social Welfare among the k users.

Decision variable: A binary variable x_i indicates if node n_i is a Validator or not.

$$x_i = \begin{cases} 1, & \text{if } n_i \in V \\ 0, & otherwise \end{cases} \tag{5.4}$$

Objective: The objective is to maximize the Social welfare among the users. For this purpose, we consider α-Fairness as the utility function. Let W_α be the Social welfare for a giver α, and $\delta_{i,j}$ the normalized Transaction Confirmation Time (TCT) perceived by the user j from the node i. The Utility for the user j, denoted as $\mathcal{U}_j(X)$, is the inverse of the TCT perceived by user j, which is the maximum among all validation times with respect to each Validator node, *i.e.*, $\mathcal{U}_j(X) = \frac{1}{max\{x_i \delta_{i,j}\}}; \forall x_i \in X$.

$$W_\alpha = \begin{cases} \frac{1}{1-\alpha} \sum_{i=1}^{u} \left(\frac{1}{max\{x_i \delta_{i,j}\}} \right)^{1-\alpha}, & \alpha \neq 1 \\ \sum_{i=1}^{u} log \left(\frac{1}{max\{x_i \delta_{i,j}\}} \right), & \alpha = 1 \end{cases} \tag{5.5}$$

Problem formulation: With the constraint that x_i must be binary for all i and that exactly m Validators must be selected among the S blockchain nodes, the problem $\mathcal{P}$ can be formulated as the following optimization problem:

$$\begin{aligned} \max_{X} \quad & W_\alpha \\ \text{s.t.} \quad & \sum_i x_i = m \quad , \forall i \in \{1, 2, \ldots, S\}, \\ & x_i \in \{0, 1\}, \forall i \in \{1, 2, \ldots, S\} \end{aligned} \tag{5.6}$$

5.4.2 Problem Complexity

In this section, we show that problem $\mathcal{P}$ is at least as challenging as the hardest problems within the class NP (nondeterministic polynomial time).

Theorem 10. *The problem $\mathcal{P}$ is NP-hard.*

Proof. The complexity of problem $\mathcal{P}$ is demonstrated by a polynomial-time reduction from the well-established NP-hard P-Median Problem (PMP) [190].

P-Median Problem

For a directed weighted graph $G = (V, A, C)$, with a number of vertices $|V|$, set of arcs $(i, j) \in A \subseteq V \times V$, and weights (distances, similarities, etc.) $C = \{c(i, j) : (i, j) \in A\}$, the PMP consists of determining p nodes (the median nodes, $1 \leq p \leq |V|$) minimizing the total sum of weights to all other nodes of the graph.

Reduction

Let's map each location in PMP to a node and interpret the facility placement as the selection of nodes in problem $\mathcal{P}$ and the weights interpreted as the validation times between each node and each user. On the other hand, assigning a customer (user) to a facility (node) is interpreted as that node being the one with the longest validation time for that user. In addition, the cost minimization in the PMP aligns to maximize W_α in our problem. Minimization can be converted into maximization by multiplying by -1. With this transformation, the objective function of the PMP is equivalent to the objective function (See Equation 5.6) W_α of problem $\mathcal{P}$ with $\alpha = 0$, and different values of α not make the problem easier. Facilities are placed optimally to minimize costs, while nodes are selected optimally to maximize fairness.

Since all operations used in reduction are in polynomial time, the reduction shows that, if is possible to solve the problem of selecting m nodes among S to maximize the W_α in polynomial time, it is also possible to solve the PMP in polynomial time. Therefore, the $\mathcal{P}$ problem is at least as hard as the PMP problem, which is NP-hard. □

5.5 Proposed Solution for Fair Validator Selection Process

In this section, we describe the Proposed Validator Selection Algorithm, which addresses the problem of Validator selection in a distributed manner. As shown in Fig. 5.1, we divide the algorithm into three stages: Problem Initialization, Solution, and Aggregation. Each of these stages is detailed below.

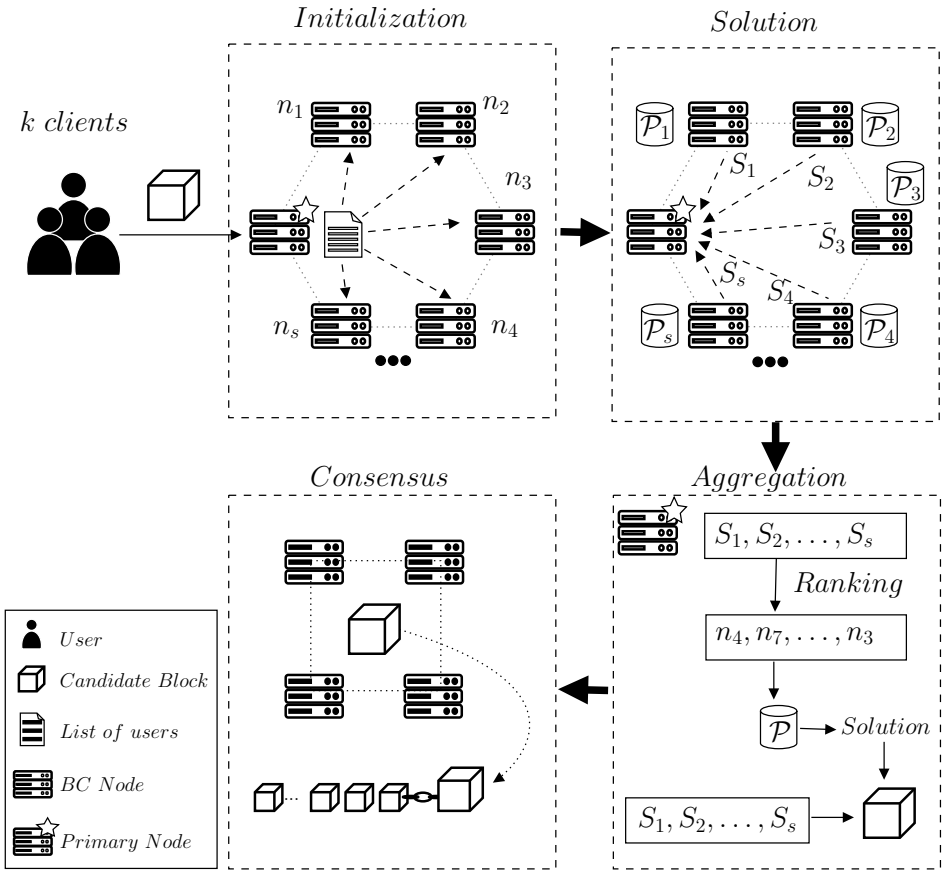

Figure 5.1: Proposed Validator Selection process

5.5.1 Problem Initialization

In this stage, the Primary Node sends the list of users in the current block to each of the nodes through a broadcast message, including the digital signature of the Primary Node. With this information, the nodes identify the list of users participating in the problem. As each node receives this message and verifies the authentication of the digital signature, it begins to address the corresponding optimization problem.

5.5.2 Distributed solution

Given the vast search space when the number of nodes is large, inspired by the algorithm proposed in [191], we employ a strategy of dividing the search space. We partition the main problem $\mathcal{P}$ into S subproblems, $\mathcal{P}_1$, $\mathcal{P}_2, \ldots, \mathcal{P}_S$, and assign a distinct search space to each subproblem. In contrast to the approach described in [191], where the space division is designed for parallel processing by different (High-Performance Computing) HPC servers, our algorithm achieves a distributed solution by assigning each subproblem to an individual node.

The division of the search space is performed in a round-robin fashion, in block number b, each subproblem $\mathcal{P}_i$ involves finding the m nodes that maximize W_α in the search space containing nodes from $(b+i) \mod S$ to $(b+i+T) \mod S$, where T is the size of the search space for each subproblem. In the proposed algorithm, we use $T = \min(3m, S)$ to ensure that faulty nodes do not affect the result, assuming that the Consensus algorithm between the Validator resists a maximum of f faulty nodes. This value of T guarantees that each node appears in $\lfloor \frac{S}{3m} \rfloor$ subproblems.

Once each node establishes the parameters of its assigned subproblem, it begins solving it. In our algorithm, we assume that each subproblem is sufficiently large and should be solved using heuristic methods. For this purpose, we employ the evolutionary algorithms described in Section 5.3.2. Once node i finds the solution S_i to subproblem i, it sends S_i to the Primary Node with the respective digital signature. The Primary Node collects the solutions from each node, verifies the authenticity of the signatures, and proceeds with the aggregation process.

5.5.3 Aggregation

This process involves only the Primary Node, which must select the m Validator nodes. For this purpose, the primary node sorts the found solutions from best to worst and starts verifying them in that order until it finds the $3f+1$ best-verified solutions. This avoids a collision attack and reduces processing time. Each node is then assigned a score based on how often it appears in the solutions of the $3m$ subproblems. Nodes that appear frequently in the solutions receive a high weight, while nodes that were never selected or are far away from other promising nodes receive very low or no weight. We consider unit weights, but weights that depend on the trust in each node could also be taken into account. Once all weights are calculated, all S nodes are sorted according to their weights, and the first $3m$ nodes with the highest weights are selected as candidates for the original problem. Subsequently, the Primary Node solves the problem once again, but this time using only the $3m$ candidate nodes.

Once the problem is solved by the Primary node, the m nodes in the solution will be the Validators for the current block. For consensus, one can employ one of the voting-based consensus mechanisms. To ensure transparency in the selection process, the list of the S solutions received by the Primary Node will be recorded in the block as a transaction and will be validated during the consensus.

5.6 Security Analysis

This section provides a security analysis of the proposed FairVSP, considering its resistance to potential attacks based on the assumed attack model and the requirements for a consensus protocol presented in Section 5.2.1.

In the proposed algorithm, the following assumptions are applied to analyze security against existing attacks:

- We utilize a Permissioned Blockchain, where a maximum of f nodes can fail.
- The applied consensus protocol withstands Byzantine attacks when there are at least f Byzantine nodes and satisfies the properties of Liveness, Safety, and Validity.
- The node chosen as the Primary for a block could be Byzantine.

Under those assumptions, we provide the proof that a vote-based consensus $\mathcal{C}$ that employs the proposed algorithm for the Validator Selection process FairVSP resists Sybil, Collusion, Eclipse, Denial-of-Service (DoS) Attacks, and maintains the properties of Liveness, Safety, and Validity, even when the primary node is Byzantine.

Lemma 11. *Consensus protocol $\mathcal{C}$ resists Sybil attacks.*

Proof. Sybil attacks involve creating multiple fake identities (nodes or Validators) to gain disproportionate influence in the Validator selection process. Attackers may use these fake identities to increase their chances of being selected as Validators, potentially undermining the security and decentralization of the blockchain. It is not possible for an adversary to create multiple fake identities in a Permissioned Blockchain since the nodes are well-known in this network. The proposed algorithm does not facilitate the creation of fake identities in any way, thus the consensus protocol remains resistant to Sybil attacks when using the proposed algorithm. □

Lemma 12. *Consensus protocol $\mathcal{C}$ resists Collusion attacks.*

Proof. Byzantine nodes may attempt to collude to influence the selection process in their favor. This may compromise the integrity of the set of Validators and affect the quality of service and fairness among users. For this attack, Byzantine nodes must include all Byzantine nodes found in their search subspace in the solution of the problem instance that corresponds to them, and find a solution that ranks among the $3m$ best solutions. Given the complexity of the problem, this attack is unlikely to succeed. In the worst case, when the attack succeeds, the consensus protocol will support up to f Byzantine nodes and thus maintain security and liveness properties. □

Lemma 13. *Consensus protocol $\mathcal{C}$ resists Eclipse Attacks and DoS Attacks.*

Proof. Eclipse attacks involve isolating a node by controlling its network connections. Attackers can manipulate the node's view of the network, potentially preventing it from participating in the Validator selection process or influencing its decisions. In both cases, the outcome of the selection is not significantly affected because the rest of the nodes participate. Since each node appears in $3m$ subproblems, Byzantine nodes must find a solution that includes them in the ranking, which is unlikely due to the complexity of the problem to be solved. DoS attacks target nodes with the aim of disrupting their operation and preventing them from participating in the selection process. Similarly, given the complexity of the problem, an attacker would need to simultaneously attack at least $m - f$ honest nodes, which is impractical if the network size is sufficiently large. The distributed nature of the proposed algorithm itself protects the Validator selection process from Eclipse Attacks and DoS Attacks. □

Lemma 14. *Consensus protocol $\mathcal{C}$ maintains Liveness, Safety, and Validity properties when the selected Primary Node is Byzantine.*

Proof. When the Primary Node is Byzantine, it may attempt to influence the Validator selection process and choose other Byzantine nodes for consensus. Since the solution to the final problem that the Primary Node solves during the Aggregation process is added to the blockchain, honest nodes participating in the consensus will detect the irregularity of the solution and will not validate the current block. If the selection of a node as the Primary occurs randomly with a uniform distribution, a Byzantine node is chosen as the primary with a probability of f/S, which could affect the performance of the system if it causes the block not to be accepted. Instead, an algorithm for selecting the primary node based on reputation would be more suitable. □

5.7 Simulation and Numerical Analysis

This section describes the implementation of the Evolutionary algorithms used for the distributed solution of the Validator Selection problem described in Section 5.5.3, comparing the distributed solution using the search space division strategy with the centralized solution. The numerical results of the solution quality and the convergence time in different scenarios are presented. Additionally, the numerical results of the simulations are analyzed in terms of the Price of Fairness, in addition to the analysis of the computational and communication cost of the proposed protocol.

5.7.1 Implementation

Python (version 3.8) is used to implement and simulate the distributed solution of different instances of the Validator Selection problem [1]. In these scenarios, the S sub-problems are solved independently. Subsequently, the $3m$ nodes that appear most frequently in the solutions are selected, and a final instance of the problem is solved with these selected nodes.

In addition to the proposed solution using Evolutionary algorithms ACO, SA, and GA, in addition, two baseline approaches are simulated for comparison: i) the case where Validator selection is based on a lottery, which is simulated as a random selection of Validators; and ii) sequential selection using the specific case that follows a round-robin fashion (as in PBFT [169]).

For the Evolutionary algorithms, two stopping criteria are considered: convergence of the solution, meaning that in 10 consecutive iterations, the objective function of the new solutions does not improve (less than $\epsilon = 0.001$), or reaching the maximum number of iterations. The solution is represented as a binary vector of n positions, where the position i of the vector is equal to 1 if the node i is a Validator or zero otherwise.

Ant Colony Optimization Implementation

The implementation of ACO follows the Alg. 6. In this case, each ant traverses exactly m nodes without replacement, and the utility function used is W_α. The *UpdatePheromone* function at the end of each ant's traversal updates the pheromone level of each path, depositing the pheromone on the paths that were part of the solution. This pheromone deposit (γ) is inversely proportional to the cost of the solution, *i.e.*, $\gamma = 1/solution\_cost$. Conversely, the *PheromoneEvaporation* function reduces the pheromone on all paths at a constant evaporation rate (θ), governed by the parameter θ; $0 < \theta < 1$, which is multiplied by the pheromone of each path at the end of each iteration.

[1] available: https://github.com/juliocpg/validators_paper.git

Simulated Annealing Implementation

In this case of SA, the pseudocode shown in Alg. 5 is implemented. For the *GenerateNeighbor* function, a set is formed with all vectors that are valid solutions, *i.e.,* having exactly m positions equal to one, which is at a Hamming distance of two bits from the current solution. Each of these neighboring solutions is then evaluated, and one is randomly selected based on the temperature and the improvement it represents over the current solution (see line 7 of Alg. 5). At the end of each iteration, the *CoolingSchedule* function decreases the temperature at a rate controlled by the parameter cooling rate (ϕ); $0 < \phi < 1$, which is multiplied by the current temperature.

Genetic Algorithm Implementation

As in the pseudocode of Alg. 7, in GA, a population of different solutions (individuals) is maintained, where the number of elements (chromosomes) equal to 1 is exactly m. *Crossover* function must ensure that the offspring maintain this property. Two parents (solution vectors) are selected, and a vector is generated with all positions where both parents have chromosomes equal to 1. Subsequently, m values are randomly chosen from the list, creating a new individual with chromosomes equal to 1 in the selected m positions and 0 in the remaining positions. Additionally, for the *Mutation* function, from the individual to be mutated, two lists are created: one with the positions of the chromosomes equal to 1 and the other with the 0. A value is randomly selected from each list, and both chromosomes are swapped. This ensures that the new individual has m chromosomes of 1.

Baselines Implementation

In our baseline use for comparison, we employ traditional approaches for the selection of Validators: the random selection, denoted as Random, and the sequential rotating selection of Validators, for which we use a rotating strategy following a round-robin fashion, denoted as Round-Robin. For the implementation of the Random approach, m nodes are randomly chosen from the total S to be Validators. The selection process employs a uniform distribution in which each node has an equal probability $\left(\frac{m}{S}\right)$ of being selected as a Validator. On the other hand, for the implementation of the Round-Robin approach, for block i, nodes are selected from $(i \mod S)$ to $((i+m) \mod S)$ in a cyclic manner. Fig. 5.2 illustrates an example using the Round-Robin fashion in which 3 Validators are selected from 4 nodes. In this example, for the first block, nodes N_1, N_2, and N_3 are chosen, for the second block, N_2, N_3, and N_4, and so forth.

5.7.2 Simulation Setup

The network is simulated as a fully connected graph, generated randomly for each problem instance. The adjacency matrix of every graph contains the normalized validation times perceived by each user for each node, which follow a uniform distribution in the interval $(0.01, 1)$. Once each graph is generated for each scenario, the same graph is used to find the solutions with the different algorithms studied. Table 5.2 shows the configuration parameters used. The number of nodes in the network is increased, ranging from 1000 to 4000 with a step of 1000. In each case, 100, 200, and 300 Validators are selected from the total of nodes. Since the value of α does not directly affect the performance of the algorithms, for the evaluation of the distributed solution quality and convergence time it is set to 3. The next section presents the numerical results of the quality of the solution and the convergence time of the Evolutionary algorithms using the search space division strategy described in Section 5.5.3, the centralized solution, and the baselines. In the case of baselines, the average of metrics is computed from 10^6 runs on each problem, *i.e.,* graph. For a confidence level of 95%, this number of iterations guarantees

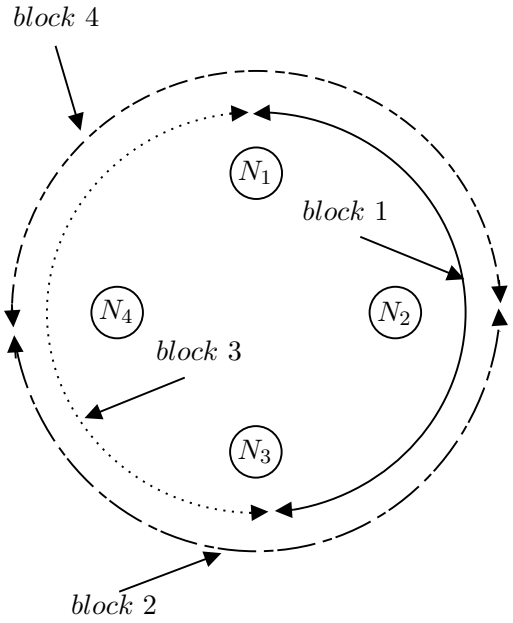

Figure 5.2: Example of Validator selection using Round-Robin fashion for select $m = 3$ Validators among $S = 4$ nodes.

that the maximum error in metric estimation is below 5%. All simulations were run on a laptop with Intel Core $i7 - 7700$ CPU $3.60GHz \times 8$, and $16GB$ of RAM.

For the Evolutionary algorithms, two stopping criteria are considered: convergence of the solution, meaning that in 10 consecutive iterations, the objective function of the new solutions does not improve (less than $\epsilon = 0.001$), or reaching the maximum number of iterations. The solution is represented as a binary vector of n positions, where the position i of the vector is equal to 1 if the node i is a Validator or zero otherwise.

The probabilistic theory of records is a mathematical framework that deals with the extreme values or "records" observed in a sequence of independent random samples from a probability distribution. In optimization problems, particularly in the field of operations research, obtaining a new incumbent solution refers to finding a new best solution to a given problem [192]. According to this theory [193], [194] the probability of obtaining a record in iteration n is $1/n$ and, consequently, the probability of obtaining it in h consecutive iterations immediately after iteration n is equal to $h/(n+h)$. Therefore if the property is to stop the algorithm when the probability of improvement in h consecutive iterations is lower than η:

$$N = \left\lfloor \frac{1-\eta}{\eta} h \right\rfloor + 1 \tag{5.7}$$

In our case with $N = 1000$ iterations, from Eq. 5.7 we obtain that with $h = 10$ iterations it is sufficient to ensure a probability $\eta = 0.001$.

5.7.3 Distributed Solution Quality and Convergence Time

Given that the optimal solution is not known for the simulated scenarios, we evaluate the solution quality using the optimality gap relative to the centralized solution and relative to the best-known solution (the best found by all the algorithms). The relative optimality gap, denoted as Δ_r, serves as a metric for quantifying the disparity between the objective value of the reference solution and the objective value of the solution obtained through an algorithm. It offers valuable insights into how closely the algorithm's solution aligns with the reference solution.

Table 5.2: Simulation setup

Description	**Values**
Blockchain Nodes (S)	$[1000, 2000, 3000, 4000]$
Validators (m)	$[300, 500, 800]$
Users (k)	200
Parameter of α-Fairness (α)	3
Max. iterations	1000
Normalized Validation time($\delta_{i,j}$)	$\delta_{i,j} \sim \mathrm{U}(0.01, 1)$
ACO Parameters	
Number of Ants	100
Pheromone exponent	0.5
Utility exponent	0.5
Evaporation rate	0.1
Elitist probability	0.3
SA Parameters	
Initial Temperature	100
Cooling rate	0.8
GA Parameters	
Population size	100
Crossover probability	0.8
Mutation probability	0.1

Mathematically, the relative optimality gap is defined as:

$$\Delta_r = \frac{f(X_r) - f(X)}{f(X_r)} \tag{5.8}$$

Here, X_r represents the relative solution and X represents the value of the solution obtained through the algorithm optimization process. A lower Δ_r indicates that the algorithm solution is closer to the relative solution. Ideally, a smaller Δ_r suggests that the algorithm has found a solution that is closer to the optimal solution, demonstrating its effectiveness in solving the optimization problem. We denote as Δ_C the relative optimality gap with respect to the centralized solution of the same algorithm and as Δ_B relative to the best-known solution.

In addition to the solution's quality, we measure the convergence time of the algorithms in each scenario. The convergence time of a heuristic algorithm quantifies the speed at which the algorithm approaches a satisfactory or optimal solution for a given problem. It serves as a fundamental performance indicator, indicating how quickly the algorithm converges towards a solution of acceptable quality or optimal in the best case.

Table 5.3 shows the results of the simulations in terms of W_α, Δ_B, Δ_C, and the convergence time of the algorithms for all the simulated scenarios. The baselines are denoted as Round-robin and Random in Table 5.3, and the scenarios that use the proposed solution with the algorithm ACO, GA, and SA are denoted as FairVSP_ACO, FairVSP_GA, and FairVSP_SA respectively.

Table 5.3: Solution Quality and Convergence Time Comparison

		S											
		1000			**2000**			**3000**			**4000**		
	m	**100**	**200**	**300**	**100**	**200**	**300**	**100**	**200**	**300**	**100**	**200**	**300**
FairVSP_ACO	W	**−28.01**	**−26.93**	**−22.77**	−26.74	−25.47	−23.81	**−25.28**	−24.03	−23.06	−24.51	**−22.36**	**−21.73**
	$\Delta_C(\%)$	10.0	9.5	9.2	9.3	8.2	7.5	7.9	6.3	5.3	7.3	6.1	5.0
	$\Delta_B(\%)$	0	0	0	5.6	0.2	8.3	0	0.2	1.5	2.8	0	0
	$t(s)$	0.127	0.246	0.516	0.230	0.433	1.004	0.305	0.691	1.480	0.702	1.372	1.874
FairVSP_GA	W	−30.30	-28.83	−26.29	−27.76	−24.85	−23.72	−28.7	−25.47	−23.59	−24.88	−24.37	−23.72
	$\Delta_C(\%)$	12.9	11.5	10.2	11.9	10.2	9.2	9.3	9.1	8.4	8.1	7.6	6.3
	$\Delta_B(\%)$	8.2	7.1	15.5	5.6	0.2	8.3	13.5	6.2	3.8	4.4	9.0	9.2
	$t(s)$	0.352	0.665	0.683	0.984	1.004	1.386	1.005	1.019	1.118	1.130	1.437	1.876
FairVSP_SA	W	−29.84	−27.49	−26.27	**−26.28**	**−24.89**	**−21.9**	-25.93	**−23.98**	**−22.73**	**−23.84**	-23.33	−21.97
	$\Delta_C(\%)$	7.6	6.2	5.1	6.8	5.9	5.2	6.5	6.3	4.9	5.9	4.7	4.5
	$\Delta_B(\%)$	6.5	2.1	15.4	0	0	0	2.6	0	0	0	4.3	1.1
	$t(s)$	0.035	0.060	0.119	0.186	0.268	0.354	0.154	0.917	1.0872	0.302	0.715	1.619
Round-robin	W	−48.51	−49.11	−45.12	−48.67	−49.28	−49.51	−48.68	−49.36	−49.52	−48.93	−49.27	−49.60
	$\Delta_B(\%)$	73.2	82.4	98.2	85.2	98.0	126.1	92.6	105.8	117.9	105.2	120.3	128.3
Random	W	−59.02	−59.99	−58.95	−59.27	−60.27	−60.47	−59.38	−60.14	−60.45	−59.19	−60.09	−60.52
	$\Delta_B(\%)$	110.7	122.8	158.9	125.5	142.1	176.1	134.9	150.79	165.95	148.3	168.7	178.5

Centralized vs. Distributed

To compute Δ_C, each problem is solved for all nodes without applying the search space division strategy, which is equivalent to solving the problem in a centralized manner. As observed in Table 5.3, in the FairVSP_ACO, FairVSP_GA, and FairVSP_SA methods, as the number of nodes in the network increases, Δ_C increases. For instance, in scenarios solved using ACO, with 1000, 2000, 3000, and 4000 nodes to select 100 Validators, the Δ_C is 10, 9.3, 7.9 and 7.3 with improvements of 0.7, 1.4 and 0.6 percent respectively. This behavior is due to the fact that although an increase in the number of nodes expands the search space, the more nodes there are, the more times each node will be present in the subproblems, resulting in a greater number of subproblems to be solved. This results in a greater exploration of the search space, while the complexity of the problem remains the same for the centralized solution, thus favoring the distributed solution.

While the centralized solution obtains better solutions, the time required is much longer than solving a subproblem with a smaller search space. In the simulations, the time to find the centralized solution exceeded 540 seconds, which is excessively high for most IoT applications. However, with the proposed approach, in all scenarios, Δ_C is less than 13 percent, which demonstrates the effectiveness in solution quality of the search space partitioning strategy.

Distributed vs. Baselines

To calculate Δ_B, the best solution found among the three Evolutionary algorithms and the two algorithms used as baselines were employed as the reference. In Table 5.3, the best solutions found in each scenario are highlighted in bold. In most scenarios, the FairVSP_SA method exhibits the best performance, which can be attributed to its ability to escape local minima due to the high level of exploration it achieves. Nevertheless, the difference between the FairVSP_ACO and FairVSP_GA methods is not excessively large, being less than 15 percent in all cases. However, as can be observed, the baselines are in all cases more than 70 percent away from the best solutions, demonstrating the low levels of fairness associated with the utilization of classical algorithms.

Scalability

The increase in the number of nodes, as observed, improves the quality of the solutions found. However, increasing the number of Validators to be selected increases the computational cost of the algorithms, leading to a longer convergence time. Depending on the application, it is necessary to adjust the convergence time and reduce the search space to achieve faster results at the expense of solution quality. This trade-off must be carefully taken into account when determining the size of the Validator set. In all the simulated scenarios, the convergence time is less than 1.5 second for every subproblem, which is acceptable for many IoT applications.

5.7.4 Price of Fairness

The Price of Fairness (PoF) is a metric for quantifying the cost or penalty incurred when a fair selection (X_α) is made for a given α, compared to a more efficient but unfair selection (Utilitarian Solution, X_{util}). The PoF metric helps assess the trade-off between fairness and efficiency in decision-making processes [195]. Mathematically, the PoF can be defined as follows:

$$PoF_\alpha = \frac{\mathcal{U}(X_{util}) - \mathcal{U}(X_\alpha)}{\mathcal{U}(X_{util})} \tag{5.9}$$

Note that the PoF is a number between 0 and 1 because the sum of utilities under the utilitarian solution attains its maximum value. When the sum of utilities is an efficiency measure,

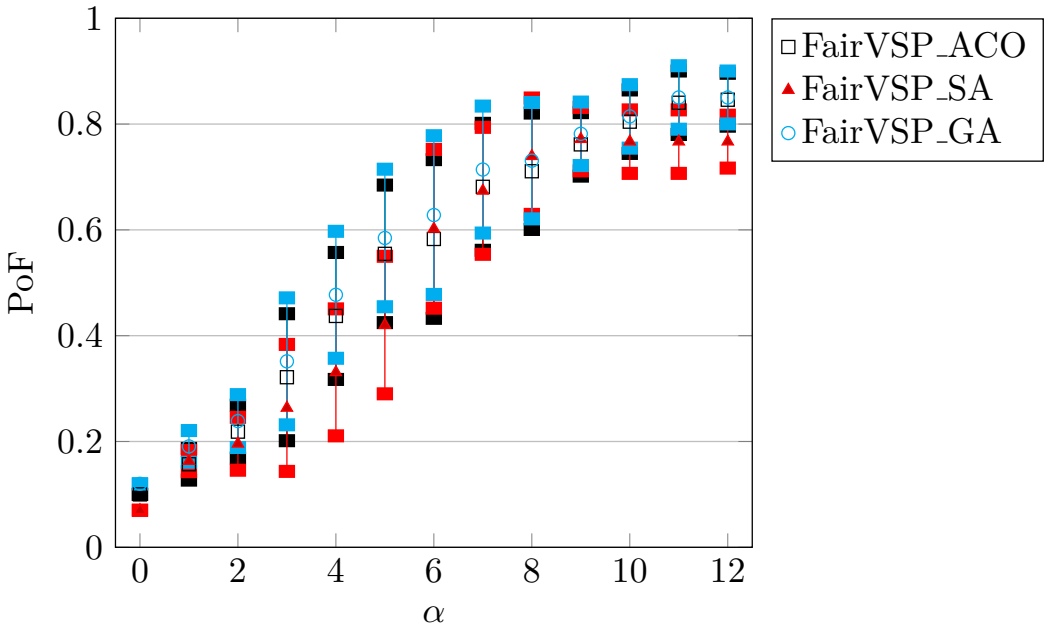

Figure 5.3: Impact of α on PoF

values closer to 0 are preferable for the PoF, and in other cases, the system combines efficiency and fairness.

For the calculation of the PoF, a scenario with 100 nodes, 30 Validators, and 50 users is employed. To obtain the Utilitarian solution in any given graph, it is imperative to the optimal solution for $\alpha = 0$, which is NP-hard (see Section 5.4.2). Consequently, we construct a graph in which the Utilitarian solution is known. To achieve this, a graph is generated in which the first 30 nodes possess the minimum normalized validation time (0.01) for all users. This ensures that it is the Utilitarian solution, meaning there is no alternative solution that minimizes the validation time more than when this set of the first 30 nodes is chosen as Validators. Analogous to previous experiments, the remainder of the adjacency matrix is randomly generated with values uniformly distributed in the interval $(0.01, 1)$. Once the graph is generated, 1000 runs are conducted for different values of α using each of the Evolutionary algorithms, employing the search space division strategy. The value of α ranged from 0 to 10 in increments of 1. Additionally, in each scenario, the difference between the utilities of each of the 50 users is calculated, and the greatest difference among all users is determined. This represents the utility difference between the user with the highest and lowest utility, *i.e.,* $max(\mathcal{U}(X_i) - \mathcal{U}(X_j)); \forall i \neq j)$.

Fig. 5.3 illustrates the simulation results in terms of PoF. It is evident that, on average, as α increases, PoF also increases. This phenomenon is due to the fact that the solutions are increasingly fairer and move away from the most effective and unfair solution. For all algorithms, when α is lower than 2, PoF is below 0.3. Additionally, the increase in utility difference observed in Fig. 5.4 significantly decreases with higher values of α, resulting in an increase in fairness. However, for α values exceeding 4, the utility difference between users approaches zero, and PoF exceeds 0.5, rendering the system inefficient. Given this analysis, the choice of α should be made with careful consideration of the desired levels of efficiency and fairness required by the application.

5.7.5 Computational and Communication Overhead

In addition to evaluating the performance of the algorithms, it is important to assess the computational and communication costs associated with the use of the proposed algorithm. The computational cost of the algorithm largely depends on the convergence time of the algorithms

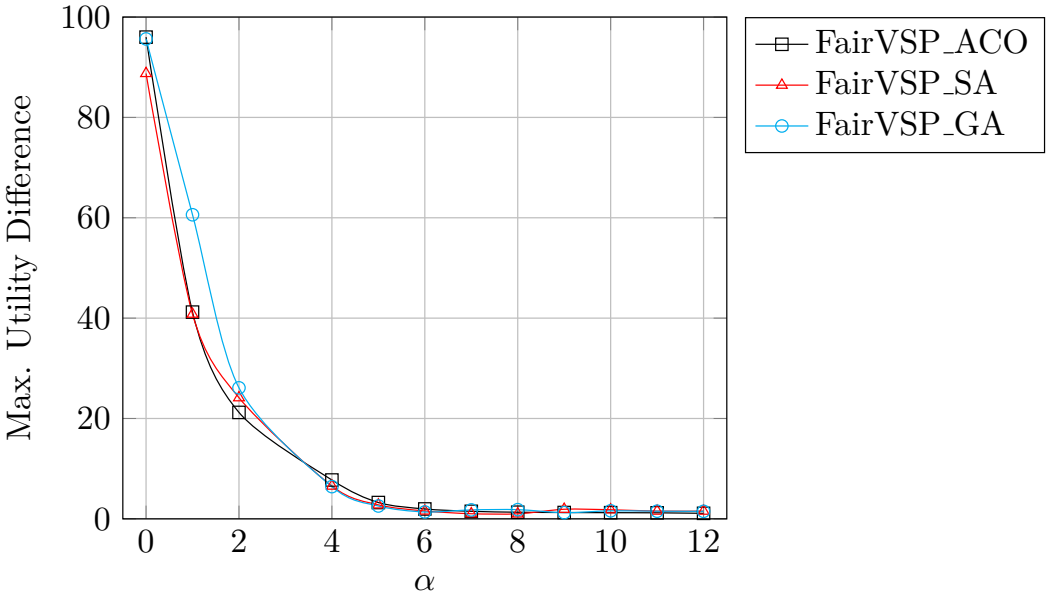

Figure 5.4: Impact of α on users utilities

used to find the solution. In any case, each node will solve an instance (subproblem) of the main problem in the Solution stage, and the Primary node will also solve an instance of the problem in the Aggregation stage. Therefore, the computational cost of applying the proposed algorithm for a given node is the cost of solving one instance of the problem. Note that each subproblem has a much smaller search space than the overall problem which improves the computational cost without losing too much in the quality of the solution as demonstrated above.

On the other hand, for communication costs, the messages exchanged during the algorithm must be considered. In the Initialization stage, the Primary node broadcasts a message with the list of users in the current block, resulting in 1 messages being sent and S received by the blockchain nodes. Subsequently, each node sends the found solution, adding another S sent and received message to the network. In conclusion, the number of required packets is $S + 1$ sent messages and $2S$ received, which grows linearly with the increase in S. This cost is similar to the communication cost of lottery-based approaches where an election round is required and S packets need to be sent and received.

5.8 Conclusion of the Chapter

Since most consensus protocols for Permissioned Blockchain use a predefined or randomly selected set of Validators, users do not perceive the same confirmation time for their transactions. This unfairness discourages user participation in the Blockchain. In this chapter, we introduce FairVSP, a novel algorithm for Validator Selection for Permissioned Blockchain, which maximizes the α-Fairness as a Social Welfare utility function. We provide a method for the distributed solution of the optimization problem from which Validator Selection derives. We also analyze the security, computational, and communication costs of the algorithm. We also study the quality of the solutions found through the Evolutionary Algorithms used, as well as the cost of using the algorithm in terms of Price of Fairness. These results demonstrate that the algorithm is secure and efficient, and in contrast to existing methods, guarantees fairness among users.

Chapter 6

Conclusion and Future Directions

6.1 Conclusion

The work delves into the integration of the Internet of Things and Blockchain technology, recognizing their transformative potential across various domains of daily life. The sensitivity of data in IoT applications necessitates robust security measures, especially in light of resource-constrained IoT devices. To this end, lightweight cryptographic techniques, particularly those based on elliptic curves, are identified as key enablers of efficient encryption methods within the IoT ecosystem. Centralized architectures, often employed in traditional security solutions, present scalability and security challenges within the dynamic IoT environment. The integration of Blockchain, characterized by its decentralized and immutable ledger, emerges as a promising avenue to address these concerns, offering a foundation for increased trust and transparency.

This work explores specific applications of Blockchain within IoT, addressing challenges related to authentication in drone-to-drone communication and group key management for efficient group communication. The proposed solutions leverage Blockchain decentralized nature to enhance security while maintaining efficiency. The Internet of Drones (IoD) manages and coordinates communications between drones in Internet of Things (IoT) applications. Ensuring security and privacy in Unmanned Aerial Vehicles (UAVs) networks, *i.e.,* drones, is essential to protect data from cyber-attacks. In this context, providing authentication is a major challenge due to the fact that drones are devices limited in power capabilities. The problem is aggravated by the dynamism of IoD networks due to the high mobility of drones, which are sensitive to packet loss and handovers. Blockchain technology is attractive to address the problem of centralization of existing authentication protocols. In this work, we provide a decentralized, secure, and efficient authentication protocol, based on μTesla, that relies on Blockchain to manage drone authentication. We analyze the security and performance of the proposed solution. Simulation results show that the proposed solution outperforms several approaches in the literature, achieving an authentication delay of less than 250 ms with a low information exchange of 1024 bits for a 128-bit security level while maintaining low computational requirements.

On the other hand, Group communications play a crucial role in enhancing the Quality of Service (QoS) of Internet of Things (IoT) networks, enabling efficient information dissemination while minimizing resource utilization. However, ensuring information security and privacy in IoT group communications necessitates the implementation of an efficient and lightweight key management scheme due to the limited capabilities of most IoT devices. This work presents a novel key management protocol for group communications that employs distributed Blockchain technology in IoT networks. The proposed scheme considers nodes belonging to multiple groups. By utilizing an asymmetric key shared among group members, secure communication is established between outsiders and group members while preserving anonymity inside the group. A distinguishing feature of the protocol is its combination of group member anonymity and automatic key revocation facilitated by a Smart Contract. Furthermore, simulation results demonstrate the efficiency of the proposed scheme, consuming less than 300 mJ of energy and taking less than 7 seconds to establish a group key among 1000 nodes, outperforming several existing approaches in the literature in terms of computation and communication costs

However, the integration of Blockchain and IoT brings with it many challenges, selecting appropriate consensus mechanisms tailored to specific applications becomes paramount, given

their profound impact on system performance. Striking the right balance between security, efficiency, and scalability remains a critical consideration. The consensus algorithm must ensure reliability, security, and transparency within the Blockchain, particularly in Permissioned Blockchains where participating nodes (i.e., Validators) are known but susceptible to attacks or lack of trust. Existing consensus protocols for blockchains primarily focus on increasing transaction processing speeds, without considering the fairness in the transaction validation time among blockchain users, thereby potentially disadvantaging some users or even denying access to the blockchain. This work introduces FairVSP, an algorithm for Validator selection in Permissioned Blockchains. In the proposed algorithm, blockchain nodes collectively maximize Social Welfare among users via the distributed solution of the Validator Selection problem using the Evolutionary algorithms, Ant Colony Optimization, Simulated Annealing, and Genetic algorithm. The model employs the well-known α-Fairness as the utility function and uses a search space division strategy to solve the problem in an efficient and distributed manner. We provide an analysis of the security aspects and the impact of the proposed algorithm on system performance. The numerical findings illustrate that the suggested algorithm effectively enhances Social Welfare for users, achieving improvements of over 70% when compared to existing methods. Moreover, the algorithm demonstrates rapid convergence, completing computations in under 2 seconds. These attributes collectively establish the proposed algorithm as both efficient and secure, making it suitable for a diverse set of Permissioned Blockchain applications.

In conclusion, this work underscores the imperative of robust security and privacy measures within IoT, advocating for the integration of Blockchain technology as a solution. It emphasizes the need for lightweight security solutions tailored to the resource constraints of IoT devices. Additionally, active efforts toward achieving fairness in Blockchain processes are identified as pivotal for fostering an equitable environment. The contributions made in authentication, group communication, and Validator Selection further enhance the potential for secure and efficient IoT applications enabled by Blockchain.

6.2 Future directions

The future directions outlined below represent key avenues of research and development at the nexus of Internet of Things (IoT) and Blockchain technologies. As these fields continue to evolve, addressing emerging challenges and exploring untapped potentials is paramount for realizing the full transformative impact of their integration. From scalability and privacy preservation to quantum-resilience and user-centric applications, the following directions offer a roadmap for researchers, industry practitioners, and policymakers to navigate this dynamic landscape and shape the trajectory of IoT-Blockchain convergence.

Scalability and Efficiency Optimization: As the IoT ecosystem continues to expand, there is a pressing need to further research and develop consensus mechanisms that strike an optimal balance between security, efficiency, and scalability. Investigating novel approaches, such as sharding or consensus protocols with reduced computational overhead, holds promise in this regard.

Security and Resilience against Quantum Threats: With the potential advent of quantum computing, it is imperative to anticipate and prepare for potential threats to existing cryptographic algorithms. Research into post-quantum cryptography and its integration with Blockchain for future-proofing security measures is a critical area of study.

Privacy-Preserving Techniques: As data privacy concerns continue to gain prominence, future research should delve into advanced cryptographic techniques and privacy-preserving protocols that ensure the confidentiality of sensitive information while still allowing for meaningful data analysis and utilization.

Energy-Efficient Blockchain Implementations: Given the resource constraints of many IoT devices, there is a need for energy-efficient implementations of Blockchain technology. This could involve exploring consensus mechanisms that require less computational power or investigating energy-saving techniques in the design and operation of Blockchain networks.

Diverse Domain Applications: A crucial future direction lies in the widespread application of the proposed solutions across diverse domains. Investigating how Blockchain-integrated IoT solutions can be tailored and optimized for specific industries, such as healthcare, logistics, smart cities, and agriculture, will unlock new realms of innovation and address domain-specific challenges. Understanding the unique requirements of each sector and adapting the technology accordingly will catalyze advancements in a wide range of industries.

By directing research efforts towards these future directions, we can further unlock the full potential of Blockchain-integrated IoT applications, advancing the security, efficiency, and usability of these transformative technologies.

Appendix A

A.1 Computing Simulation Error

In simulation studies, accurately assessing the error associated with average values is essential for gauging the reliability and precision of the results. If the average values of the metrics in the simulation follow a normal distribution, we can employ statistical techniques to estimate the error. With normality assumed, the standard error (SE) of the mean can be computed using:

$$SE = \frac{\sigma}{\sqrt{n}} \tag{A.1}$$

where σ represents the standard deviation of the sample averages and n denotes the sample size (number of simulations). The confidence interval (CI) for the average value can be determined using the equation:

$$CI = \bar{x} \pm z_\alpha \times SE \tag{A.2}$$

here $\bar{x}$ is the sample mean of the average values and z_α is the critical value from the standard normal distribution corresponding to the desired confidence level α(e.g., 1.96 for a 95% confidence level). The confidence interval provides a range within which the true average value is likely to fall, given the assumed normal distribution of the data. A narrower confidence interval indicates higher precision in estimating the true average.

We verify in all simulations the assumption of normality by conducting a normality test on the metric sample averages in different scenarios experiments using the well-known Kolmogorov-Smirnov test. In this test, if the p-value associated with the test is greater than a chosen significance level (we use 0.05), it provides evidence in support of the normality assumption. Figure A.1 shows the Matlab function used to run all the tests.

```
function [result] = performKSTest(sample_means, hypothesized_distribution)
    % Perform Kolmogorov-Smirnov test
    [h, ~, kstat, cv] = kstest(sample_means, 'CDF', hypothesized_distribution);

    % Display the test results
    disp('Kolmogorov-Smirnov Test Results:');
    disp(['Hypothesized Distribution: ' hypothesized_distribution]);
    disp(['Test Statistic (D): ' num2str(kstat, '%.4f')]);
    disp(['Critical Value at 5% significance level: ' num2str(cv, '%.4f')]);
    disp(['P-value: ' num2str(p, '%.4f')]);

    % Interpret the results
    if h == 0
        result = 'Do not reject the null hypothesis';
    else
        result = 'Reject the null hypothesis';
    end

end
```

Figure A.1: Matlab code Kolmogorov-Smirnov test used in simulations.

After verifying the normality of the simulation means, Eq. A.2 was used iteratively to calculate the confidence interval for a given number of simulations, computing the sample standard

deviation and deciding whether more simulations are required. Based on this procedure, an approximate number of simulations is calculated for each scenario. In all cases, upon completing the simulations of the scenarios, the confidence interval is recalculated and checked to ensure it falls within the initially assumed parameters.

Appendix B

B.1 Python implementation of ACO Class

```
class ACO:
    def __init__(self, p,graph, num_ants, evaporation_rate, alpha_ant, beta, q0,alpha):
        self.graph = graph
        self.num_ants = num_ants
        self.evaporation_rate = evaporation_rate
        self.alpha_ant = alpha_ant
        self.beta = beta
        self.q0 = q0
        self.p = p
        self.alpha = alpha
        self.pheromone_matrix = [[1 / graph.num_servers] * graph.num_servers for _ in range(graph.
            num_servers+1)]

    def run(self, iterations):
        start_time = time.time()
        best_solution = None
        best_solution_cost = float('inf')

        for _ in range(iterations):
            solutions = []
            for _ in range(self.num_ants):
                solution = self.construct_solution()
                solutions.append(solution)

            for ant_solution in solutions:
                solution_cost = self.evaluate_solution(ant_solution)
                if solution_cost < best_solution_cost:
                    best_solution_cost = solution_cost
                    best_solution = ant_solution

                self.update_pheromone(ant_solution, solution_cost)

        convergence_time = time.time() - start_time
        return np.array(best_solution), best_solution_cost, convergence_time/iterations

    def construct_solution(self):

        init_vertice = random.randint(0,self.graph.num_servers-1)
        remaining_vertices = list(range(self.graph.num_servers))

        remaining_vertices.remove(init_vertice) # Start with the first vertex in the list

        solution = [init_vertice]

        while len(solution) < self.p:
            next_vertex = self.choose_next_vertex(remaining_vertices, solution)
            solution.append(next_vertex)
            remaining_vertices.remove(next_vertex)

        return solution

    def choose_next_vertex(self, remaining_vertices, solution):
        probabilities = []
        last_vertex = solution[-1]

        for vertex in remaining_vertices:

            p_path = self.graph.adj_matrix[last_vertex][vertex]
            if(p_path > 0):
```

```
                probability = (self.pheromone_matrix[last_vertex][vertex] ** self.alpha_ant) * \
                              ((1/p_path) ** self.beta)
            else:
                probability = 0.000001

            if(not np.isnan(float(probability))):
                probabilities.append(probability)

            else:
                probabilities.append(0.000001)
                print("NAN")

        if random.random() < self.q0:
            max_probability = max(probabilities)
            max_probability_index = probabilities.index(max_probability)
            return list(remaining_vertices)[max_probability_index]
        else:
            total_probabilities = sum(probabilities)
            probabilities = [p / total_probabilities for p in probabilities]
            #print(probabilities)
            return np.random.choice(list(remaining_vertices),size=1,p=probabilities)[0]

    def evaluate_solution(self, solution):
        solution_cost = socialWelfare(self.graph,solution,alpha=self.alpha) # Penalty cost
        return solution_cost

    def update_pheromone(self, solution, solution_cost):
        pheromone_deposit = 1 / solution_cost
        for i in range(len(solution) - 1):
            from_vertex = solution[i]
            to_vertex = solution[i + 1]
            self.pheromone_matrix[from_vertex][to_vertex] += pheromone_deposit
            self.pheromone_matrix[to_vertex][from_vertex] += pheromone_deposit

        self.pheromone_matrix = [[(1 - self.evaporation_rate) * pheromone + self.evaporation_rate / self.
             graph.num_servers
                                  for pheromone in row]
                                 for row in self.pheromone_matrix]
```

B.2 Python implementation of GA Class

```
class GA:
    def __init__(self,p, graph, population_size, mutation_rate, crossover_rate, elitism_rate,alpha):
        self.graph = graph
        self.population_size = population_size
        self.mutation_rate = mutation_rate
        self.crossover_rate = crossover_rate
        self.elitism_rate = elitism_rate
        self.p = p
        self.alpha = alpha

    def run(self, iterations):
        start_time = time.time()

        population = self.generate_initial_population()

        best_solution = None
        best_solution_cost = float('inf')

        for _ in range(iterations):
            population = self.evolve_population(population)

            best_individual = min(population, key=lambda ind: self.evaluate_solution(ind))
            if self.evaluate_solution(best_individual) < best_solution_cost:
                best_solution_cost = self.evaluate_solution(best_individual)
                best_solution = best_individual

        convergence_time = time.time() - start_time
        return np.array(best_solution), best_solution_cost, convergence_time/iterations
```

```
    def generate_initial_population(self):
        population = []
        for _ in range(self.population_size):
            individual = random.sample(range(self.graph.num_servers), self.p)
            population.append(individual)
        return population

    def evolve_population(self, population):
        new_population = []

        elitism_size = max(1, int(self.elitism_rate * self.population_size))
        elitism_pool = sorted(population, key=lambda ind: self.evaluate_solution(ind))[:elitism_size]
        new_population.extend(elitism_pool)

        while len(new_population) < self.population_size:
            if random.random() < self.crossover_rate and len(population)>2:
                parent1, parent2 = random.sample(population, k=2)
                offspring1, offspring2 = self.crossover(parent1, parent2)
                new_population.extend([offspring1, offspring2])
            if random.random() < self.mutation_rate:
                ind = random.choice(population)
                mutatedInd = self.mutate(ind)
                new_population.append(mutatedInd)


        return new_population

    def crossover(self,parent1, parent2):
        # Select a random crossover point
        crossover_point = random.randint(1, len(parent1) - 1)

        # Create the first child by combining the genes of parent1 and parent2
        child1 = parent1[:crossover_point] + parent2[crossover_point:]

        # Create the second child by combining the genes of parent2 and parent1
        child2 = parent2[:crossover_point] + parent1[crossover_point:]

        # Remove duplicate genes in the children
        child1 = list(set(child1))
        child2 = list(set(child2))

        # Fill in missing genes in the children
        missing_genes1 = list(set(parent1) - set(child1))
        missing_genes2 = list(set(parent2) - set(child2))

        for i in range(self.p - len(child1)):
            child1.append(missing_genes1[i])
        for i in range(self.p - len(child2)):
            child2.append(missing_genes2[i])

        return child1, child2


    def mutate(self, individual):

        # Generate a new number between 1 and n is not in the list
        available_numbers =[num for num in range(self.graph.num_servers) if num not in individual]
        if len(available_numbers) < 2:
            raise ValueError("Not␣enough␣numbers␣available␣in␣the␣range.")
        else:
            new_gens = random.sample(available_numbers,2)


        index1, index2 = random.sample(range(self.p), 2)
        individual[index1], individual[index2] = new_gens[0], new_gens[1]

        return individual

    def evaluate_solution(self, solution):
        solution_cost = socialWelfare(self.graph,solution,alpha=self.alpha) # Penalty cost
        return solution_cost
```

B.3 Python implementation of SA Class

```
class SA:
def __init__(self,p, graph, initial_temperature, cooling_rate,alpha):
    self.graph = graph
    self.initial_temperature = initial_temperature
    self.cooling_rate = cooling_rate
    self.alpha = alpha
    self.p = p

def run(self, iterations):
    start_time = time.time()

    current_solution = random.sample(range(self.graph.num_servers),self.p)
    current_solution_cost = self.evaluate_solution(current_solution)

    best_solution = current_solution
    best_solution_cost = current_solution_cost

    temperature = self.initial_temperature

    for _ in range(iterations):
        new_solution = self.neighbor_solution(current_solution)
        new_solution_cost = self.evaluate_solution(new_solution)

        if new_solution_cost < current_solution_cost or \
                random.random() < np.exp((current_solution_cost - new_solution_cost) / temperature):
            current_solution = new_solution
            current_solution_cost = new_solution_cost

        if new_solution_cost < best_solution_cost:
            best_solution = new_solution
            best_solution_cost = new_solution_cost

        temperature *= self.cooling_rate

    convergence_time = time.time() - start_time
    return np.array(best_solution), best_solution_cost, convergence_time/iterations

def neighbor_solution(self, solution):

    n = int(0.2 * self.p)
    # Generate a neighbor solution by making a small modification
    new_solution = solution[:]

    available_numbers = list(set(range(self.graph.num_servers)) - set(solution))
    indexes = random.sample(range(self.p), n)

    for i in range(n):
        new_solution[indexes[i]] = available_numbers[i]

    return new_solution

def evaluate_solution(self, solution):
    solution_cost = socialWelfare(self.graph,solution,alpha=self.alpha) # Penalty cost
    return solution_cost
```

Bibliography

[1] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, "Internet of things: A survey on enabling technologies, protocols, and applications", *IEEE Communications Surveys & Tutorials*, vol. 17, no. 4, pp. 2347–2376, 2015. DOI: 10.1109/COMST.2015.2444095.

[2] J. Lin, W. Yu, N. Zhang, X. Yang, H. Zhang, and W. Zhao, "A survey on internet of things: Architecture, enabling technologies, security and privacy, and applications", *IEEE Internet of Things Journal*, vol. 4, no. 5, pp. 1125–1142, 2017. DOI: 10.1109/JIOT.2017.2683200.

[3] M. Chiang and T. Zhang, "Fog and iot: An overview of research opportunities", *IEEE Internet of Things Journal*, vol. 3, no. 6, pp. 854–864, 2016. DOI: 10.1109/JIOT.2016.2584538.

[4] H.-N. Dai, Z. Zheng, and Y. Zhang, "Blockchain for internet of things: A survey", *IEEE Internet of Things Journal*, vol. 6, no. 5, pp. 8076–8094, 2019. DOI: 10.1109/JIOT.2019.2920987.

[5] M. Salimitari and M. Chatterjee, "A survey on consensus protocols in blockchain for iot networks", *arXiv preprint arXiv:1809.05613*, 2018.

[6] E. Danna, A. Hassidim, H. Kaplan, *et al.*, "Upward max-min fairness", *Journal of the ACM (JACM)*, vol. 64, no. 1, pp. 1–24, 2017.

[7] J. C. Perez-Garcia, A. Benslimane, A. Braeken, and Z. Su, "μTesla-based authentication for reliable and secure broadcast communications in iod using blockchain", *IEEE Internet of Things Journal*, 2023.

[8] J. C. Perez-Garcia, A. Benslimane, and Z. Su, "Analysis on the aoi in blockchain-based iot networks with different sensing mechanisms", in *ICC 2022 - IEEE International Conference on Communications*, 2022, pp. 4763–4768. DOI: 10.1109/ICC45855.2022.9838842.

[9] J. C. Perez-Garcia, A. Benslimane, and S. Boutalbi, "Blockchain-based system for e-voting using blind signature protocol", in *2021 IEEE Global Communications Conference (GLOBECOM)*, 2021, pp. 01–06. DOI: 10.1109/GLOBECOM46510.2021.9685189.

[10] S. Boutalbi, J. C. Perez-Garcia, and A. Benslimane, "Blockchain-based secure handover for iot using zero-knowledge proof protocol", in *2021 IEEE Global Communications Conference (GLOBECOM)*, 2021, pp. 1–6. DOI: 10.1109/GLOBECOM46510.2021.9685733.

[11] K. Ashton, "That "internet of things" thing", *RFID journal*, vol. 22, no. 7, pp. 97–114, 2009.

[12] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of things (iot): A vision, architectural elements, and future directions", *Future generation computer systems*, vol. 29, no. 7, pp. 1645–1660, 2013.

[13] L. D. Xu, E. L. Xu, and L. Li, "Industry 4.0: State of the art and future trends", *International journal of production research*, vol. 56, no. 8, pp. 2941–2962, 2018.

[14] M. Elkahlout, M. M. Abu-Saqer, A. F. Aldaour, A. Issa, and M. Debeljak, "Iot-based healthcare and monitoring systems for the elderly: A literature survey study", in *2020 International Conference on Assistive and Rehabilitation Technologies (iCareTech)*, 2020, pp. 92–96. DOI: 10.1109/iCareTech49914.2020.00025.

[15] V. P. Kour and S. Arora, "Recent developments of the internet of things in agriculture: A survey", *IEEE Access*, vol. 8, pp. 129 924–129 957, 2020. DOI: 10.1109/ACCESS.2020.3009298.

[16] G. Verma and V. Sharma, "A novel rf energy harvester for event-based environmental monitoring in wireless sensor networks", *IEEE Internet of Things Journal*, vol. 9, no. 5, pp. 3189–3203, 2022. DOI: 10.1109/JIOT.2021.3097629.

[17] S. S. Hajam and S. A. Sofi, "Iot-fog architectures in smart city applications: A survey", *China Communications*, vol. 18, no. 11, pp. 117–140, 2021. DOI: 10.23919/JCC.2021.11.009.

[18] V. Hassija, V. Chamola, V. Gupta, S. Jain, and N. Guizani, "A survey on supply chain security: Application areas, security threats, and solution architectures", *IEEE Internet of Things Journal*, vol. 8, no. 8, pp. 6222–6246, 2021. DOI: 10.1109/JIOT.2020.3025775.

[19] A. Aldahmani, B. Ouni, T. Lestable, and M. Debbah, "Cyber-security of embedded iots in smart homes: Challenges, requirements, countermeasures, and trends", *IEEE Open Journal of Vehicular Technology*, vol. 4, pp. 281–292, 2023. DOI: 10.1109/OJVT.2023.3234069.

[20] N. Sharma and R. D. Garg, "Real-time iot-based connected vehicle infrastructure for intelligent transportation safety", *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, no. 8, pp. 8339–8347, 2023. DOI: 10.1109/TITS.2023.3263271.

[21] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, "Internet of things: A survey on enabling technologies, protocols, and applications", *IEEE Communications Surveys & Tutorials*, vol. 17, no. 4, pp. 2347–2376, 2015.

[22] K.-A. Shim, "A survey of public-key cryptographic primitives in wireless sensor networks", *IEEE Communications Surveys & Tutorials*, vol. 18, no. 1, pp. 577–601, 2016. DOI: 10.1109/COMST.2015.2459691.

[23] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, "Fog computing and its role in the internet of things", in *Proceedings of the first edition of the MCC workshop on Mobile cloud computing*, 2012, pp. 13–16.

[24] V. Galetić, I. Bojić, M. Kušek, G. Ježić, S. Dešić, and D. Huljenić, "Basic principles of machine-to-machine communication and its impact on telecommunications industry", in *2011 proceedings of the 34th international convention MIPRO*, IEEE, 2011, pp. 380–385.

[25] G. Sun, S. Huang, W. Bao, Y. Yang, and Z. Wang, "A privacy protection policy combined with privacy homomorphism in the internet of things", in *2014 23rd International Conference on Computer Communication and Networks (ICCCN)*, IEEE, 2014, pp. 1–6.

[26] C. Li and B. Palanisamy, "Privacy in internet of things: From principles to technologies", *IEEE Internet of Things Journal*, vol. 6, no. 1, pp. 488–505, 2018.

[27] 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*, vol. 21, no. 2, pp. 1676–1717, 2018.

[28] M. Erol-Kantarci and H. T. Mouftah, "Energy-efficient information and communication infrastructures in the smart grid: A survey on interactions and open issues", *IEEE Communications Surveys & Tutorials*, vol. 17, no. 1, pp. 179–197, 2014.

[29] G. White, V. Nallur, and S. Clarke, "Quality of service approaches in iot: A systematic mapping", *Journal of Systems and Software*, vol. 132, pp. 186–203, 2017.

[30] A. Musaddiq, Y. B. Zikria, O. Hahm, H. Yu, A. K. Bashir, and S. W. Kim, "A survey on resource management in iot operating systems", *IEEE Access*, vol. 6, pp. 8459–8482, 2018.

[31] M. Mohammadi and A. Al-Fuqaha, "Enabling cognitive smart cities using big data and machine learning: Approaches and challenges", *IEEE Communications Magazine*, vol. 56, no. 2, pp. 94–101, 2018.

[32] A. Bröring, S. Schmid, C.-K. Schindhelm, *et al.*, "Enabling iot ecosystems through platform interoperability", *IEEE software*, vol. 34, no. 1, pp. 54–61, 2017.

[33] V. Gazis, "A survey of standards for machine-to-machine and the internet of things", *IEEE Communications Surveys & Tutorials*, vol. 19, no. 1, pp. 482–511, 2016.

[34] D. S. Nunes, P. Zhang, and J. S. Silva, "A survey on human-in-the-loop applications towards an internet of all", *IEEE Communications Surveys & Tutorials*, vol. 17, no. 2, pp. 944–965, 2015.

[35] J. Zhou, Z. Cao, X. Dong, and A. V. Vasilakos, "Security and privacy for cloud-based iot: Challenges", *IEEE Communications Magazine*, vol. 55, no. 1, pp. 26–33, 2017.

[36] A. Barki, A. Bouabdallah, S. Gharout, and J. Traore, "M2m security: Challenges and solutions", *IEEE Communications Surveys & Tutorials*, vol. 18, no. 2, pp. 1241–1254, 2016.

[37] J. Sengupta, S. Ruj, and S. D. Bit, "A comprehensive survey on attacks, security issues and blockchain solutions for iot and iiot", *Journal of Network and Computer Applications*, vol. 149, p. 102 481, 2020.

[38] M. N. Aman, K. C. Chua, and B. Sikdar, "A light-weight mutual authentication protocol for iot systems", in *GLOBECOM 2017-2017 IEEE Global Communications Conference*, IEEE, 2017, pp. 1–6.

[39] T. Gomes, F. Salgado, A. Tavares, and J. Cabral, "Cute mote, a customizable and trustable end-device for the internet of things", *IEEE Sensors Journal*, vol. 17, no. 20, pp. 6816–6824, 2017.

[40] P. Porambage, C. Schmitt, P. Kumar, A. Gurtov, and M. Ylianttila, "Pauthkey: A pervasive authentication protocol and key establishment scheme for wireless sensor networks in distributed iot applications", *International Journal of Distributed Sensor Networks*, vol. 10, no. 7, p. 357 430, 2014.

[41] X. Hei, X. Du, J. Wu, and F. Hu, "Defending resource depletion attacks on implantable medical devices", in *2010 IEEE Global Telecommunications Conference GLOBECOM 2010*, IEEE, 2010, pp. 1–5.

[42] S. Sicari, A. Rizzardi, D. Miorandi, and A. Coen-Porisini, "Reato: Reacting to denial of service attacks in the internet of things", *Computer Networks*, vol. 137, pp. 37–48, 2018.

[43] J. Liu, C. Zhang, and Y. Fang, "Epic: A differential privacy framework to defend smart homes against internet traffic analysis", *IEEE Internet of Things Journal*, vol. 5, no. 2, pp. 1206–1217, 2018. DOI: 10.1109/JIOT.2018.2799820.

[44] U. Guin, A. Singh, M. Alam, J. Canedo, and A. Skjellum, "A secure low-cost edge device authentication scheme for the internet of things", in *2018 31st International Conference on VLSI Design and 2018 17th International Conference on Embedded Systems (VLSID)*, IEEE, 2018, pp. 85–90.

[45] G. Glissa, A. Rachedi, and A. Meddeb, "A secure routing protocol based on rpl for internet of things", in *2016 IEEE Global Communications Conference (GLOBECOM)*, 2016, pp. 1–7. DOI: 10.1109/GLOCOM.2016.7841543.

[46] C. Pu and S. Hajjar, "Mitigating forwarding misbehaviors in rpl-based low power and lossy networks", in *2018 15th IEEE Annual Consumer Communications & Networking Conference (CCNC)*, 2018, pp. 1–6. DOI: 10.1109/CCNC.2018.8319164.

[47] C. Cervantes, D. Poplade, M. Nogueira, and A. Santos, "Detection of sinkhole attacks for supporting secure routing on 6lowpan for internet of things", in *2015 IFIP/IEEE International Symposium on Integrated Network Management (IM)*, 2015, pp. 606–611. DOI: 10.1109/INM.2015.7140344.

[48] M. Singh, M. Rajan, V. Shivraj, and P. Balamuralidhar, "Secure mqtt for internet of things (iot)", in *2015 Fifth International Conference on Communication Systems and Network Technologies*, 2015, pp. 746–751. DOI: 10.1109/CSNT.2015.16.

[49] N. Park and N. Kang, "Mutual authentication scheme in secure internet of things technology for comfortable lifestyle", *Sensors*, vol. 16, no. 1, p. 20, 2015.

[50] D. Airehrour, J. A. Gutierrez, and S. K. Ray, "Sectrust-rpl: A secure trust-aware rpl routing protocol for internet of things", *Future Generation Computer Systems*, vol. 93, pp. 860–876, 2019.

[51] Y. Ashibani and Q. H. Mahmoud, "An efficient and secure scheme for smart home communication using identity-based signcryption", in *2017 IEEE 36th International Performance Computing and Communications Conference (IPCCC)*, 2017, pp. 1–7. DOI: 10.1109/PCCC.2017.8280497.

[52] V. Adat and B. B. Gupta, "A ddos attack mitigation framework for internet of things", in *2017 International Conference on Communication and Signal Processing (ICCSP)*, 2017, pp. 2036–2041. DOI: 10.1109/ICCSP.2017.8286761.

[53] D. Yin, L. Zhang, and K. Yang, "A ddos attack detection and mitigation with software-defined internet of things framework", *IEEE Access*, vol. 6, pp. 24 694–24 705, 2018. DOI: 10.1109/ACCESS.2018.2831284.

[54] A. J. Menezes, P. C. Van Oorschot, and S. A. Vanstone, *Handbook of applied cryptography*. CRC press, 2018.

[55] D. Hankerson and A. Menezes, "Elliptic curve cryptography", in *Encyclopedia of Cryptography, Security and Privacy*, Springer, 2021, pp. 1–2.

[56] P. Porambage, A. Braeken, and C. Schmitt, "Public key based protocols–ec crypto", *IoT Security: Advances in Authentication*, pp. 85–99, 2020.

[57] M. Liyanage, A. Braeken, P. Kumar, and M. Ylianttila, *IoT security: Advances in authentication*. John Wiley & Sons, 2020.

[58] M. Campagna, "Sec 4: Elliptic curve qu-vanstone implicit certificate scheme (ecqv)", *Standards for Efficient Cryptography, Version*, vol. 1, 2013.

[59] P. Porambage, C. Schmitt, P. Kumar, A. Gurtov, and M. Ylianttila, "Two-phase authentication protocol for wireless sensor networks in distributed iot applications", in *2014 IEEE Wireless Communications and Networking Conference (WCNC)*, Ieee, 2014, pp. 2728–2733.

[60] P. Porambage, P. Kumar, C. Schmitt, A. Gurtov, and M. Ylianttila, "Certificate-based pairwise key establishment protocol for wireless sensor networks", in *2013 IEEE 16th International Conference on Computational Science and Engineering*, IEEE, 2013, pp. 667–674.

[61] T. M. Hewa, Y. Hu, M. Liyanage, S. S. Kanhare, and M. Ylianttila, "Survey on blockchain-based smart contracts: Technical aspects and future research", *IEEE Access*, vol. 9, pp. 87 643–87 662, 2021. DOI: 10.1109/ACCESS.2021.3068178.

[62] V. Y. Kemmoe, W. Stone, J. Kim, D. Kim, and J. Son, "Recent advances in smart contracts: A technical overview and state of the art", *IEEE Access*, vol. 8, pp. 117 782–117 801, 2020. DOI: 10.1109/ACCESS.2020.3005020.

[63] N. Szabo, "Formalizing and securing relationships on public networks", *First monday*, 1997.

[64] J. A. Fairfield, "Smart contracts, bitcoin bots, and consumer protection", *Wash. & Lee L. Rev. Online*, vol. 71, p. 35, 2014.

[65] S. Omohundro, "Cryptocurrencies, smart contracts, and artificial intelligence", *AI matters*, vol. 1, no. 2, pp. 19–21, 2014.

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

[67] M. Crosby, P. Pattanayak, S. Verma, V. Kalyanaraman, *et al.*, "Blockchain technology: Beyond bitcoin", *Applied Innovation*, vol. 2, no. 6-10, p. 71, 2016.

[68] J. R. Douceur, "The sybil attack", in *International workshop on peer-to-peer systems*, Springer, 2002, pp. 251–260.

[69] M. Niranjanamurthy, B. Nithya, and S. Jagannatha, "Analysis of blockchain technology: Pros, cons and swot", *Cluster Computing*, vol. 22, pp. 14 743–14 757, 2019.

[70] M. Raynal, *Communication and agreement abstractions for fault-tolerant asynchronous distributed systems*. Springer Nature, 2022.

[71] W. Wang, D. T. Hoang, P. Hu, *et al.*, "A survey on consensus mechanisms and mining strategy management in blockchain networks", *Ieee Access*, vol. 7, pp. 22 328–22 370, 2019.

[72] S. Bano, A. Sonnino, M. Al-Bassam, *et al.*, "Consensus in the age of blockchains", *arXiv preprint arXiv:1711.03936*, 2017.

[73] M. Salimitari, M. Chatterjee, M. Yuksel, and E. Pasiliao, "Profit maximization for bitcoin pool mining: A prospect theoretic approach", in *2017 IEEE 3rd international conference on collaboration and internet computing (CIC)*, IEEE, 2017, pp. 267–274.

[74] J. Debus, "Consensus methods in blockchain systems", *Frankfurt School of Finance & Management, Blockchain Center, Tech. Rep*, 2017.

[75] L. Chen, L. Xu, N. Shah, Z. Gao, Y. Lu, and W. Shi, "On security analysis of proof-of-elapsed-time (poet)", in *Stabilization, Safety, and Security of Distributed Systems: 19th International Symposium, SSS 2017, Boston, MA, USA, November 5–8, 2017, Proceedings 19*, Springer, 2017, pp. 282–297.

[76] M. Salimitari, M. Chatterjee, and Y. P. Fallah, "A survey on consensus methods in blockchain for resource-constrained iot networks", *Internet of Things*, vol. 11, p. 100 212, 2020.

[77] R. Shostak, M. Pease, and L. Lamport, "The byzantine generals problem", *ACM Transactions on Programming Languages and Systems*, vol. 4, no. 3, pp. 382–401, 1982.

[78] D. Mingxiao, M. Xiaofeng, Z. Zhe, W. Xiangwei, and C. Qijun, "A review on consensus algorithm of blockchain", in *2017 IEEE international conference on systems, man, and cybernetics (SMC)*, IEEE, 2017, pp. 2567–2572.

[79] S. Popov, "The tangle white paper. 2018", *URL: https://iota. org/IOTA Whitepaper. pdf*, 2017.

[80] W. F. Silvano and R. Marcelino, "Iota tangle: A cryptocurrency to communicate internet-of-things data", *Future generation computer systems*, vol. 112, pp. 307–319, 2020.

[81] S. Benouar and A. Benslimane, "Robust blockchain for iot security", in *2019 IEEE Global Communications Conference (GLOBECOM)*, 2019, pp. 1–6. DOI: 10.1109/GLOBECOM38437.2019.9013580.

[82] M. Aazam and E.-N. Huh, "Fog computing and smart gateway based communication for cloud of things", in *2014 International conference on future Internet of Things and cloud*, IEEE, 2014, pp. 464–470.

[83] A. Reyna, C. Martın, J. Chen, E. Soler, and M. Dıaz, "On blockchain and its integration with iot. challenges and opportunities", *Future generation computer systems*, vol. 88, pp. 173–190, 2018.

[84] S.-C. Cha, J.-F. Chen, C. Su, and K.-H. Yeh, "A blockchain connected gateway for ble-based devices in the internet of things", *ieee access*, vol. 6, pp. 24 639–24 649, 2018.

[85] B. Confais, A. Lebre, and B. Parrein, "An object store service for a fog/edge computing infrastructure based on ipfs and a scale-out nas", in *2017 IEEE 1st International Conference on Fog and Edge Computing (ICFEC)*, IEEE, 2017, pp. 41–50.

[86] J. Kang, R. Yu, X. Huang, and Y. Zhang, "Privacy-preserved pseudonym scheme for fog computing supported internet of vehicles", *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 8, pp. 2627–2637, 2017.

[87] M. Dıaz, C. Martın, and B. Rubio, "State-of-the-art, challenges, and open issues in the integration of internet of things and cloud computing", *Journal of Network and Computer applications*, vol. 67, pp. 99–117, 2016.

[88] A. Asayag, G. Cohen, I. Grayevsky, *et al.*, "A fair consensus protocol for transaction ordering", in *2018 IEEE 26th International Conference on Network Protocols (ICNP)*, IEEE, 2018, pp. 55–65.

[89] A. Miller, Y. Xia, K. Croman, E. Shi, and D. Song, "The honey badger of bft protocols", in *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, 2016, pp. 31–42.

[90] Y. Sokolik and O. Rottenstreich, "Age-aware fairness in blockchain transaction ordering", in *2020 IEEE/ACM 28th International Symposium on Quality of Service (IWQoS)*, IEEE, 2020, pp. 1–9.

[91] K. Lev-Ari, A. Spiegelman, I. Keidar, and D. Malkhi, "Fairledger: A fair blockchain protocol for financial institutions", *arXiv preprint arXiv:1906.03819*, 2019.

[92] M. Kelkar, F. Zhang, S. Goldfeder, and A. Juels, "Order-fairness for byzantine consensus", in *Advances in Cryptology–CRYPTO 2020: 40th Annual International Cryptology Conference, CRYPTO 2020, Santa Barbara, CA, USA, August 17–21, 2020, Proceedings, Part III 40*, Springer, 2020, pp. 451–480.

[93] K. Kursawe, "Wendy, the good little fairness widget: Achieving order fairness for blockchains", in *Proceedings of the 2nd ACM Conference on Advances in Financial Technologies*, 2020, pp. 25–36.

[94] R. K. Jain, D.-M. W. Chiu, W. R. Hawe, *et al.*, "A quantitative measure of fairness and discrimination", *Eastern Research Laboratory, Digital Equipment Corporation, Hudson, MA*, vol. 21, 1984.

[95] J. Liu, W. Li, G. O. Karame, and N. Asokan, "Toward fairness of cryptocurrency payments", *IEEE Security & Privacy*, vol. 16, no. 3, pp. 81–89, 2018. DOI: 10.1109/MSP.2018.2701163.

[96] T. Lan, D. Kao, M. Chiang, and A. Sabharwal, *An axiomatic theory of fairness in network resource allocation*. IEEE, 2010.

[97] V. Chen and J. Hooker, "Welfare-based fairness through optimization", *arXiv preprint arXiv:2102.00311*, 2021.

[98] M. Yahuza, M. Y. I. Idris, I. B. Ahmedy, *et al.*, "Internet of drones security and privacy issues: Taxonomy and open challenges", *IEEE Access*, vol. 9, pp. 57 243–57 270, 2021. DOI: 10.1109/ACCESS.2021.3072030.

[99] M. A. Khan, S. Abbas, A. Rehman, *et al.*, "A machine learning approach for blockchain-based smart home networks security", *IEEE Network*, vol. 35, no. 3, pp. 223–229, 2021. DOI: 10.1109/MNET.011.2000514.

[100] Y. Tan, J. Wang, J. Liu, and N. Kato, "Blockchain-assisted distributed and lightweight authentication service for industrial unmanned aerial vehicles", *IEEE Internet of Things Journal*, pp. 1–13, 2022. DOI: 10.1109/JIOT.2022.3142251.

[101] O. Samuel, N. Javaid, A. Khalid, M. Imrarn, and N. Nasser, "A trust management system for multi-agent system in smart grids using blockchain technology", in *GLOBECOM 2020 - 2020 IEEE Global Communications Conference*, 2020, pp. 1–6. DOI: 10.1109/GLOBECOM42002.2020.9348231.

[102] S. Aggarwal, N. Kumar, and S. Tanwar, "Blockchain-envisioned uav communication using 6g networks: Open issues, use cases, and future directions", *IEEE Internet of Things Journal*, vol. 8, no. 7, pp. 5416–5441, 2021. DOI: 10.1109/JIOT.2020.3020819.

[103] M. A. Khan, I. Ullah, N. Kumar, *et al.*, "An Efficient and Secure Certificate-Based Access Control and Key Agreement Scheme for Flying Ad-Hoc Networks", *IEEE Transactions on Vehicular Technology*, vol. 70, no. 5, pp. 4839–4851, 2021, ISSN: 19399359. DOI: 10.1109/TVT.2021.3055895.

[104] M. W. Akram, A. K. Bashir, S. Shamshad, *et al.*, "A Secure and Lightweight Drones-Access Protocol for Smart City Surveillance", *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–10, 2021, ISSN: 15580016. DOI: 10.1109/TITS.2021.3129913.

[105] T. Hewa, A. Braeken, M. Ylianttila, and M. Liyanage, "Blockchain-based Automated Certificate Revocation for 5G IoT", *IEEE International Conference on Communications*, vol. 2020-June, 2020, ISSN: 15503607. DOI: 10.1109/ICC40277.2020.9148820.

[106] C. Lin, D. He, N. Kumar, X. Huang, P. Vijayakumar, and K. K. R. Choo, "Homechain: A blockchain-based secure mutual authentication system for smart homes", *IEEE Internet of Things Journal*, vol. 7, no. 2, pp. 818–829, 2020, ISSN: 23274662. DOI: 10.1109/JIOT.2019.2944400.

[107] C. Lin, D. He, X. Huang, K.-K. R. Choo, and A. V. Vasilakos, "Bsein: A blockchain-based secure mutual authentication with fine-grained access control system for industry 4.0", *Journal of Network and Computer Applications*, vol. 116, pp. 42–52, 2018, ISSN: 1084-8045. DOI: https://doi.org/10.1016/j.jnca.2018.05.005. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1084804518301619.

[108] Y. Tan, J. Wang, J. Liu, and N. Kato, "Blockchain-assisted distributed and lightweight authentication service for industrial unmanned aerial vehicles", *IEEE Internet of Things Journal*, pp. 1–1, 2022. DOI: 10.1109/JIOT.2022.3142251.

[109] C. Feng, B. Liu, Z. Guo, K. Yu, Z. Qin, and K.-K. R. Choo, "Blockchain-based cross-domain authentication for intelligent 5g-enabled internet of drones", *IEEE Internet of Things Journal*, vol. 9, no. 8, pp. 6224–6238, 2022. DOI: 10.1109/JIOT.2021.3113321.

[110] A. Perrig, R. Szewczyk, J. D. Tygar, V. Wen, and D. E. Culler, "Spins: Security protocols for sensor networks", *Wireless networks*, vol. 8, no. 5, pp. 521–534, 2002.

[111] M. Wazid, A. K. Das, and J.-H. Lee, "Authentication protocols for the internet of drones: Taxonomy, analysis and future directions", *Journal of Ambient Intelligence and Humanized Computing*, pp. 1–10, 2018.

[112] Y. Tian, J. Yuan, and H. Song, "Efficient privacy-preserving authentication framework for edge-assisted Internet of Drones", *Journal of Information Security and Applications*, vol. 48, p. 102 354, 2019, ISSN: 2214-2126. DOI: https://doi.org/10.1016/j.jisa.2019.06.010. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2214212618307038.

[113] M. Barbareschi, V. Casola, A. De Benedictis, E. L. Montagna, and N. Mazzocca, "On the adoption of physically unclonable functions to secure iiot devices", *IEEE Transactions on Industrial Informatics*, vol. 17, no. 11, pp. 7781–7790, 2021. DOI: 10.1109/TII.2021.3059656.

[114] A. Perrig, R. Canetti, J. Tygar, and D. Song, "Efficient authentication and signing of multicast streams over lossy channels", in *Proceeding 2000 IEEE Symposium on Security and Privacy. S P 2000*, 2000, pp. 56–73. DOI: 10.1109/SECPRI.2000.848446.

[115] L. Wang, Y. Tian, and D. Zhang, "Toward Cross-Domain Dynamic Accumulator Authentication Based on Blockchain in Internet of Things", *IEEE Transactions on Industrial Informatics*, vol. 18, no. 4, pp. 2858–2867, 2022, ISSN: 19410050. DOI: 10.1109/TII.2021.3116049.

[116] X. Yang, X. Yang, X. Yi, *et al.*, "Blockchain-Based Secure and Lightweight Authentication for Internet of Things", *IEEE Internet of Things Journal*, vol. 9, no. 5, pp. 3321–3332, 2022, ISSN: 23274662. DOI: 10.1109/JIOT.2021.3098007.

[117] S. Boutalbi, J. C. P. García, and A. Benslimane, "Blockchain-based secure handover for iot using zero-knowledge proof protocol", in *2021 IEEE Global Communications Conference (GLOBECOM)*, 2021, pp. 1–6. DOI: 10.1109/GLOBECOM46510.2021.9685733.

[118] Z. Haddad, M. Baza, M. M. Mahmoud, W. Alasmary, and F. Alsolami, "Secure and Efficient AKA Scheme and Uniform Handover Protocol for 5G Network Using Blockchain", *IEEE Open Journal of the Communications Society*, vol. 2, no. December, pp. 2616–2627, 2021, ISSN: 2644125X. DOI: 10.1109/OJCOMS.2021.3131552.

[119] Y. Yu, Y. Zhao, Y. Li, X. Du, L. Wang, and M. Guizani, "Blockchain-Based Anonymous Authentication with Selective Revocation for Smart Industrial Applications", *IEEE Transactions on Industrial Informatics*, vol. 16, no. 5, pp. 3290–3300, 2020, ISSN: 19410050. DOI: 10.1109/TII.2019.2944678.

[120] M. Shen, H. Liu, L. Zhu, *et al.*, "Blockchain-Assisted Secure Device Authentication for Cross-Domain Industrial IoT", *IEEE Journal on Selected Areas in Communications*, vol. 38, no. 5, pp. 942–954, 2020, ISSN: 15580008. DOI: 10.1109/JSAC.2020.2980916.

[121] D. S. Gupta, A. Karati, W. Saad, and D. B. Da Costa, "Quantum-Defended Blockchain-Assisted Data Authentication Protocol for Internet of Vehicles", *IEEE Transactions on Vehicular Technology*, vol. 71, no. 3, pp. 3255–3266, 2022, ISSN: 19399359. DOI: 10.1109/TVT.2022.3144785.

[122] A. Yazdinejad, R. M. Parizi, A. Dehghantanha, H. Karimipour, G. Srivastava, and M. Aledhari, "Enabling Drones in the Internet of Things with Decentralized Blockchain-Based Security", *IEEE Internet of Things Journal*, vol. 8, no. 8, pp. 6406–6415, 2021, ISSN: 23274662. DOI: 10.1109/JIOT.2020.3015382.

[123] B. Liu, K. Yu, C. Feng, and K. K. R. Choo, "Cross-domain authentication for 5G-enabled UAVs: A Blockchain approach", *DroneCom 2021 - Proceedings of the 4th ACM MobiCom Workshop on Drone Assisted Wireless Communications for 5G and Beyond*, pp. 25–30, 2021. DOI: 10.1145/3477090.3481053.

[124] C.-M. Chen, X. Deng, W. Gan, J. Chen, and S. H. Islam, "A secure blockchain-based group key agreement protocol for iot", *The Journal of Supercomputing*, vol. 77, pp. 9046–9068, 2021.

[125] A. Perrig, J. Tygar, A. Perrig, and J. Tygar, "Tesla broadcast authentication", *Secure Broadcast Communication: In Wired and Wireless Networks*, pp. 29–53, 2003.

[126] G. Choudhary, V. Sharma, I. You, K. Yim, I.-R. Chen, and J.-H. Cho, "Intrusion detection systems for networked unmanned aerial vehicles: A survey", in *2018 14th International Wireless Communications & Mobile Computing Conference (IWCMC)*, 2018, pp. 560–565. DOI: 10.1109/IWCMC.2018.8450305.

[127] A. Perrig, D. Song, R. Canetti, J. Tygar, and B. Briscoe, "Timed efficient stream loss-tolerant authentication (tesla): Multicast source authentication transform introduction", *Request For Comments*, vol. 4082, 2005.

[128] B. Blanchet, V. Cheval, and V. Cortier, "Proverif with lemmas, induction, fast subsumption, and much more", in *2022 IEEE Symposium on Security and Privacy (SP)*, 2022, pp. 69–86. DOI: 10.1109/SP46214.2022.9833653.

[129] H. Lamrani Alaoui, A. El Ghazi, M. Zbakh, A. Touhafi, and A. Braeken, "A highly efficient ecc-based authentication protocol for rfid", *Journal of Sensors*, p. 8 876 766, 2021.

[130] M. T. Damir, T. Meskanen, S. Ramezanian, and V. Niemi, "A beyond-5g authentication and key agreement protocol", in *Network and System Security*, X. Yuan, G. Bai, C. Alcaraz, and S. Majumdar, Eds., Springer Nature Switzerland, 2022, pp. 249–264.

[131] H. Krawczyk, M. Bellare, and R. Canetti, *Hmac: Keyed-hashing for message authentication*, 1997.

[132] F. Meneghello, M. Calore, D. Zucchetto, M. Polese, and A. Zanella, "Iot: Internet of threats? a survey of practical security vulnerabilities in real iot devices", *IEEE Internet of Things Journal*, vol. 6, no. 5, pp. 8182–8201, 2019. DOI: 10.1109/JIOT.2019.2935189.

[133] J. Park, M. Jung, and E. P. Rathgeb, "Survey for secure iot group communication", in *2019 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*, 2019, pp. 1026–1031. DOI: 10.1109/PERCOMW.2019.8730750.

[134] T. Chen, L. Zhang, K.-K. R. Choo, R. Zhang, and X. Meng, "Blockchain-based key management scheme in fog-enabled iot systems", *IEEE Internet of Things Journal*, vol. 8, no. 13, pp. 10 766–10 778, 2021. DOI: 10.1109/JIOT.2021.3050562.

[135] V. S. Naresh, V. D. Allavarpu, and S. Reddi, "Provably secure blockchain privacy-preserving smart contract centric dynamic group key agreement for large wsn", *The Journal of Supercomputing*, pp. 1–25, 2022.

[136] J. Li, Z. Qiao, and J. Peng, "Asymmetric group key agreement protocol based on blockchain and attribute for industrial internet of things", *IEEE Transactions on Industrial Informatics*, vol. 18, no. 11, pp. 8326–8335, 2022. DOI: 10.1109/TII.2022.3176048.

[137] Q. Zhang, B. Wang, X. Zhang, J. Yuan, and R. Wang, "Blockchain-based dynamic group key agreement protocol for ad hoc network", *Chinese Journal of Electronics*, vol. 29, no. 2020-3-447, p. 447, 2020. DOI: 10.1049/cje.2020.02.020.

[138] A. Braeken, "Pairing free asymmetric group key agreement protocol", *Computer Communications*, vol. 181, pp. 267–273, 2022, ISSN: 0140-3664. DOI: https://doi.org/10.1016/j.comcom.2021.10.011. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0140366421003893.

[139] P. Porambage, A. Braeken, C. Schmitt, A. Gurtov, M. Ylianttila, and B. Stiller, "Group key establishment for enabling secure multicast communication in wireless sensor networks deployed for iot applications", *IEEE Access*, vol. 3, pp. 1503–1511, 2015. DOI: 10.1109/ACCESS.2015.2474705.

[140] T. Salman, M. Zolanvari, A. Erbad, R. Jain, and M. Samaka, "Security services using blockchains: A state of the art survey", *IEEE communications surveys & tutorials*, vol. 21, no. 1, pp. 858–880, 2018.

[141] C.-M. Chen, X. Deng, W. Gan, J. Chen, and S. H. Islam, "A secure blockchain-based group key agreement protocol for iot", *The Journal of Supercomputing*, vol. 77, pp. 9046–9068, 2021.

[142] X. Li, Y. Wang, P. Vijayakumar, D. He, N. Kumar, and J. Ma, "Blockchain-based mutual-healing group key distribution scheme in unmanned aerial vehicles ad-hoc network", *IEEE Transactions on Vehicular Technology*, vol. 68, no. 11, pp. 11 309–11 322, 2019. DOI: 10.1109/TVT.2019.2943118.

[143] G. Heo, K. Chae, and I. Doh, "Hierarchical blockchain-based group and group key management scheme exploiting unmanned aerial vehicles for urban computing", *IEEE Access*, vol. 10, pp. 27 990–28 003, 2022. DOI: 10.1109/ACCESS.2022.3157753.

[144] X. Li and X. Yin, "Blockchain-based group key agreement protocol for vehicular ad hoc networks", *Computer Communications*, vol. 183, pp. 107–120, 2022.

[145] M. A. Shawky, A. Jabbar, M. Usman, *et al.*, "Efficient blockchain-based group key distribution for secure authentication in vanets", *IEEE Networking Letters*, pp. 1–1, 2023. DOI: 10.1109/LNET.2023.3234491.

[146] X. Hu, X. Song, G. Cheng, *et al.*, "Towards efficient co-audit of privacy-preserving data on consortium blockchain via group key agreement", in *2021 17th International Conference on Mobility, Sensing and Networking (MSN)*, 2021, pp. 494–501. DOI: 10.1109/MSN53354.2021.00079.

[147] Z. Xu, F. Li, H. Deng, M. Tan, J. Zhang, and J. Xu, "A blockchain-based authentication and dynamic group key agreement protocol", *Sensors*, vol. 20, no. 17, 2020, ISSN: 1424-8220. DOI: 10.3390/s20174835. [Online]. Available: https://www.mdpi.com/1424-8220/20/17/4835.

[148] Y. B. Taçyıldız, O. Ermiş, G. Gür, and F. Alagöz, "Dynamic group key agreement for resource-constrained devices using blockchains", in *Applied Cryptography and Network Security Workshops: ACNS 2020 Satellite Workshops, AIBlock, AIHWS, AIoTS, Cloud S&P, SCI, SecMT, and SiMLA, Rome, Italy, October 19–22, 2020, Proceedings 18*, Springer, 2020, pp. 58–76.

[149] M. Campagna, "Standards for efficient cryptography sec 4: Elliptic curve qu-vanstone implicit certificate scheme (ecqv)", *Certicom Corp*, 2013.

[150] A. Braeken, J.-J. Chin, and S.-Y. Tan, "Ecqv-ibi: Identity-based identification with implicit certification", *Journal of Information Security and Applications*, vol. 63, p. 103 027, 2021, ISSN: 2214-2126. DOI: https://doi.org/10.1016/j.jisa.2021.103027.

[151] C.-S. Park, "A secure and efficient ecqv implicit certificate issuance protocol for the internet of things applications", *IEEE Sensors Journal*, vol. 17, no. 7, pp. 2215–2223, 2017. DOI: 10.1109/JSEN.2016.2625821.

[152] Y. Xiao, N. Zhang, W. Lou, and Y. T. Hou, "A survey of distributed consensus protocols for blockchain networks", *IEEE Communications Surveys & Tutorials*, vol. 22, no. 2, pp. 1432–1465, 2020. DOI: 10.1109/COMST.2020.2969706.

[153] J. Liu and Z. Liu, "A survey on security verification of blockchain smart contracts", *IEEE Access*, vol. 7, pp. 77 894–77 904, 2019. DOI: 10.1109/ACCESS.2019.2921624.

[154] D. Dolev and A. Yao, "On the security of public key protocols", *IEEE Transactions on Information Theory*, vol. 29, no. 2, pp. 198–208, 1983. DOI: 10.1109/TIT.1983.1056650.

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

[156] H. Lamrani Alaoui, S. K. Islam, A. El Ghazi, M. Zbakh, A. Touhafi, and A. Braeken, "A highly efficient ecc-based authentication protocol for rfid", *Journal of Se,sprs*, p. 8 876 766, 2021. DOI: https://doi.org/10.1155/2021/8876766.

[157] C. Jacomme, E. Klein, S. Kremer, and M. Racouchot, "A comprehensive, formal and automated analysis of the EDHOC protocol", in *USENIX Security '23 - 32nd USENIX Security Symposium*, Anaheim, CA, United States, Aug. 2023. [Online]. Available: https://hal.inria.fr/hal-03810102.

[158] M. N. M. Bhutta, A. A. Khwaja, A. Nadeem, *et al.*, "A survey on blockchain technology: Evolution, architecture and security", *IEEE Access*, vol. 9, pp. 61 048–61 073, 2021. DOI: 10.1109/ACCESS.2021.3072849.

[159] M.-V. Vladucu, Z. Dong, J. Medina, and R. Rojas-Cessa, "E-voting meets blockchain: A survey", *IEEE Access*, vol. 11, pp. 23 293–23 308, 2023. DOI: 10.1109/ACCESS.2023.3253682.

[160] J. C. Perez Garcia, A. Benslimane, and S. Boutalbi, "Blockchain-based system for e-voting using blind signature protocol", in *2021 IEEE Global Communications Conference (GLOBECOM)*, 2021, pp. 01–06. DOI: 10.1109/GLOBECOM46510.2021.9685189.

[161] Y. Fu and J. Zhu, "Big production enterprise supply chain endogenous risk management based on blockchain", *IEEE Access*, vol. 7, pp. 15 310–15 319, 2019. DOI: 10.1109/ACCESS.2019.2895327.

[162] U. Agarwal, V. Rishiwal, S. Tanwar, *et al.*, "Blockchain technology for secure supply chain management: A comprehensive review", *IEEE Access*, vol. 10, pp. 85 493–85 517, 2022. DOI: 10.1109/ACCESS.2022.3194319.

[163] E. A. Shammar, A. T. Zahary, and A. A. Al-Shargabi, "A survey of iot and blockchain integration: Security perspective", *IEEE Access*, vol. 9, pp. 156 114–156 150, 2021. DOI: 10.1109/ACCESS.2021.3129697.

[164] R. Huo, S. Zeng, Z. Wang, *et al.*, "A comprehensive survey on blockchain in industrial internet of things: Motivations, research progresses, and future challenges", *IEEE Communications Surveys & Tutorials*, vol. 24, no. 1, pp. 88–122, 2022. DOI: 10.1109/COMST.2022.3141490.

[165] M. S. Arbabi, C. Lal, N. R. Veeraragavan, D. Marijan, J. F. Nygård, and R. Vitenberg, "A survey on blockchain for healthcare: Challenges, benefits, and future directions", *IEEE Communications Surveys & Tutorials*, vol. 25, no. 1, pp. 386–424, 2023. DOI: 10.1109/COMST.2022.3224644.

[166] S. Khatri, F. A. Alzahrani, M. T. J. Ansari, A. Agrawal, R. Kumar, and R. A. Khan, "A systematic analysis on blockchain integration with healthcare domain: Scope and challenges", *IEEE Access*, vol. 9, pp. 84 666–84 687, 2021. DOI: 10.1109/ACCESS.2021.3087608.

[167] M. Tahir, M. H. Habaebi, M. Dabbagh, A. Mughees, A. Ahad, and K. I. Ahmed, "A review on application of blockchain in 5g and beyond networks: Taxonomy, field-trials, challenges and opportunities", *IEEE Access*, vol. 8, pp. 115 876–115 904, 2020. DOI: 10.1109/ACCESS.2020.3003020.

[168] Y. Xiao, N. Zhang, W. Lou, and Y. T. Hou, "A survey of distributed consensus protocols for blockchain networks", *IEEE Communications Surveys & Tutorials*, vol. 22, no. 2, pp. 1432–1465, 2020. DOI: 10.1109/COMST.2020.2969706.

[169] M. Castro, B. Liskov, *et al.*, "Practical byzantine fault tolerance", in *OsDI*, vol. 99, 1999, pp. 173–186.

[170] J. Kwon, "Tendermint: Consensus without mining", *Draft v. 0.6, fall*, vol. 1, no. 11, pp. 1–11, 2014.

[171] H. J. Singh and A. S. Hafid, "Prediction of transaction confirmation time in ethereum blockchain using machine learning", in *Blockchain and applications: International congress*, Springer, 2020, pp. 126–133.

[172] V. Xinying Chen and J. Hooker, "A guide to formulating fairness in an optimization model", *Annals of Operations Research*, pp. 1–39, 2023.

[173] Y. Jiao, P. Wang, D. Niyato, and Z. Xiong, "Social welfare maximization auction in edge computing resource allocation for mobile blockchain", in *2018 IEEE international conference on communications (ICC)*, IEEE, 2018, pp. 1–6.

[174] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system", *Decentralized business review*, 2008.

[175] I. Eyal, A. E. Gencer, E. G. Sirer, and R. Van Renesse, "{Bitcoin-ng}: A scalable blockchain protocol", in *13th USENIX symposium on networked systems design and implementation (NSDI 16)*, 2016, pp. 45–59.

[176] S. Aggarwal and N. Kumar, "Hyperledger", in *Advances in computers*, vol. 121, Elsevier, 2021, pp. 323–343.

[177] M. Benji and M. Sindhu, "A study on the corda and ripple blockchain platforms", in *Advances in Big Data and Cloud Computing: Proceedings of ICBDCC18*, Springer, 2019, pp. 179–187.

[178] I. Abraham, S. Devadas, D. Dolev, K. Nayak, and L. Ren, "Synchronous byzantine agreement with expected o (1) rounds, expected communication, and optimal resilience", in *International Conference on Financial Cryptography and Data Security*, Springer, 2019, pp. 320–334.

[179] S. De Angelis, L. Aniello, R. Baldoni, F. Lombardi, A. Margheri, V. Sassone, *et al.*, "Pbft vs proof-of-authority: Applying the cap theorem to permissioned blockchain", in *CEUR workshop proceedings*, CEUR-WS, vol. 2058, 2018.

[180] J. Innerbichler and V. Damjanovic-Behrendt, "Federated byzantine agreement to ensure trustworthiness of digital manufacturing platforms", in *Proceedings of the 1st Workshop on Cryptocurrencies and Blockchains for Distributed Systems*, 2018, pp. 111–116.

[181] D. Schwartz, N. Youngs, A. Britto, *et al.*, "The ripple protocol consensus algorithm", *Ripple Labs Inc White Paper*, vol. 5, no. 8, p. 151, 2014.

[182] D. Mazieres, "The stellar consensus protocol: A federated model for internet-level consensus", *Stellar Development Foundation*, vol. 32, pp. 1–45, 2015.

[183] L. Chen, L. Xu, N. Shah, Z. Gao, Y. Lu, and W. Shi, "On security analysis of proof-of-elapsed-time (poet)", in *Stabilization, Safety, and Security of Distributed Systems: 19th International Symposium, SSS 2017, Boston, MA, USA, November 5–8, 2017, Proceedings 19*, Springer, 2017, pp. 282–297.

[184] F. Yang, W. Zhou, Q. Wu, R. Long, N. N. Xiong, and M. Zhou, "Delegated proof of stake with downgrade: A secure and efficient blockchain consensus algorithm with downgrade mechanism", *IEEE Access*, vol. 7, pp. 118 541–118 555, 2019.

[185] N. Modina, M. Datar, R. El-Azouzi, and F. de Pellegrini, "Multi resource allocation for network slices with multi-level fairness", in *ICC 2022 - IEEE International Conference on Communications*, 2022, pp. 4872–4877. DOI: 10.1109/ICC45855.2022.9838759.

[186] E. Alba and M. Tomassini, "Parallelism and evolutionary algorithms", *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 5, pp. 443–462, 2002. DOI: 10.1109/TEVC.2002.800880.

[187] D. Bertsimas and J. Tsitsiklis, "Simulated annealing", *Statistical science*, vol. 8, no. 1, pp. 10–15, 1993.

[188] M. Dorigo, M. Birattari, and T. Stutzle, "Ant colony optimization", *IEEE computational intelligence magazine*, vol. 1, no. 4, pp. 28–39, 2006.

[189] J. H. Holland, "Genetic algorithms", *Scientific american*, vol. 267, no. 1, pp. 66–73, 1992.

[190] O. Kariv and S. L. Hakimi, "An algorithmic approach to network location problems. i: The p-centers", *SIAM Journal on Applied Mathematics*, vol. 37, no. 3, pp. 513–538, 1979.

[191] W. Mu and D. Tong, "On solving large p-median problems", *Environment and Planning B: Urban Analytics and City Science*, vol. 47, no. 6, pp. 981–996, 2020.

[192] A. Corominas, "On deciding when to stop metaheuristics: Properties, rules and termination conditions", *Operations Research Perspectives*, p. 100 283, 2023.

[193] M. Ahsanullah and V. Nevzorov, "Records via probability theory", in *Records via Probability Theory*, 2015, pp. 1–255.

[194] A. Stepanov, "On the Mathematical Theory of Records", *Communications in Mathematics*, vol. Volume 29 (2021), Issue 1 (Special Issue: Ostrava Mathematical Seminar), no. 1, pp. 151–162, Apr. 2021. DOI: 10.2478/cm-2021-0009. [Online]. Available: https://hal.science/hal-03665013.

[195] D. Bertsimas, V. F. Farias, and N. Trichakis, "The price of fairness", *Operations research*, vol. 59, no. 1, pp. 17–31, 2011.