BLOCKCHAIN BASED DECENTRALIZED IDENTITY SYSTEM FOR INTERNET OF THINGS DEVICES

NESNELERİN İNTERNETİ CİHAZLARI İÇİN BLOKZİNCİR TABANLI MERKEZİ OLMAYAN KİMLİK SİSTEMİ

SAİM BUĞRAHAN ÖZTÜRK

ASSOC. PROF. DR. MURAT AYDOS

Thesis Supervisor

Submitted to

Graduate School of Science and Engineering of Hacettepe University

as a Partial Fulfillment to the Requirements

for the Award of the Degree of Master of Science

in Computer Engineering

2024

ABSTRACT

BLOCKCHAIN BASED DECENTRALIZED IDENTITY SYSTEM FOR INTERNET OF THINGS DEVICES

Saim Buğrahan ÖZTÜRK

Master of Science, Department of Computer Engineering

Supervisor: Assoc. Prof. Dr. Murat AYDOS

January 2024, 93 pages

The quantity of interconnected Internet of Things (IoT) devices has been increasing recently as a result of advancements in communication and hardware technologies. Predictions indicate that, by the end of 2025, there will be more than 30 billion globally interconnected IoT devices due to the broader deployment of 5G and subsequent technologies [1]. This rapid expansion and the complex networks these devices operate within increase the challenges of developing an Identity and Access Management (IAM) system available to all interconnected devices within the network. Such an IAM system must be self-sufficient, universally unique, and compatible across various devices and networks. Blockchain technology, identified by its unique features, such as decentralization, immutability, and cryptographic capabilities, presents a viable solution for the challenges associated with designing an IoT IAM system. Blockchain is a distributed ledger technology that enables a secure, transparent, and immutable way of exchanging data and value without central authority [2]. There are many different blockchain implementations; as of the writing of this thesis, it is estimated that there are over 1000 blockchain implementations worldwide [3].

Many of these implementations offer a feature called chaincode or smart contract that allows the creation of applications that execute in a decentralized manner inside the blockchain network. In this thesis, we have intersected blockchain technology and IoT by proposing an IAM and trust evaluation framework solely based on blockchain technology by leveraging smart contracts within the blockchain network. Several critical functionalities of an IoT IAM system, such as authorization, authentication, auditing, and identity management, were examined. As a result, these functions were redesigned to operate in a decentralized manner within our proposed framework. Throughout the thesis work, existing IoT IAM solutions were identified and compared with the proposed framework in terms of functionality, performance, and cybersecurity-related aspects. In the last part of this study, the proposed framework was fully implemented on the Hyperledger Fabric platform, and it was tested for various predefined use-case scenarios. Besides the functionality, the framework was also tested for the performance aspects, and the results were examined within the study. Additionally, a feature not available in traditional IoT IAM, a trust evaluation mechanism based on the reputation mechanism and trust scores, was designed and implemented within the proposed framework. This mechanism allows devices to validate the trust of each other and make informed decisions on connections in a decentralized manner. In conclusion, our results point out that blockchain technology can be used in designing an IoT IAM system that can operate in a decentralized manner. Although the proposed framework has advantages over the traditional solutions, it may have issues related to scalability and performance, which are inherited from blockchain technology. However, it is essential to note that blockchain technology is still in its early stages and that many researchers worldwide are concentrating on its challenges. Therefore, as blockchain technology matures, its challenges will be resolved, thus opening the door for its broad use in real-world scenarios.

Keywords: Blockchain, Internet of Things, Identity and Access Management, Decentralized Identity, Trust Evaluation, Blockchain Applications

ÖZET

NESNELERİN İNTERNETİ CİHAZLARI İÇİN BLOKZİNCİR TABANLI MERKEZİ OLMAYAN KİMLİK SİSTEMİ

Saim Buğrahan ÖZTÜRK

Yüksek Lisans, Bilgisayar Mühendisliği

Tez Danışmanı: Doç. Dr. Murat AYDOS

Ocak 2024, 93 sayfa

Haberleşme ve donanım teknolojilerindeki gelişmelerin bir sonucu olarak, son yıllarda birbirine bağlı Nesnelerin İnterneti (IoT) cihazlarının sayısı hızla artmaktadır. Tahminler, 5G ve sonrası haberleşme teknolojilerinin de yaygınlaşmasıyla beraber, 2025 yılı sonuna kadar 30 milyardan fazla IoT cihazının birbirine bağlı şekilde konumlandırılacağını göstermektedir [1]. Bu hızlı artış ve IoT cihazlarının içinde bulunduğu karmaşık ağ sistemleri, ağ içerisinde birbirine bağlı IoT cihazlarının ortak kullanabileceği bir Kimlik ve Erişim Yönetimi (IAM) sistemini tasarlamanın zorluklarını arttırmaktadır. İlgili IAM sisteminin kendi kendine çalışabilen, tüm cihazlar için aynı ve çeşitli cihazlar ve ağlar arasında çalışabilir olması gerekmektedir. Bu doğrultuda blokzincir teknolojisi, sahip olduğu merkeziyetsizlik, değiştirilemezlik ve birtakım kriptografik yetenekler gibi benzersiz bazı özellikler sebebiyle IoT IAM sistemlerinin tasarımında bir çözüm olarak karşımıza çıkmaktadır. Blokzincir, merkezi otorite olmadan veri alışverişinin güvenli, şeffaf ve değişmez bir yolunu sağlayan bir dağıtık defter teknolojisidir [2]. Mevcutta birçok blokzincir uygulaması bulunmakla beraber bu tezin yazıldığı an itibariyle dünya çapında

1000’in üzerinde blokzincir uygulamasının bulunduğu tahmin edilmektedir [3]. Bu uygulamaların çoğunda, blokzincir ağı içerisinde merkeziyetsiz uygulamalar geliştirmeye olanak sağlayan akıllı sözleşme adında bir özellik bulunmaktadır. Bu tez kapsamında, blokzincir ağı içerisindeki akıllı sözleşmelerin kullanıldığı, tamamen blokzincir tabanlı bir kimlik, erişim yönetimi ve güven değerleme çerçevesinin tasarımı tarif edilmektedir. Bu doğrultuda, öncelikli olarak mevcut IoT IAM sistemlerinin kritik bileşenleri olan yetkilendirme, kimlik doğrulama ve kimlik yönetimi fonksiyonları analiz edilmiştir. Sonuç olarak bu bileşenler, tez kapsamında önerilen çerçeve bünyesinde merkeziyetsiz biçimde blokzincir üzerinde çalışacak şekilde yeniden tasarlanmıştır. Tez çalışması boyunca, mevcutta kullanılmakta olan IoT IAM çözümleri belirlenerek işlevsellik, performans ve siber güvenlik konuları açısından analiz edilmiş ve önerilen çerçeve ile kıyas edilmiştir. Çalışmanın son aşamasında, önerilen çerçeve Hyperledger Fabric platformu üzerinde gerçeklenmiş ve önceden belirlenmiş bazı senaryolara göre test edilmiştir. İşlevsellik testlerinin yanı sıra, çerçeve performans açısından da test edilmiş ve sonuçlar detaylı bir şekilde incelenmiştir. Sahip olduğu özelliklere ek olarak, önerilen çerçeve bünyesinde geleneksel IoT IAM sistemlerinde bulunmayan bir özellik olan güven değerleme mekanizması bulunmaktadır. Bu mekanizma, cihazların birbirine karşı olan güvenlerini ölçmelerine olanak sağlamakta ve haberleşme esnasında cihazların bilinçli kararlar almasına yardımcı olmaktadır. Sonuç olarak, yapılan çalışma kapsamında elde edilen bulgular, merkeziyetsiz şekilde çalışabilen bir IoT IAM sisteminin tasarımında blokzincir teknolojisinin kullanılabileceğini göstermektedir. Önerilen çerçevenin geleneksel çözümlere göre avantajları olmasına rağmen, ölçeklenebilirlik ve performans gibi konularda blokzincir kaynaklı dezavantajları bulunabilmektedir. Bu doğrultuda, blokzincir teknolojisinin henüz emekleme aşamasında olduğu ve bu teknolojiden kaynaklı zorlukların birçok araştırmacı tarafından araştırıldığı ve çözümlerin üretilmeye çalışıldığı unutulmamalıdır. Blokzincir teknolojisinin olgunlaşmasıyla beraber, bu tarz zorlukların giderileceği ve teknolojinin gerçek dünya uygulamalarında yaygın bir şekilde kullanılmaya başlanacağı değerlendirilmektedir.

Anahtar Kelimeler: Blokzincir, Nesnelerin İnterneti, Kimlik ve Erişim Yönetimi, Dağıtık Kimlik, Güven Değerlemesi, Blokzincir Uygulamaları

ACKNOWLEDGEMENTS

I want to state my special thanks to my supervisor, Assoc. Prof. Dr. Murat AYDOS, for guiding me with his knowledge and patience throughout my graduate study.

I also would like to sincerely thank my managers and colleagues at ASELSAN Inc. for sharing their valuable ideas while writing this thesis. Their support and encouragement were invaluable to me.

I want to thank my family for their moral support whenever I felt stuck throughout my studies.

Finally, I would like to express my gratitude and appreciation to my dear wife, Havva. Her unwavering support, understanding, and patience have been a formidable force behind my academic pursuits.

Saim Buğrahan ÖZTÜRK
January 2024, Ankara

CONTENTS

LIST OF FIGURES

LIST OF TABLES

SYMBOLS AND ABBREVIATIONS

Symbols

Ω	Greek Symbol Omega
β	Greek Symbol Beta
α	Greek Symbol Alpha

Abbreviations

CA	Certificate Authority
CapBAC	Capability Based Access Control
CPU	Central Processing Unit
dApp	Decentralized Application
DB	Database
DID	Decentralized Identity
DLT	Distributed Ledger Technology
DPKI	Decentralized Public Key Infrastructure
GSM	Global System for Mobile Communications
HLF	Hyperledger Fabric
IAM	Identity and Access Management
ICMP	Internet Control Message Protocol
ID	Identity
IDaaS	Identity as a Service
IdM	Identity Management
IIoT	Industrial Internet of Things
IoT	Internet of Things

IP	Internet Protocol
IPFS	Inter Planetary File System
IPv6	Internet Protocol Version 6
IT	Information Technology
ITS	Initial Trust Score
InTS	Interaction Trust Score
JSON	JavaScript Object Notation
KPI	Key Performance Indicator
LoraWAN	Long Range Wide Area Network
MSP	Membership Service Provider
MFA	Multi-Factor Authentication
MQTT	Message Queuing Telemetry Transport
M2M	Machine-to-Machine
NB-IoT	Narrowband Internet of Things
NFC	Near Field Communication
NFT	Non-Fungible Token
Nonce	Number Only Used Once
OAuth	Open Authentication
PKI	Public Key Infrastructure
PoW	Proof of Work
PoS	Proof of Stake
P2P	Peer-to-peer
RAM	Random Access Memory
RFID	Radio Frequency Identification
SaaS	Software as a Service

SAML	Security Assertion Markup Language
SC	Smart Contract
SDK	Software Development Kit
SOA	Service Oriented Architecture
SSI	Self-Sovereign Identity
SSO	Single Sign-On
TCP	Transmission Control Protocol
TPS	Transaction Per Second
UDP	User Datagram Protocol
URI	Uniform Resource Identifier
VC	Verifiable Credential
Wi-Fi	Wireless Fidelity
W3C	The World Wide Web Consortium
6LoWPAN	IPv6 over Low-Power Wireless Personal Area Networks

1. INTRODUCTION

IoT refers to the idea of connecting a broad range of items, such as coffee makers and industrial machinery, to one another. These smart devices can communicate and share data with or without human interaction, thus enabling increased automation and efficiency across various tasks. IoT technology is widely used by businesses and different industrial sectors to provide new services, enhance operations, and sharpen decision making processes. As advantageous as these advancements could be, IoT adoption comes with its difficulties, especially regarding scalability, interoperability, security, and identity management [4]. Such difficulties, mixed with the rapid expansion of IoT devices, render existing server-client-based solutions unsuitable for the future of IoT technology, thereby necessitating the development of new solutions. IoT devices connect with one another through several procedures, such as mutual authentication, device discovery, and authorization. In a typical IoT network, these processes are handled centrally by one organization or server that authenticates and verifies every device ID on the network, ensuring network's permission handling, auditing, and authentication. Although this strategy has worked well in smaller and more straightforward networks, it has considerable drawbacks when applied to more complex and large networks that the IoT technology will eventually evolve into. A graph that shows the forecasts on the number of actively connected IoT devices by region is given in Figure 1.1.

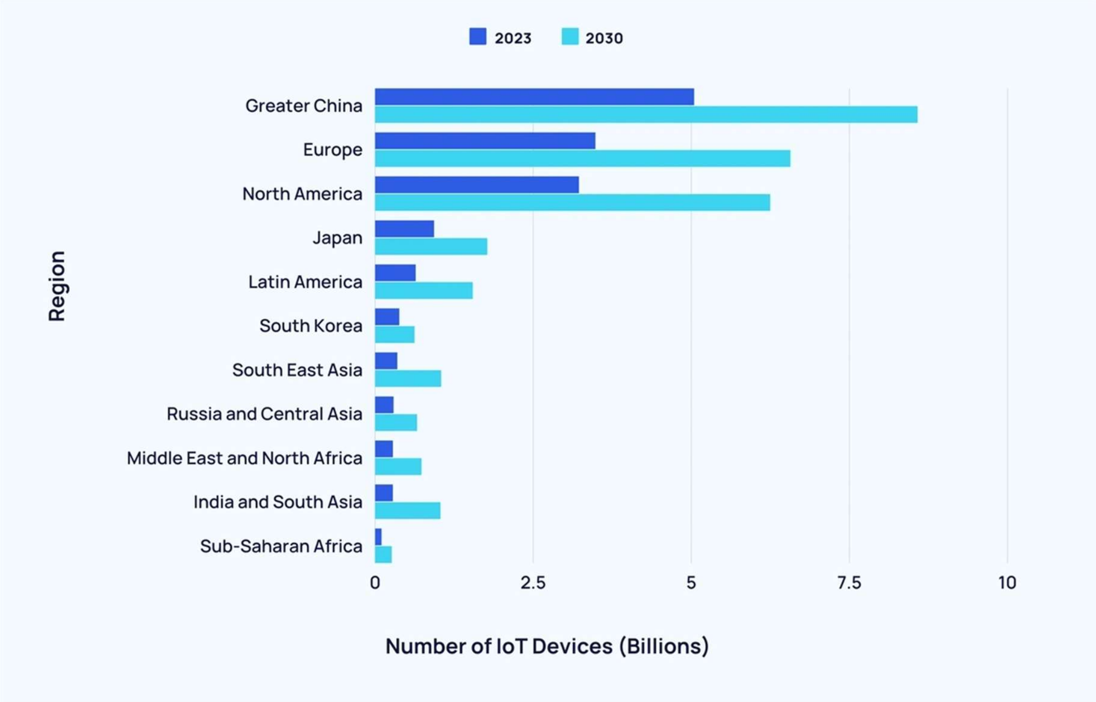


Figure 1.1. Number of Active IoT Devices by Region [5]

Graph by itself shows that the current active IoT network is expected to grow by around %94 in the upcoming seven years. It is known that centralized systems are prone to single points of failure, which is a significant disadvantage when considering IoT networks. The entire IoT network is at risk in the event of a central server failure, possibly resulting in significant data breaches or system failures. Furthermore, the current server-client models find it difficult to manage the growing volume of transactions between devices as the IoT ecosystem grows and additional devices become globally interconnected as a result of the developments in more advanced communication technologies. This leads to decreased performance and slower responses within the IoT network, damaging the technology's future adoption. Additionally, the IoT ecosystem's reliance on central authorities for interaction authorization and identity identification limits its capacity for autonomous interaction and undercuts the possible advantages of the IoT technology's vision. A reliable, secure, scalable, and globally unique IoT IAM system is required to overcome the current limitations of IoT, a system where devices can identify each other and verify the trustworthiness of other devices.

The distributed ledger technology known as blockchain, renowned for its provenance, immutability, decentralization, and cryptographic capabilities, fits nicely with the needs of a trustable, globally unique IAM system designed especially for IoT devices. With its ability to operate decentralized, blockchain can remove the need for a centralized authority in IoT networks for authentication, authorization, and trust assessment by utilizing a p2p network structure. Every transaction is registered on the blockchain structure and is verifiable by each network member separately, encouraging accountability and transparency. Furthermore, the data stored in the blockchain is guaranteed to be immutable and of high integrity, thanks to its cryptographic features [6]. Additionally, blockchain technology offers a tool named smart contracts or chaincodes, which allows the development of applications that can execute in a decentralized manner. Smart contracts in blockchain technology are self-executing digital contracts with the terms of execution written directly into lines of code, typically residing on a blockchain network [7]. Smart contracts can revolutionize business processes by enabling the automation of complex agreements, streamlining operations, and ensuring compliance through code. These digital contracts are typically embedded with the terms and conditions of an operation, written directly into the blockchain, offering a secure, transparent, and efficient way to manage various processes. Smart contracts can significantly reduce operational costs and increase process efficiency by automating tasks and eliminating the need for intermediaries. Transparency and immutability features of the blockchain ensure that once a contract is deployed, its terms and execution are visible to all blockchain network participants, hence fostering trust among all the parties involved. Although the contracts are highly effective in enhancing business efficiency, the effectiveness of a smart contract depends on the accuracy and reliability of the underlying code. They can be used to design and manage various processes, from supply chain management and automated payments to industrial-specific operations.

1.1. Motivation of the Thesis

Identity and Access Management is an essential component of IoT technology that defines the authentication and authorization rules of the network. It is responsible for controlling access to the network and ensuring that only authorized users and devices have access to the system. In general, identity management is used to authenticate users, devices, and services and establish secure communication channels between them. It also enables secure data transfer between different parts of the system and helps to protect the system from malicious attacks. IdM is crucial for ensuring the security and reliability of IoT systems. In many of the traditional IoT networks, identity and access management processes are based on CAs, centralized credential storage databases, and centralized authorities. Such IoT networks may be subject to adverse impacts related to IAM. These impacts are listed below [8]:

- **Lack of Security Updates:** Within extensive IoT networks, many devices relying on different IoT IAM systems can lead to significant delays in responding to cyber threats. This issue is intensified when a security update in one IAM system potentially disrupts others, creating a cascade of security vulnerabilities.
- **Poor Integration:** Many IoT IAM systems struggle with integrating seamlessly with other systems, applications, and networks due to the complexities of the IAM protocol and the vast diversity of devices. This can lead to compatibility issues and hinder efficient operations.
- **Lack of Scalability:** The quantity of interconnected IoT devices is increasing rapidly. This proliferation, mixed with the centralized operation of existing IAM solutions, raises issues related to scalability.
- **Fragmentations Inside the IoT Network:** Inside a global scale IoT network, multiple IAM systems may coexist due to the scale of the network and the variations in protocols and standards used by the devices. The coexistence of multiple IAM systems inside a network may lead to fragmentation, interfering with interoperability and the long-term goal of a connected, international IoT ecosystem.
- **Centralization Issues:** The centralized nature of many IoT IAMs can create bottlenecks and single points of failure inside the IoT network, raising unpredictable performance and security issues.
- **Interoperability Challenges:** Given their complexity, IoT Identity and Access Management (IAM) systems often face challenges in achieving interoperability among the vast range of devices, applications, and networks. This issue is heightened because different manufacturers produce many devices, each potentially employing unique standards or protocols for IAM. Additionally, a lack of mutual trust between these manufacturers can lead to further diversification in these standards, raising interoperability issues. This results in a fragmented IoT landscape where seamless communication and integration between devices and systems are hindered.

In order to solve existing challenges in the IoT IAM and aid the future vision of a globally connected IoT ecosystem, in this thesis, we have proposed a blockchain-based IoT IAM and trust evaluation system that can fully operate in a decentralized fashion. The proposed system is fully implemented on a blockchain platform, and aspects such as cyber security, performance, and functionality are analyzed. Additionally, the potential benefits of the proposed framework are analyzed in the context of IoT technology. The results are compared with the existing IoT IAM solutions to prove that blockchain technology can be a critical solution in addressing the IoT IAM issues related to the rapid proliferation of IoT devices globally.

1.2. Technical Framework and Methodology

Blockchain is a distributed ledger system in which the data is recorded and stored across a network of nodes. It provides a secure and transparent way of exchanging information and assets in a decentralized fashion without a central authority. Briefly said blockchain is an immutable, distributed database that has increased resilience to data tampering and manipulation [9]. Blockchain is made of blocks of data linked to each other through cryptographic algorithms, where each block contains a cryptographic hash of the previous block, a timestamp, and transaction data. The chain of blocks is secured by consensus algorithms and distributed across the whole network. Each block contains validated transactions in an immutable form, providing an immutable record of transactions that cannot be changed or deleted. Blockchain technology is widely used to create distributed applications (dApps) and smart contracts, self-executing applications embedded in the blockchain network.

In this study, the existing IAM solutions were analyzed to identify the critical functionalities of an IAM system. As a result, these functionalities were redesigned to operate in a decentralized manner. Each functionality was designed using sequence diagrams and further analyzed in terms of performance and cyber-security. To implement the proposed framework and functionalities of the IAM, an already established and popular blockchain platform, Hyperledger Fabric, was used. In order to implement and test the proposed framework, various technologies and programming languages were used, including Golang [10], Python [11], NodeJS [12], and Bash. Out of these, Golang was used to implement each IAM functionality as a smart contract inside the blockchain network. Various libraries and design choices were used effectively to achieve maximum performance inside the smart contracts. The blockchain network was designed using Docker containers and installed on a virtual Linux machine using tools such as Kubernetes [13] and Docker Compose [14]. Predefined test scenarios were built to test the framework for functionality. JavaScript language was used to develop test scripts, and the whole testing environment was automated using Bash scripts. Performance tests were held using a tool named Hyperledger Caliper [15]. A primary smart contract was also deployed in the testing environment to serve as a truth anchor on the performance results. The results of the primary smart contract and the proposed framework were compared and analyzed. Lastly, two separate Docker applications were designed to facilitate the function of an IoT device. These applications were limited in hardware resources in terms of CPU and RAM, therefore imitating the function of an actual IoT device. The Docker applications were used to further test the proposed framework from the side of the IoT devices, and the results were analyzed.

Main contributions of this thesis are;

- **Development of a Blockchain-Based IAM Framework for IoT:** This thesis study introduces a novel Identity and Access Management framework utilizing blockchain technology, specifically designed to address the unique needs of IoT ecosystems. This framework not only enhances security and privacy but also introduces decentralization to the forefront of identity management in IoT networks.
- **Integration of Trust Evaluation Protocol:** A significant aspect of this research is the conceptualization and integration of trust evaluation protocols within the blockchain-based IAM framework. This integration is crucial for establishing and maintaining trust among the interconnected IoT devices, ensuring secure, reliable, and autonomous interactions.
- **Implementation Strategy Using Hyperledger Fabric:** The thesis study presents a comprehensive implementation strategy using HLF, a permissioned blockchain platform. This approach demonstrates the practical applicability of the proposed

framework, offering insights into the deployment and actualization of blockchain technology in IoT contexts.

- **Performance and Cybersecurity Analysis:** Detailed performance analysis and cybersecurity considerations form a core part of this thesis study. By analyzing the framework and evaluating its resilience against potential cyber threats, the study provides valuable data on the efficacy and robustness of the proposed solution.
- **Real-World Application Scenarios:** This thesis study extends beyond theoretical development to explore real-world applications. This exploration highlights the versatility and practical significance of the framework in various IoT-enabled scenarios.
- **Proposal of a Decentralized, User-Centric Approach:** A key contribution is the proposal of a decentralized, user-centric approach to IAM in IoT. This approach marks a shift from traditional centralized models, empowering users with greater control over their personal data, devices, and interactions within IoT networks.
- **Foundation for Future Research and Development:** Finally, the thesis study lays the groundwork for future research in this domain. It identifies areas for further development, such as performance optimization and integration with emerging IoT technologies, paving the way for advanced iterations of the framework.

2. BACKGROUND AND RELATED WORK

IoT can be considered a network of devices ranging from basic home appliances and sensors to complex industrial machinery and vehicles that can connect, analyze, and exchange data. It is a technology with the aim of revolutionizing the interaction between physical and computer-based worlds through interconnectedness. This connectivity allows devices to gain a larger view of the physical world around them.

The primary purpose of IoT is to create a more responsive and smarter environment by filling the gap between the physical and digital ecosystems. IoT technology enhances efficiency, convenience, and quality of life. For the smart home case, IoT devices are used to automate tasks such as adjusting lighting and temperature based on smart home user interaction or predefined intervals. IoT devices are also used in industries to optimize manufacturing processes, improve supply chain management, enhance operations, etc. Another vision of IoT is creating smart cities where everything from traffic lights to utility systems are connected to each other to improve city management and living conditions.

The main vision of IoT technology is to create an ecosystem where physical objects of all types can communicate and cooperate with each other to make decisions and perform tasks with minimal human intervention. This vision promotes a scheme where, besides the interconnectedness, the devices are also intelligent and autonomous, capable of providing new insights and services to improve operations. Possible application domains of IoT technology are given in Figure 2.1.

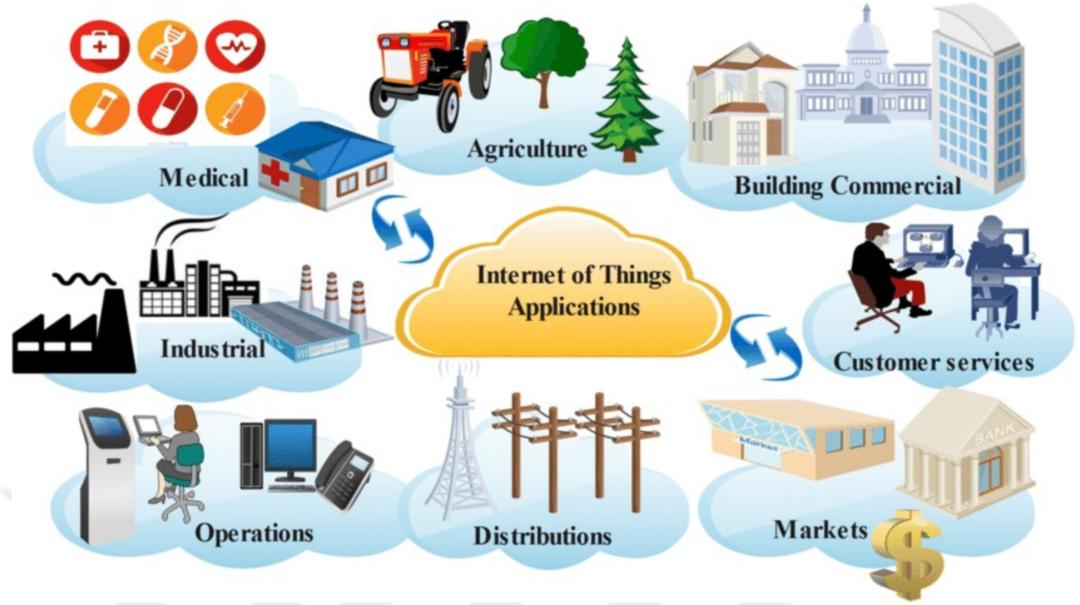


Figure 2.1. Application Domains of IoT Technology [16]

2.1. System Architecture of IoT

The IoT represents a dynamic and evolving technology stack where multiple sub-technologies and frameworks are converged to create interconnected and intelligent systems. Integrating diverse technologies such as sensors, communication protocols, data analytics, cloud computing, and artificial intelligence characterizes IoT. This integration allows the seamless interaction between physical objects and digital platforms and enables devices to collect, exchange, and process data autonomously. There is a vast range of applications and requirements, which results in various architectures for IoT systems. Each of these architectures is customized to meet specific requirements and contexts related to the application [17]. The diversity in IoT architectures can be due to several factors, including varying methods of deployment, industry-specific requirements, security considerations, and technological advancements. For instance, a home automation system requires user-friendliness and low-cost solutions, while industrial applications focus more on robustness, scalability, and real-time data processing. This adaptability feature in IoT architecture promotes IoT technology's versatile and inclusive nature, allowing it to be modified to fit a wide range of use cases across different sectors. The IoT architecture typically consists of several layers, each with its unique role, ranging from physical devices to user interfaces. Although IoT does not have a fixed architectural model, the model should meet specific criteria at its core. These criteria are availability, reliability, mobility, scalability, heterogeneity, interoperability, performance, and absence of security/privacy measures [18].

- **Availability:** Availability in IoT architecture refers to the system's ability to provide continuous and reliable service. It involves ensuring that IoT devices and services are consistently accessible and operational, even in the face of hardware failures, network issues, or high demand. High availability is achieved through redundant systems, resilient network design, and robust disaster recovery plans, ensuring minimal downtime and interruption [19].
- **Reliability:** Reliability ensures that the data is successfully transmitted from one object to another or that data in both the sender and receiver objects are consistent. Reliability aims to ensure the correct and smooth operation of objects/devices connected to the IoT environment and the entire system, reducing delays and errors that may occur in service communication [19]. In the event of a fault in a connected object, another object should be able to rectify this error and prevent potential data loss, thus avoiding delayed decision-making processes or the acquisition of incorrect results. These potential error situations should be anticipated and prevented with the help of methods and algorithms such as the Markov Chain [20] or other methods, ensuring healthy data transmission to the recipients.
- **Mobility:** Mobility in the context of IoT technology refers to the capacity of an IoT system to maintain seamless connectivity and operational functionality as devices move within or between various network environments. This feature is crucial in the increasingly mobile world, where many IoT devices, from personal wearables to vehicle-mounted sensors, are not static but move through different spaces and network zones. Mobility ensures that these devices can continuously communicate and perform their designated tasks without interruption, regardless of their physical location. The mobility challenge in IoT lies in ensuring consistent service quality and connectivity as devices transition across different network types and coverage areas. For instance, a wearable health monitor must continuously send patient data to healthcare providers, whether the patient is at home (connected via Wi-Fi), walking outside (connected via cellular network), or in a vehicle (potentially switching between networks). Effective mobility in IoT thus involves advanced

network technologies and protocols that support dynamic addressing, location awareness, and seamless handover between different network infrastructures. This requires robust and adaptable network protocols, such as mobile IP [21], to ensure devices can maintain a persistent connection as they move.

- **Scalability:** Scalability is a critical feature that refers to the system's ability to handle increasing work efficiently and effectively within the system. In the context of IoT, scalability is about more than just handling a larger number of devices. However, it also refers to the ability to manage increased data volume, processing, and communication requirements that come with the expansion of the network. One of the fundamental challenges in achieving scalability in IoT is the vast diversity and volume of interconnected devices and sensors. Each device generates data in various formats and requires different communication protocols. A scalable IoT system must incorporate these diverse devices seamlessly, allowing new devices to be added or removed without disrupting the overall system functionality [22].
- **Heterogeneity:** The concept of heterogeneity in IoT refers to the capability of the system to support and integrate various types of devices, protocols, data formats, and applications. The diversity inherent in IoT is a result of its wide range of applications across various sectors, which have different standards and requirements. Different types of devices, from sensors to complex machines, can interact and work together in the same ecosystem as a result of heterogeneity, which is central to IoT functionality.
- **Interoperability:** The capability of different IoT systems and devices to communicate and work together effectively is referred to as interoperability within the IoT context. It involves standardizing protocols and data formats to ensure that devices from different manufacturers or systems can exchange and interpret data correctly [23]. Interoperability is an essential feature for creating cohesive and efficient IoT ecosystems. For instance, a device that is based on Wi-Fi should be able to communicate with other IoT devices that are based on NFC or GSM.
- **Security/Privacy Measures:** In the context of IoT, vast amounts of personal and sensitive data are collected and transmitted within the network, hence promoting the importance of security and privacy measures. A secure IoT architecture must include measures like encryption, access control, and regular security updates to protect

against unauthorized access and cyber threats. Privacy measures ensure that personal data is collected, used, and shared ethically and complies with regulations.

- **Performance:** Performance refers to the IoT system's efficiency in processing and responding to data inputs. High-performance IoT systems can handle large volumes of data with minimal latency and ensure timely response in critical applications such as real-time monitoring and control systems. Optimized algorithms, efficient data processing capabilities, and robust network infrastructures are used to increase performance within the IoT network [22].

The architecture of IoT is characterized by a vast range of models, each designed to meet the diverse requirements and complexities of IoT applications. Among the most prominent architecture models are the Three-Layer, SOA-Based, Middleware-Based, and Five-Layer architectures, each offering a unique approach to organizing the functionalities and interactions of IoT systems.

2.1.1. Three-Layered Architecture

There-Layered IoT Architecture is one of the earliest and simplest models, which consists of the Perception Layer, Network Layer, and Application Layer. This model is known for its straightforwardness, making it suitable for simple IoT applications. The model’s architectural view is given in Figure 2.2.

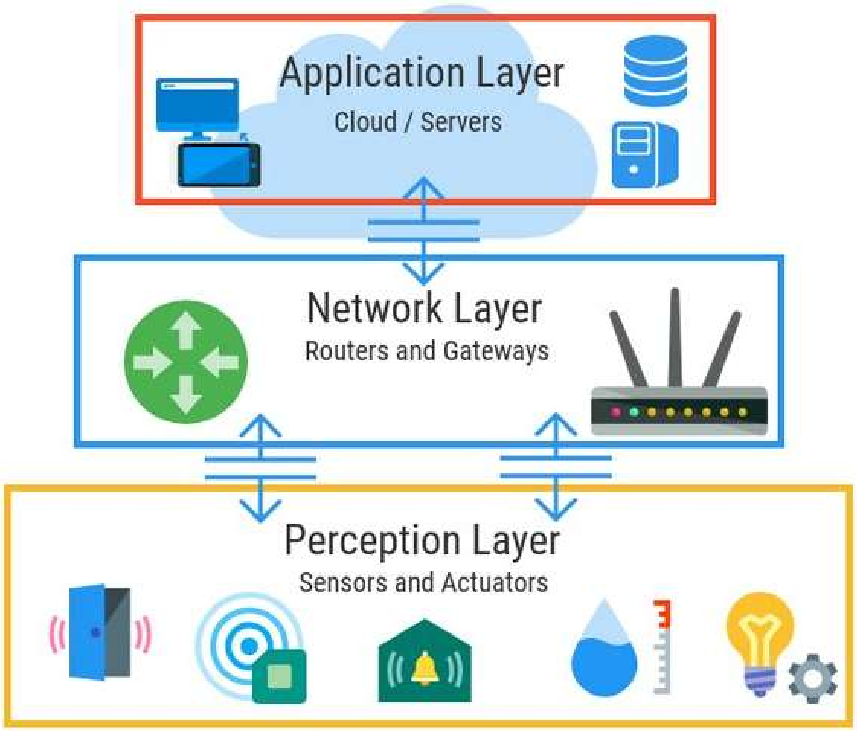


Figure 2.2. Three-Layer Based Architecture of IoT [24]

Three-Layered IoT Architecture consists of three primary layers, each explained below.

- **Perception Layer:** This is the lowest layer of the architecture and is otherwise referred to as the physical layer. It consists of the physical devices and sensors interacting with the physical environment. The primary role of the Perception Layer is to collect data from the environment. This data collection is achieved via various devices, such as sensors that measure physical parameters (temperature, humidity, pressure, etc.), cameras, RFID tags, and other data acquisition devices [25]. These devices gather data from the environment or the objects they are attached to and convert this physical data into digital signals that can be further processed.
- **Network Layer:** This layer is responsible for transmitting the data collected by the Perception Layer to the Application Layer. The Network Layer is critical for ensuring data is transmitted reliably and securely. It uses a range of both wired and wireless communication technologies and protocols, such as Wi-Fi, Bluetooth, LoraWAN [26], ZigBee [27], Z-Wave [28], and cellular networks. The Network Layer handles connectivity challenges, ensuring that IoT devices can communicate

with each other and central servers or cloud platforms. It also plays a crucial role in addressing bandwidth, latency, and network security issues. Protocols such as IPv6, 6LoWPAN [29], UDP, TCP, and ICMP are effectively used within this layer.

- **Application Layer:** The topmost layer in the architecture is the Application Layer. This layer is where the digital data is transformed into services for the end user. The application layer is designed to meet the specific requirements of various IoT applications, such as smart homes, healthcare monitoring, environmental monitoring, and industrial automation. It includes various algorithms and software necessary for data analysis, decision-making processes, and user interface design. The Application Layer is considered the phase where the IoT system interacts with the users, providing them with information, control capabilities, and automation based on the data collected by previous layers. Protocols and frameworks such as MQTT [30], CoAP [31], HTTP/HTTPS, and WebSockets [32] are used within this layer to provide IoT devices with the necessary data exchange and processing capabilities required for the Three-Layered IoT Architecture.

2.1.2. Service-Oriented Architecture for IoT

The SOA-Based IoT is a design framework that applies the principles of service orientation to the IoT. This approach is motivated towards creating a flexible, modular, and scalable IoT system where functionalities are packaged as interoperable services. In the SOA-Based IoT framework, the system is divided into distinct layers, with each layer focusing on different aspects of service delivery [33]. An overview of the typical layers in an SOA-based IoT framework is given below.

- **Object Layer:** This is the foundational layer of the architecture, comprising the physical objects in the IoT ecosystem. These objects include sensors, actuators, RFID tags, and other IoT devices. The main function of this layer is to interact with the physical environment, collect data, and send these data to the next layer. Within the layer, actuators and sensors work in a coordinated manner and produce big data for the IoT system.
- **Object Abstraction Layer:** This layer serves as an intermediary between the physical objects and the higher layers of the architecture by abstracting the details

of the hardware devices, providing a uniform interface for accessing data and functionalities. Object Abstraction Layer ensures that the data from various devices is normalized and made ready for further processing, promoting interoperability between different types of IoT objects.

- **Service Management Layer:** Service Management Layer is responsible for managing the services that are offered by the IoT system. It involves registration, discovery, and management of IoT devices. Furthermore, this layer catalogs the services, handles service requests and responses, and ensures that the services run effectively. Shortly, this layer acts as a directory and controller for available services and, therefore, plays a crucial role in maintaining the service-oriented structure of the whole architecture.
- **Service Composition Layer:** Individual services are combined to create composite services or complete workflows within this layer. It is where the business logic is implemented. Service Composition Layer combines different services to work together, creating more complex and value-added functionalities. This orchestration can be based on predefined rules, user inputs, or dynamic decision-making algorithms. This layer, adds versatility, and functionality to the IoT system, enabling it to perform complex tasks and processes that are more than the sum of individual services.
- **Application Layer:** The Application Layer is the user-facing and topmost layer of the architecture. It presents services and functionalities of the IoT system to the end-users in an accessible and usable manner. This layer includes user interfaces, dashboards, and application software that interact with the underlying services to deliver the final value to the users. It translates the complex processes and data from the IoT system into human-understandable and actionable information, therefore enabling users to make informed decisions and interact effectively with the IoT environment.

2.1.3. Middleware Based Architecture

Middleware-based architecture acts as a critical intermediary layer between the hardware and application layers, addressing the challenges of heterogeneity, scalability, and interoperability in IoT systems. The architecture is designed to manage the complexity and interactions of diverse IoT devices. An overview of the layers is given below.

- **Physical Layer:** The Physical Layer forms the base of this architecture. Physical components of an IoT system, such as sensors, actuators, and other IoT devices, are part of this layer.
- **Backbone Network Layer:** The Backbone Network Layer, sometimes referred to as the Network Layer, is responsible for transmitting data from the Physical Layer to other parts of the IoT system.
- **Coordination Layer:** This layer is an intermediary facilitating communication and data flow between the Physical and Middleware Layers. It is responsible for tasks such as network coordination, device management, and initial data processing. The Coordination Layer often handles the more immediate and local decision-making processes based on the data received from the physical devices.
- **Middleware Layer:** The Middleware Layer is at the core of this architecture and acts as a bridge between the hardware and application layers. It provides essential services such as data aggregation and processing, protocol translation, security, and device management. It ensures that the data from various devices is standardized and made available to applications in a coherent format. The Middleware Layer abstracts the complexities of the underlying hardware and network, providing a unified and simplified interface for the Application Layer [34].
- **Application Layer:** This layer is where the processed data is utilized to deliver specific IoT services and applications to end-users. It includes various applications and user interfaces that leverage the data processed by the Middleware Layer to provide practical and user-centric solutions.

2.1.4. Five Layered Architecture

The Five-Layered Architecture for IoT is a more advanced and comprehensive framework compared to simpler models like the Three-Layer Architecture. It provides a more detailed structure by addressing the complexities and scale of modern IoT systems. The architectural view is given in Figure 2.3.

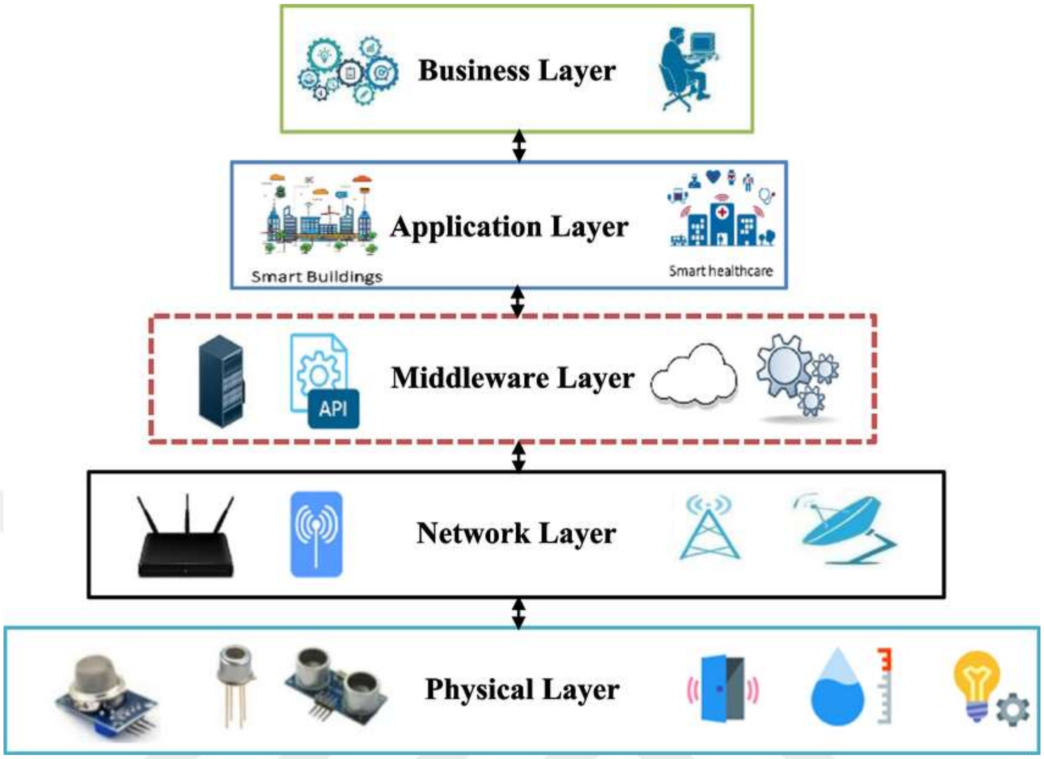


Figure 2.3. Five Layered Architecture for IoT [35]

Detailed information about each layer is given below.

- **Physical Layer:** Also known as Perception Layer, this foundational layer consists of physical devices like sensors, actuaries, RFID tags, and other data collection devices. The goal of this layer is to gather data from the environment. This includes various physical parameters such as temperature, humidity, motion, and light. The devices in this layer are responsible for converting the physical data into digital signals that can be transmitted and processed [36].
- **Network Layer:** Also known as the Transport Layer, it handles the transmission of the data collected by the Physical Layer to the Middleware Layer and vice versa. The Network Layer makes use of various communication technologies, both wired and wireless, to manage the communication between layers. This layer focuses on ensuring efficient, reliable, and secure data transmission across the network.
- **Middleware Layer:** Middleware Layer is where the data transmitted by the Network Layer is stored, managed, and processed. This layer may include cloud computing platforms, data centers, and edge computing nodes, which provide the

computational power and storage capacity needed for large-scale data processing. It handles tasks like data analytics, database management, and information processing.

- **Application Layer:** Specific IoT applications are developed and implemented on the Application Layer to provide solutions to the IoT end-users. This layer translates the processed data into applications, such as smart home automation, health monitoring systems, smart agriculture, and industrial automation. It is modified to the user's needs, focusing on delivering a user-friendly interface and relevant functionalities based on the processed data.
- **Business Layer:** The topmost layer is the Business Layer, which manages the entire IoT system, including the applications. This layer is responsible for the overall business strategy, planning, and management of the IoT system. This includes aspects like business models, user privacy, service management, and ensuring the alignment of IoT operations with business objectives. The Business Layer's role is crucial in decision-making processes, policy formulation, and ensuring the economic viability of the IoT system.

The Five-Layered Architecture provides a detailed approach to designing IoT systems. It ensures that every aspect, from data collection to business implications, is addressed, making it suitable for complex and large-scale IoT deployments. This architecture facilitates scalability, interoperability, and the efficient integration of diverse IoT components and services.

2.2. Identity of Things Management for IoT

Identity Management in IoT is considered a critical component that involves defining, managing, and securing the digital identities of various interconnected devices within an IoT ecosystem. IoT networks comprise various devices, from simple sensors to complex machinery, each with unique roles and capabilities. Managing these identities is crucial for the security, efficiency, and reliability of the IoT system. The primary goal of identity management in IoT, otherwise identity management of things, is to ensure that each device has a unique and verifiable identity. This is crucial for several reasons:

- **Security:** IoT devices may collect, process, and exchange sensitive data. Ensuring that each is authenticated and authorized is vital to prevent unauthorized access and data breaches. IdM helps implement authentication and authorization mechanisms to protect the IoT network against cyber threats.
- **Interoperability and Communication:** IoT devices communicate with each other and with central servers or cloud-based services. IdM ensures that these communications are between trusted parties by verifying the identities of the entities involved.
- **Management and Configuration:** There may be thousands or millions of devices in an IoT network; efficiently managing these identities is paramount. IdM allows for correctly classifying, grouping, and managing devices, facilitating more straightforward configuration, monitoring, and maintenance.
- **Data Integrity:** Ensuring that the data collected and transmitted by IoT devices is accurate and reliable is considered a critical aspect of an IoT system. IdM contributes to data integrity by ensuring that data is coming from verified and correctly functioning devices.

Design of an IdM system in IoT involves a series of steps to ensure that each device within the network is uniquely identified, authenticated, and authorized for specific roles and actions. The IdM process starts by assigning a unique identifier to each IoT device, which could be a digital certificate, a serial number, or any other distinctive form of identification. This uniqueness is essential for differentiating each device in a vast IoT network. After the identification, the next step is registration, where devices are added to the IoT system. During registration, devices must go through authentication processes to verify their identity, which can be based on passwords or digital keys, depending on the system's security requirements. Once a device is authenticated, it must also be authorized to perform certain actions or access specific data within the IoT network. This process is managed through various methods, such as access control policies. Access control policies allow devices to be assigned specific roles based on their unique identity. By using these roles, devices can perform actions within the network. Access policies are crucial in controlling the scope of activities each device can undertake, enhancing the security and integrity of the IoT network.

Furthermore, identity management in IoT also manages the entire lifecycle of IoT devices, including deployment, maintenance, updates, and decommissioning. An IdM system ensures that these phases are handled securely. The lifecycle approach is critical in maintaining the accuracy and relevance of the identity information throughout the device's use.

2.2.1. Identity Related Concepts

Identity refers to a set of characteristics or attributes of an entity, and it represents this entity within an application domain. It is essentially the digital representation of an entity. For a person, an identity could include attributes like name, date of birth, and biometric information, whereas for an IoT device, it could involve specific data such as device type, manufacturer, and operational capabilities. In order to correctly understand the identity and operation of an IdM system, a few related concepts and their relationship must be observed. These concepts are list below.

- **Entity:** An entity refers to any distinct individual or identifiable unit, which could be a person, a device, or even a software component. In short, an entity is anything that can have an identity within a system.
- **Attributes:** Attributes are specific pieces of information that are associated with the identity of an entity. They typically describe the characteristics or properties of the entity. Attributes can be various types of data, such as name, age, job title for a person or model, software version, and location of an IoT device. Attributes play a key role in access control, where decisions on what an entity is allowed to do within a system are based on its attributes [37].
- **Credentials:** Credentials are information that can be used to prove an identity. An entity presents credentials to authenticate itself within an ecosystem. For human users, credentials often include things like passwords, PINs, or biometric data (like a fingerprint or iris scan). For devices, credentials can be digital certificates or cryptographic keys. Credentials are essential for security, as they verify that an entity is who it claims to be.
- **Identifier:** An identifier is a unique value that is used to distinguish one entity from another within a system. It can be considered as a label assigned to an entity's

identity. For an IoT device, it could be a serial number or a MAC address. Identifiers are used in the process of identifying entities within a system [37].

- **Identity Provider:** An IdP is an entity that creates, maintains, and manages identity information for principals (which can be users, services, or IoT devices) and provides authentication services to other service providers within a network or federation. The primary function of an IdP is to authenticate entities and provide information that contains the authenticated identity and, possibly, other attributes related to the entity. Service providers then use this information to grant access to their services. IdPs are used in scenarios where single sign-on (SSO) is implemented. They allow users or devices to log in once and access multiple applications or services without the need for repeated logins. Examples of IdPs include Lightweight Directory Access Protocol (LDAP) servers, Microsoft Active Directory, and cloud-based solutions like Google Identity Platform or Okta.
- **Service Provider:** A Service Provider is an entity that offers and hosts services. SPs rely on IdPs to authenticate the identity of entities before granting access to their services. When an entity attempts to access a service from an SP, the entity is redirected to an IdP for authentication. Once authenticated, the IdP sends information (typically a token) back to the SP, confirming the entity's identity and access rights. The SP uses this information to determine the level of access or the type of services that the authenticated entity is authorized to use. This way, the SP offloads the responsibility of managing user identities and authentication to the IDP.

The relationship between identity related concepts are given in Figure 2.4.

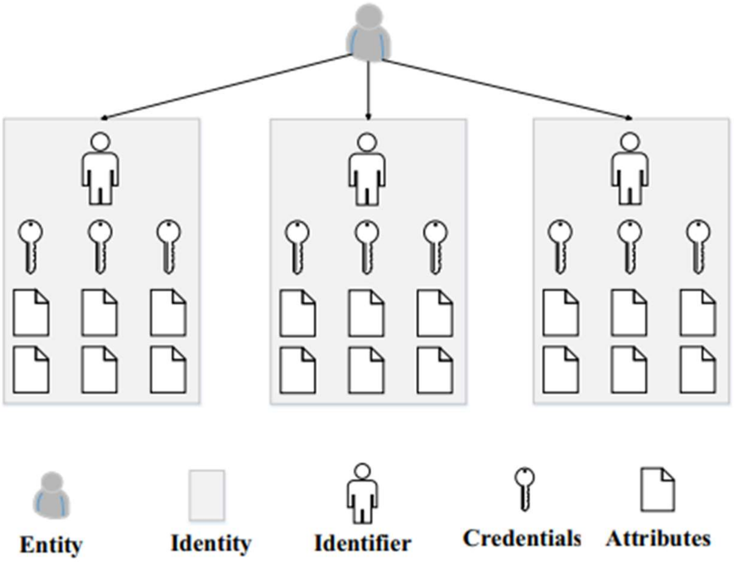


Figure 2.4. Relationship of Identity Related Concepts [38]

2.2.2. Identity Management System Architectures

IdM architectures are fundamental to the security and efficiency of both organizational IT environments and internet ecosystems. These architectures define the framework for managing identities, credentials, access rights, and the policies that govern the secure and controlled use of resources and services. IdM architecture forms the initial framework for managing digital identities within an organization or network. IdM specifically manages the creation, maintenance, and administration of identity information. It involves defining and assigning a digital identity to each entity within the system. These digital identities comprise various attributes like names, roles, personal details, and device specifications. Different IdM architectures, each with unique characteristics, have evolved to suit specific requirements.

2.2.2.1. Centralized IdM Architecture

Centralized IdM Architecture presents a model where the management of digital identities, consisting of their creation, maintenance, and deletion, is handled in a single, centralized system. This approach centralizes the control and administration of all identity-related processes and data within a network or organization, hence offering a single view and management system for all identities. In a centralized IdM architecture, a central server or set of servers is responsible for storing and managing all identity-related information. This includes the user's personal details, credentials, roles, access rights, and any other identity attributes [39]. The centralized nature of this model simplifies the management of identities by providing a single source of truth. It allows for central enforcement of identity policies and access controls across the entire network. Figure 2.5. shows the architecture of a centralized IdM system.

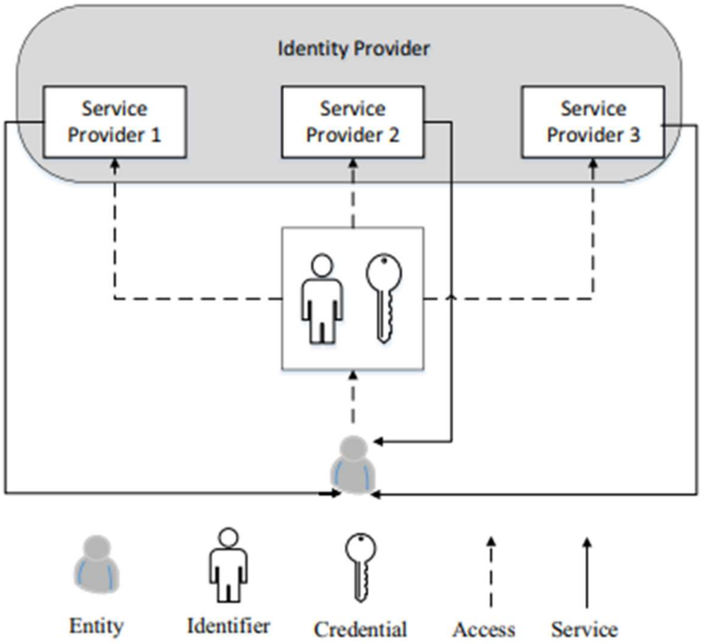


Figure 2.5. Centralized IdM Architecture

One of the main advantages of centralized IdM architecture is its simplicity and ease of management. Having a single, centralized system for identity management reduces the complexity involved in managing multiple identity stores and ensures the application of policies in a uniform manner. However, this model does come with challenges. Centralized systems can create bottlenecks and single points of failure. If the central identity management system goes down, access to a wide range of services and applications can be impacted. Additionally, as the system grows, the central system may face scalability-related issues due to the need to handle the increasing number of identities and transactions. There are also concerns regarding privacy and security, as centralizing sensitive identity data can make it an appealing target for cyber threats.

2.2.2.2. Federated IdM Architecture

Federated IdM involves multiple distinct organizations or systems agreeing to share identity information and trust each other's identity-related processes. This model is valuable in scenarios where users or devices may need to access resources across domains like cloud computing (SaaS applications), e-commerce, and academic consortia. In a federated IdM system, when a user from one domain tries to access a service in another domain, the SP trusts the authentication decision made by the user's home domain. This is handled by agreed-upon standards and protocols such as SAML [40], OAuth [41], and OpenID Connect [42], which enable secure and seamless sharing of identity-related data across different domains. The federated model offers advantages, especially regarding user convenience and efficiency. It eliminates the need for multiple accounts and passwords for different services, reducing the complexity of managing numerous credentials, which often leads to improved security practices. This model also enables the process of accessing multiple services across different platforms, enhancing the overall user experience. Federated IdM supports SSO functionality, where users log in once and gain access to various applications and resources across different domains without the need for repeated authentications [43]. This not only improves user convenience but also reduces the workload on IT departments in managing multiple accounts and helps maintain a consistent security architecture across various environments. Typical architecture of a federated IdM system is given in Figure 2.6.

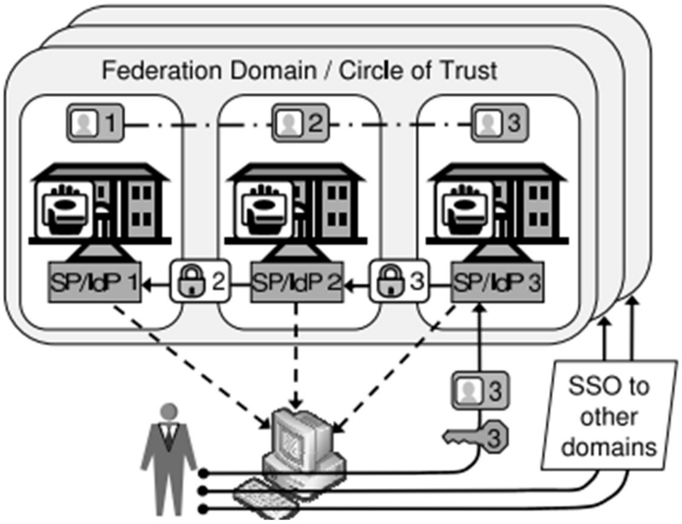


Figure 2.6. Federated IdM Architecture [44]

The success of a federated IdM architecture heavily relies on the establishment of trust relationships between participating entities. Forming these relationships requires consensus between parties, alignment on security policies, and compliance with shared standards. Furthermore, ensuring privacy in a federated system can be challenging as user information is shared across different domains. While the federated IdM reduces the number of user credentials, it also creates potential vulnerabilities, as compromising a single account could grant access to multiple services.

2.2.2.3. Self -Sovereign Identity Architecture

SSI is an emerging IdM concept where individuals have sole ownership and control over their digital identities. It is a user-centric approach to digital identity that emphasizes individual control and ownership over personal identity data. It leverages decentralized technologies like blockchain to create a secure, portable, and interoperable framework for identity management. The architecture of SSI is designed to empower users, ensure privacy, and facilitate trust without relying on centralized authorities. SSI provides enhanced privacy controls and robust security infrastructure through various cryptographic protocols, allowing users to share their identity data without exposing it unnecessarily. Users can carry their digital identities across different platforms and services without depending on any single provider [45]. Users have complete control and sovereignty over their identity data. The concept of SSI is primarily designed with human users in mind, focusing on individual control and management of personal identity data. While it offers significant user privacy and security benefits, its application to the IoT domain results in various challenges. For instance, IoT systems often rely on automated decision-making processes such as device authentication, authorization, and auditing, which are absent due to a lack of related functionalities in SSI frameworks.

2.2.2.3. Cloud-Based IdM Architecture

Identity as a Service (IDaaS) is a cloud-based IdM solution that provides identity management-related services over the cloud. It offers scalability, flexibility, and reduced infrastructure costs, making it a viable choice for such organizations that already use cloud-based applications. In cloud-based IdM solutions, the management of identities and related processes is handled remotely on cloud servers rather than on-premise systems. The primary advantage of cloud-based IdM solutions is their scalability. As organizations grow or experience increasing demand, the cloud infrastructure can scale up or down to meet these changing needs. This elasticity is particularly beneficial for businesses that experience spikes in user activity. Additionally, the cloud-based approach offers cost savings, as it reduces the need for in-house infrastructure and dedicated IT staff to manage identity systems [46]. Security is an important aspect of cloud-based IdMs. Providers typically offer robust security measures, including MFA [47], encryption, and regular security updates to protect against cyber threats. However, organizations must also consider the security implications of entrusting sensitive identity data to a third-party provider and ensure they comply with relevant data protection regulations. Some of the examples for cloud-based IdM systems include Okta [48], Microsoft Azure Active Directory [49], and Google Cloud Identity [50].

2.2.3. Identity and Access Management Systems

IAM systems are considered complex frameworks in that instead of managing digital identities; they also govern how these identities are used to access resources and services within an organization or network. IAM systems contain a more comprehensive range of functionalities than IdM systems, which primarily focus on creating, maintaining, and deleting digital identities. Typically, IAM builds upon the core identity data managed by IdM to include access control and privilege management. The main functionalities supported by IAM systems are listed below.

- **Authentication:** Authentication in IAM is considered the initial step in the security process, verifying the identity of users or entities that are trying to access a system [51]. It ensures that access to resources is granted only to legitimate and verified users. The authentication process can use various methods such as passwords, biometric verification, security tokens, and MFA, where entities must provide multiple verification factors to prove their identity. A specific form of authentication is Mutual Authentication, where both the entity and the system authenticate each other, therefore enhancing security by preventing cyber threats such as man-in-the-middle.

- A**uthorization and Access Control:** Authorization plays a crucial role in determining the level of access and specific actions that authenticated users are allowed within the system [52]. This is typically governed by policies such as RBAC [53], where access rights are assigned based on the entity's role within the organization. Another method is referred to as Attribute-Based Access Control (ABAC) [54], which uses various attributes (such as location or time of access) for more dynamic access control. These methods ensure that entities only have the required permissions to fulfill their job roles, hence protecting sensitive data and critical system functionalities from unauthorized access.
- **Auditing:** Auditing functionalities in IAM are for maintaining transparency and accountability. They involve monitoring and recording user activities within the system, creating an audit trail that can be reviewed for any suspicious activities or security breaches. This aspect of IAM is crucial for ensuring regulatory compliance, as many organizations are subject to various data protection and privacy regulations [55]. IAM systems help in achieving compliance with these standards by controlling and monitoring access to sensitive data and providing logging and reporting tools.
- I**dentity Lifecycle Management:** Identity Lifecycle Management is a functionality that handles the management of digital identities from their initial creation to eventual retirement. This process begins with the creation and provisioning of an identity, typically when a new user joins an organization, or a new device is integrated into the network [56]. The identity is configured at this stage with appropriate access rights, roles, and credentials, such as user accounts, role assignments, and permission grants. As the user's role or needs evolve, Identity Lifecycle Management ensures that their identity is updated and maintained accordingly by adjusting roles and access permissions to align with their current position and responsibilities.

IdM can be seen as a subset of IAM. While IdM focuses primarily on the accurate and secure management of digital identities, IAM takes these identities and manages how they interact with and access various resources and services. IAM solutions rely on the foundational identity data provided by IdM systems but extend their functionalities to include access control, policy enforcement, and security auditing. In summary, IAM systems provide an essential framework for managing identities and controlling how these identities are utilized within an organization's IT environment. They ensure that the right people (or entities) have access to the right resources at the correct times, all while maintaining compliance with security policies and regulations. IAM and IdM relationship is considered complementary, with IAM building upon the core functionalities of IdM to create a more secure, efficient, and compliant IT infrastructure.

2.3. Blockchain Technology

Blockchain can be identified as the driving technology behind popular cryptocurrencies such as Bitcoin [57] and Ethereum [58]. It is a specific type of DLT that gained popularity among researchers in recent years. In its most simplistic form, a blockchain is a chain of blocks, each including several transactions that have occurred inside the blockchain network. Blockchains operate on a decentralized network of nodes, each maintaining a full copy of the chain itself. By having the chain on every node, blockchain ensures transparency, immutability, integrity, and security of the data it contains. Blocks are continuously appended to the blockchain through various methods, with one of the most popular being the mining process. Through mining, network nodes check the validity of the received transactions and generate a new block based on these transactions. After the validation is complete, consensus mechanisms come into play, and as a result, a new block is created and appended to the blockchain. The key benefits of blockchain technology are listed below.

- **Decentralization:** Blockchain, due to its nature, distributes its ledger across a network of nodes, making it decentralized. This structure reduces the risks associated with central points of failure and control, such as data breaches or system outages. Decentralization also means that no single entity is authorized to modify the ledger, which enhances the system's fairness and democratizes data management.
- **Enhanced Security and Immutability:** Blockchain makes use of advanced cryptographic techniques that secure data transactions. Each blockchain within the blockchain is linked to the previous one via cryptographic hashes, creating a chain that is difficult to alter. This makes the data stored on a blockchain tamper-resistant. Immutability is a primary aspect of blockchain. Once data is recorded, it cannot be

changed without altering all subsequent blocks and gaining consensus from the network, which is computationally unfeasible.

- **Transparency and Traceability:** Blockchain's ledger is accessible to all participants with permission, making it highly transparent. Every transaction on the blockchain is recorded and can be traced back to its origin. This traceability is especially beneficial in supply chain management, as it enables tracking goods from production to delivery, ensuring authenticity and compliance.
- **Improved Privacy and User Control:** While blockchain is transparent, it also offers mechanisms to maintain privacy. For instance, users can control what information they wish to share on blockchain networks. Blockchain enables user-controlled privacy in a way that traditional systems, which are often controlled by a single entity, cannot.
- **Facilitation of Trust:** Blockchain maintains trust in environments where trust is not implicit, like unknown parties in a transaction. Its transparent, immutable, and consensus-driven nature assures parties that the data is accurate and unaltered. Blockchain eliminates the need for trust in central authority, which is particularly beneficial in international transactions where legal and financial regulations may vary.

2.3.1. Chain Structure of Blockchain

A blockchain is essentially a chain of blocks containing a list of transactions. Every block has a unique cryptographic hash of its contents and includes the previous block's hash in the chain. This linkage ensures that it becomes exceedingly difficult to alter once a block is added to the blockchain. Chain structure of blockchain technology is given in Figure 2.7.

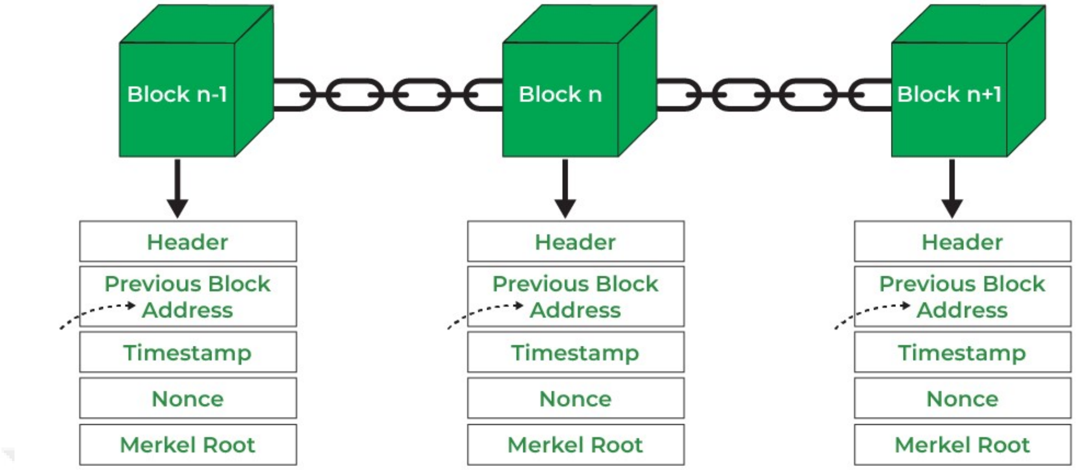


Figure 2.7. Chain Structure of Blockchain [59]

A block on a typical blockchain implementation consists of the following fields: Header, Previous Block Address, Timestamp, Nonce, and Merkel Root. Explanation for each field is given below.

- **Header:** The block header contains metadata about the block. It is used to identify the block within the blockchain network.
- P**revious Block Address:** This field contains the hash of the previous block in the blockchain. A hash is a cryptographic string generated from the data within a block. By including the previous block's hash, each block is cryptographically linked to its predecessor, forming a continuous chain.
- **Timestamp:** The timestamp records the time when the block was created. It maintains the blockchain's chronology, ensuring all network participants can agree on when each block was added.
- **Nonce:** The nonce is a value that is used in PoW-based blockchains. It's a value that miners repeatedly change to alter the block header's hash, aiming to find a hash that meets the network's difficulty target.
- Merkle **Root:** The Merkle root is a single hash representing all the transactions in the block. It is derived from the hashes of all individual transactions in the block, arranged in a Merkle tree [60]. The Merkle root enables efficient and secure verification of transaction contents within a block. It ensures that none of the

transactions have been tampered with or altered, as changing even a single transaction would result in a different Merkle root.

2.3.2. Consensus

Consensus mechanisms can be considered the backbone of blockchain technology, serving as the means to achieve agreement among distributed nodes about the ledger's state. In a decentralized environment with no central authority, consensus mechanisms ensure that every transaction is recorded and the integrity of the blockchain is maintained. These mechanisms are also used to ensure security and trust among participants. In short, a consensus mechanism is a set of rules and processes that determine how transactions are verified and added to the blockchain. The mechanism enables all nodes in the network to reach a joint agreement on the current state of the distributed ledger. This agreement prevents double spending and ensures that each copy of the ledger is identical across the blockchain network. Consensus also plays a role in maintaining the network's security by preventing fraudulent transactions and ensuring that no single entity can control or alter the ledger maliciously. Some of the popular consensus mechanisms are listed below.

- **Proof of Work (PoW):** PoW is used by various blockchain implementations such as Bitcoin, Ethereum, and Litecoin [61]. In PoW, miners compete with each other to solve complex cryptographic puzzles [62]. The first to solve the puzzle gets the right to add a new block to the blockchain and is rewarded with the blockchain's native cryptocurrency. PoW provides robust security. However, it is criticized for its high energy consumption and potential centralization through mining pools.
- **Proof of Stake (PoS):** PoS is currently implemented on various blockchain platforms, such as Cardano [63] and Polkadot [64]. PoS selects validators based on the number of coins they hold and are willing to stake or lock up as collateral. The more coins staked, the higher the chance of being chosen to validate new transactions and add new blocks. PoS is more energy-efficient than PoW and reduces the risk of centralization but raises concerns about the rich getting richer since higher stakes increase the chances of being chosen as a validator [65].
- **Raft Consensus:** Raft organizes time into terms, and each term starts with an election to appoint one node as the leader. This leader handles all client interactions

and manages log entries across the system. Suppose a node doesn't receive communication from the leader within a specified period. In that case, it assumes no active leader and initiates a new election, ensuring continuous operation even in node failures [66]. The elected leader processes client requests, replicates these commands to follower nodes, and ensures that all nodes are consistent in their ledger state.

2.3.3. Smart Contracts

Specific blockchain platforms support the execution of independent logic units, called smart contracts or chaincodes. Smart contracts can be described as decentralized autonomous agents that reside on top of a blockchain network whose behavior is defined by the code they contain. They enable the implementation of business logic in a decentralized manner, providing a way to automate processes within the blockchain network. Like users, smart contracts have unique addresses that differentiate them from other participants. Users can interact with smart contracts by sending specific transactions to their addresses, triggering predefined functions within the contract. This capability adds programmability and automation features to the blockchain network, expanding its application beyond simple transactions. Typical workflow of smart contract is summarized in Figure 2.8.

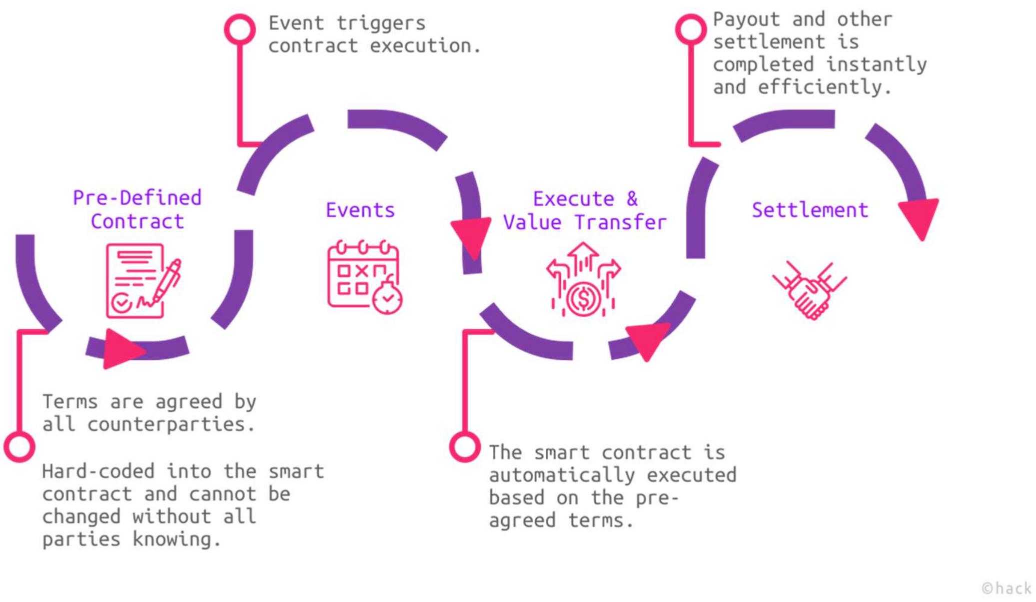

Event triggers contract execution.
Payout and other settlement is completed instantly and efficiently.
Pre-Defined Contract
Events
Execute & Value Transfer
Settlement
Terms are agreed by all counterparties.
Hard-coded into the smart contract and cannot be changed without all parties knowing.
The smart contract is automatically executed based on the pre-agreed terms.
©hack

Figure 2.8. Workflow of Smart Contract [67]

Smart contracts are applied in various sectors, such as finance, supply chain management, and real estate. They enable the creation of dApps that run on blockchain platforms. In short, smart contracts offer a way of creating and executing agreements in a digital and decentralized environment. Their potential extends beyond simple transactions, enabling complex applications that can operate autonomously, securely, and transparently.

2.3.4. Private and Public Blockchains

Private and Public blockchains are two primary types of blockchain architectures, each serving different needs and offering distinct features in terms of accessibility, control, and participation. Public blockchains are entirely open and accessible to anyone. Anyone can join the network, participate in the process of block verification, and view all transactions on the blockchain. Examples include Bitcoin and Ethereum. Ideal scenarios requiring transparency and where participants' trust is limited, such as cryptocurrencies and certain types of dApps, are suitable for public blockchains.

Private blockchains, also known as permissioned blockchains, restrict access to a specific group of users. Participation in the network is limited to authorized members only. Examples include Hyperledger Fabric (HLF) and R3 Corda [68]. Private blockchains offer greater privacy as transactions are visible only to network members. They can also be more efficient in terms of transaction processing speed, as they handle fewer transactions and can optimize protocols for the specific needs of their members. Private blockchains are well-suited for business applications, especially where privacy and data confidentiality are preferred. This includes supply chain management, enterprise resource planning, and inter-organizational record keeping. Main differences between public and private blockchains are given in Table 2.1.

Table 2.1 Differences Between Public and Private Blockchain

Feature	**Public Blockchain**	**Private Blockchain**
Accessibility	Public blockchains are open to anyone to participate.	Private blockchains restrict access to a selected group of participants.
Control	Public blockchains are decentralized with no single entity in control.	Private blockchains are managed by specific organizations or consortia.
Transparency vs. Privacy	Public blockchains offer full transparency. Whole transactions can be viewed by every participant of the blockchain network.	Private blockchains provide more privacy and control over data. Some transactions can stay private.
Speed and Scalability	Public blockchains are slower compared to private blockchains due to consensus algorithms.	Private blockchains can operate faster and be more scalable due to their restricted size and optimal processes.

2.3.5. Hyperledger Fabric

Hyperledger Fabric (HLF) is a highly modular and configurable open source blockchain platform, part of the Hyperledger suite hosted by the Linux Foundation. It's specifically designed for enterprise use, offering unique blockchain approaches that differ significantly from public blockchain systems. HLF provides a secure, scalable, and modular architecture ideal for various industry use cases, especially where privacy, confidentiality, and scalability are crucial [69]. Key features of HLF platform are listed below.

- **Permissioned Network:** Unlike public blockchains, HLF is a permissioned network, which means that participants are known to each other and have specific roles and permissions within the network.
- **Modular Architecture:** HLF's architecture is highly modular, allowing network designers to plug in their preferred components like consensus and membership services. This modularity makes it adaptable to a wide range of industry use cases.
- **Channels for Data Partitioning:** HLF supports the creation of channels, allowing a group of participants to create a separate ledger of transactions. This is useful for ensuring data privacy and confidentiality among specific network participants.
- **Chaincodes:** In HLF, smart contracts are referred to as chaincode. They are used to implement business logic and can be written in standard programming languages like Go, Java, and Node.js, making them accessible to a broad developer audience.
- **Pluggable Consensus Mechanism:** Unlike blockchains that use PoW or PoS, HLF's consensus mechanism is pluggable. It supports a variety of consensus methods, allowing organizations to choose the most suitable one for their specific needs.

Key components of HLF's structure include peers, orderers, channels, and membership services each playing a unique role in the network's functionality. Whole list of components of HLF network is given in Table 2.2.

Table 2.2 Building Blocks of HLF Network

Component	**Role**
Peers	Peers are responsible for maintaining the ledger and state of the network. They execute chaincode, endorse transactions, and interface with applications. There can

	be two types of peers: Endorsing Peers and Committing Peers.
Ordering Service (Orderers)	Orderers batch transactions into blocks and deliver them to peers for final validation and commitment to the ledger. They enforce the order of transactions and ensure consistency across the network. Orderers facilitate consensus on the transaction order but don't participate in transaction validation or chaincode execution.
Channels	Channels provide a private layer of communication between specific network members. Each channel has its own ledger, allowing a subset of the network to transact privately.
Membership Service Provider (MSP)	MSP is responsible for managing identities and authenticating participants on the network. It defines rules for identity validation and access control. MSPs issue and manage certificates, providing a way to verify the legitimacy of each participant's identity in the network.
Chaincode	Chaincode implements the business logic, defining the rules for transactions. It’s similar to smart contracts in other blockchain platforms.
Ledger	Each channel has its own ledger, comprising a blockchain for the transactions and a world state database such as CouchDB [70] or LevelDB [71] for the current state.

Whole flow of a HLF network can be seen in Figure 2.9.

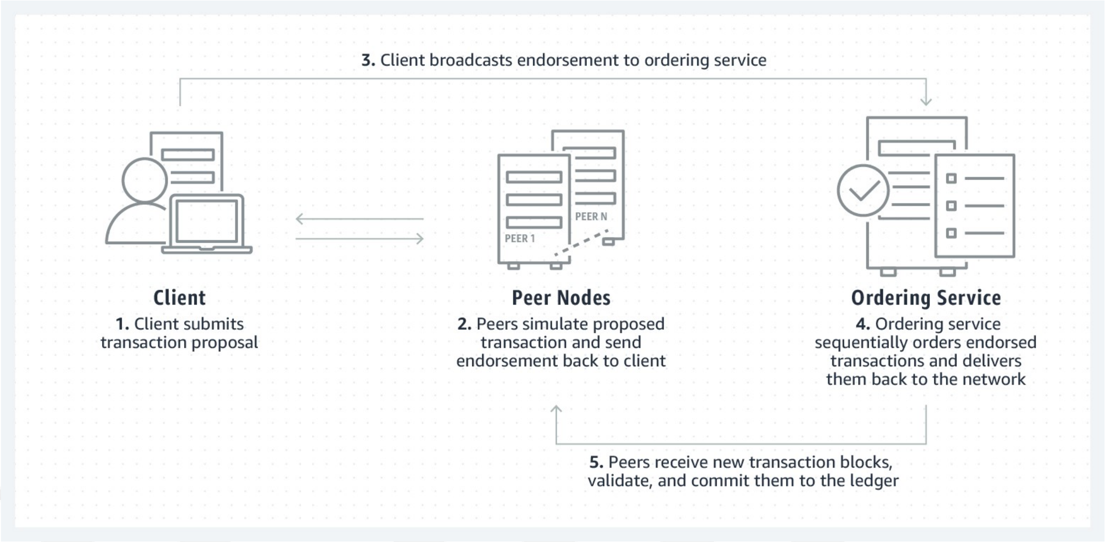

3. Client broadcasts endorsement to ordering service
PEER N
PEER 1
Client
1. Client submits transaction proposal
Peer Nodes
2. Peers simulate proposed transaction and send endorsement back to client
Ordering Service
4. Ordering service sequentially orders endorsed transactions and delivers them back to the network
5. Peers receive new transaction blocks, validate, and commit them to the ledger

Figure 2.9. HLF Workflow [72]

2.3.6. Hyperledger Caliper

Hyperledger Caliper is a blockchain benchmarking tool that falls under the Hyperledger umbrella project hosted by the Linux Foundation. As a performance benchmark framework, Caliper allows users to measure the performance of a specific blockchain implementation with a set of predefined use cases. This tool is critical for evaluating and ensuring that a blockchain solution meets the required performance standards in various scenarios [73]. Full architecture of Caliper is given in Figure 2.10.

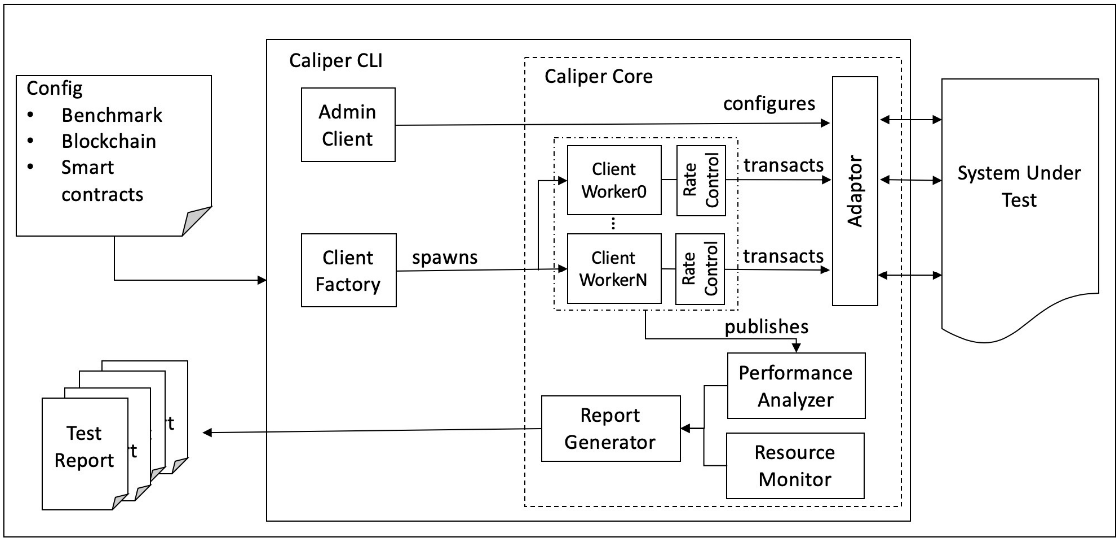


Figure 2.10. Hyperledger Caliper Architecture [74]

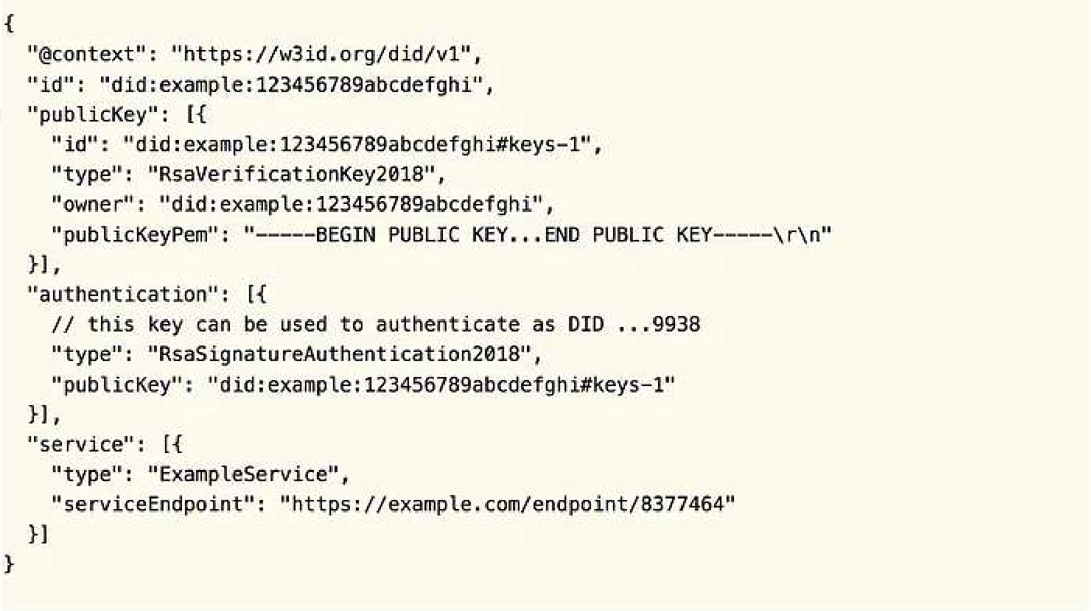
Caliper's architecture can be broadly described in two main components: Core and Adaptors. Caliper Core is responsible for implementing essential functions required to run a benchmark. Caliper Adaptors is an abstraction framework that is used to integrate different blockchain systems into the Caliper framework. Each adaptor is designed to fulfill specific blockchain platform requirements and is responsible for mapping operations to that platform's native capabilities [74].

2.4. Decentralized Identity

Decentralized identity (DID) is considered a new approach in digital identity management, which represents a move away from traditional centralized authority models towards a user-centric approach. DID leverages blockchain and related technologies to give individuals control over their own identity information, creating a more secure, private, and efficient system. Users create their DIDs, which are recorded on a distributed ledger, like a blockchain. Each DID is associated with a DID document containing public keys, authentication protocols, and service endpoints, enabling control over the identity. DIDs use cryptographic keys (public and private keys) for security. The user maintains control over their identity by keeping the private key secret and sharing the public key in an open manner. Replacing traditional PKI, DPKI in decentralized systems allows users to prove control over their DIDs and manage their identity records securely. DID specification is maintained by the W3C Credentials Community Group. It provides a standardized method for creating, resolving, updating, and deactivating decentralized digital identities without dependency on centralized registries, identity providers, or certificate authorities. DIDs are at the heart of W3C specification, where they are unique identifiers that enable verifiable, self-sovereign digital identities. DIDs are Uniform Resource Identifiers (URIs) that can be independently created and managed.

2.4.1. Decentralized Identity Documents

A DID document is a JSON document that contains specific information related to a DID. This information generally includes public keys, authentication protocols, and service endpoints. Public keys in the document are used to verify digital signatures and encrypt messages to the DID subject. They play a role in the cryptographic processes that are used under the security mechanisms of decentralized identity systems. DID documents also contain information about the DID subject's authentication methods, such as specific cryptographic keys or other verification methods necessary for digital interactions. Additionally, information related to service endpoints is available in a DID document. Service endpoints are URIs listed in the DID document that enable interaction with the DID subject. They indicate where and how to access services provided by the DID subject. An example DID document is given in Figure 2.11.



```
{
  "@context": "https://w3id.org/did/v1",
  "id": "did:example:123456789abcdefghi",
  "publicKey": [{
    "id": "did:example:123456789abcdefghi#keys-1",
    "type": "RsaVerificationKey2018",
    "owner": "did:example:123456789abcdefghi",
    "publicKeyPem": "-----BEGIN PUBLIC KEY...END PUBLIC KEY-----\r\n"
  }],
  "authentication": [{
    // this key can be used to authenticate as DID ...9938
    "type": "RsaSignatureAuthentication2018",
    "publicKey": "did:example:123456789abcdefghi#keys-1"
  }],
  "service": [{
    "type": "ExampleService",
    "serviceEndpoint": "https://example.com/endpoint/8377464"
  }]
}
```

Figure 2.11. Example DID Document [75]

2.4.2. Verifiable Credentials

VCs are digital versions of traditional credentials such as driver's licenses or university diplomas; however, they are designed for secure and verifiable usage within digital environments. VCs are based on the idea that any claim made by an individual or entity can be presented in a digital format and verified online. A VC typically includes information about the subject, the issuer, and the specific claim or claims being made. Additionally, VCs are digitally signed by the issuer using cryptographic techniques. This digital signature verifies a credential, as it provides a secure and tamper-evident way to validate its authenticity and integrity.

The process of using VCs involves three parties: the issuer, the holder, and the verifier. The issuer is the authority that issues the VC, the holder is the individual or entity that the VC is about, and the verifier is the individual or entity that needs to check the credential's validity. When a holder presents a VC to a verifier, the verifier uses the issuer's public key to verify the digital signature. If the signature is valid and the credential has not been tampered with, the verifier can be confident in the credential's authenticity. Process of using VCs is summarized in Figure 2.12.

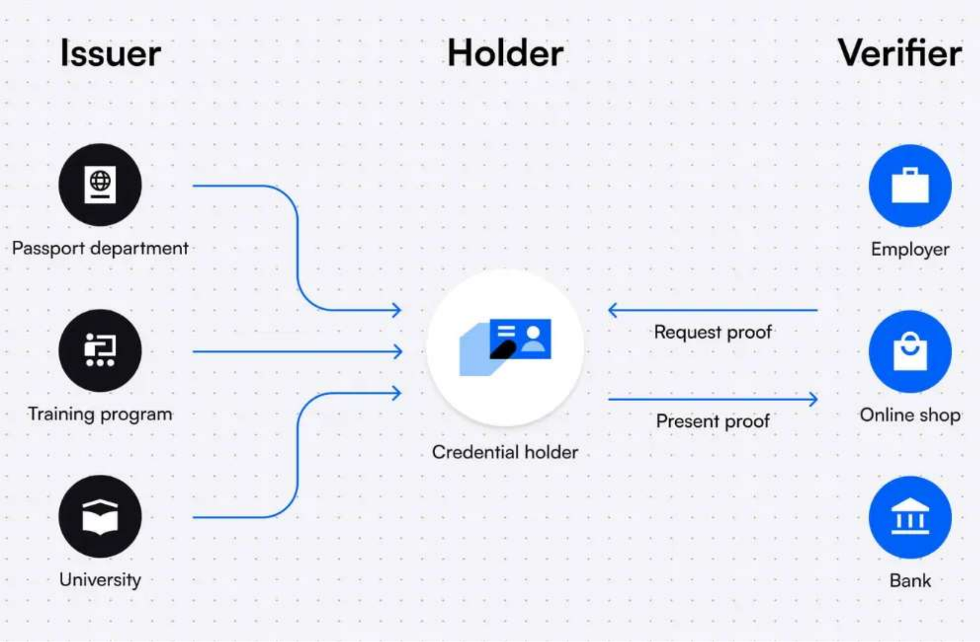


Figure 2.12. DID System Roles [76]

2.5. Related Works

The issues related to the centralization of IAM and IdM in IoT networks have gained traction in recent years, and several studies have been conducted on this topic. Some researchers focused on improving the existing centralized solutions, whereas others proposed novel approaches by mixing the latest technology in decentralized ledgers with the current systems. This thesis subsection explores various related works on DID and IoT IAMs.

In [77], a blockchain based IdM solution is proposed to establish a unique, global digital identity for IoT devices which is maintained through their lifecycle. The approach includes mechanisms for device ownership management and identity updates. The framework aims to establish a global registry for IoT devices, develop identity lifecycle processes, provide clear registration processes, handle device ownership management, and trace device lifecycle. Study in [78] also proposes a IAM framework which is DLT based for IoT devices. The framework adapts DIDs and VCs to establish an immutable and universal device identity registry on the blockchain. By doing so, authors try to enhance interoperability on global scale IoT networks and provide functionalities to manage IoT device lifecycles.

System proposed in [79] aims to create secure virtual zones (bubbles) where IoT devices can authenticate each other. It leverages public blockchain technology and smart contracts to implement this secure environment. The system allows identification of IoT devices and enables authentication methods in a decentralized manner. Another study in [80] introduces a blockchain based solution for IoT security, aiming to resolve critical issues in device authentication and data integrity. With the limitations of IoT devices such as processing power, battery life, and storage space, an adaptable security architecture for IoT is proposed. The approach aims to align with the decentralized nature of IoT applications, aiming to build a secure ecosystem through interoperability. The architecture contains two distinct blockchain networks. The first network is used to handle servers and the second manages the IoT devices. By having two separate networks, authors aim to obtain an immutable transaction audit between servers and IoT devices. Work in [81] a novel framework is designed to address challenges of digital identity and security in the Industrial Internet of Things (IIoT). The framework is designed to facilitate faster onboarding of a large number of devices, ensuring secure, scalable, and privacy preserving interactions. It supports digital identity creation and the issuance and verification of VCs in a decentralized manner. This approach integrates blockchain technology with a decentralized data storage layer, such as the Inter Planetary File System (IPFS), to establish secure and remote communications for M2M communication.

Study in [82] presents a framework aimed at addressing trust issues in IoT ecosystems through a decentralized approach. It introduces a reputation-based trust evaluation model, which aims to provide a method for IoT devices to evaluate their trust in a decentralized manner. Framework employs DLT to decentralize key mechanisms such as access control, identity management. Additionally, the DLT framework supports resilience and accountability through data provenance, which integrates with IoT network infrastructures to maintain an auditable log of immutable records for all relevant interactions within the network.

The thesis study in [83] proposes an IAM solution that utilizes the principles of decentralization inherent in DLT and the automation capabilities of smart contracts. The core of the proposed framework employs DIDs for a decentralized identity management system and leverages blockchain tokenization, including fungible and non-fungible tokens (NFTs). This approach is designed to establish a self-controlled and self-contained access control policy based on the Capability Based Access Control (CapBAC) model. By building on identity management as the foundation, the framework presents decentralized authentication and authorization processes and creates a mechanism for accounting using a standardized DLT tokenization structure.

The multifaceted approach of our proposed framework sets it apart from comparable studies. In addition to decentralized IdM feature, it integrates a range of other schemes, such as device authentication, device discovery, device authorization and access control, and trust evaluation. Each of its mechanisms underscores the framework's dedication to a decentralized approach, as they are designed to operate in a decentralized manner. Section 3 offers an in-depth exploration of every scheme and its distinct role in the broader structure.

3. THE PROPOSED FRAMEWORK

This thesis proposes a comprehensive blockchain-based IAM and trust evaluation framework designed to operate in an environment where billions of devices are interconnected through IoT technology. Day by day, more devices are anticipated to interconnect with the improvements in communication technologies such as 5G and beyond. A large-scale IoT network, combined with numerous devices using varied protocols and standards, might result in many IAM systems existing together in a single network. Such a scenario risks fragmenting the network into parts, causing issues related to scalability and interoperability, and blocking the goal of a seamlessly connected global IoT system. Our proposed solution is a robust, secure, unique, IAM and trust evaluation framework that leverages smart contracts and the inherent decentralization features of blockchain technology. The architecture of the framework is strategically layered, comprising the IoT Device Layer, the Network Infrastructure Layer, and the Blockchain Layer. An overarching architectural overview of our framework is depicted in Fig 3.1. providing a visual representation of the integration and interaction of these layers.

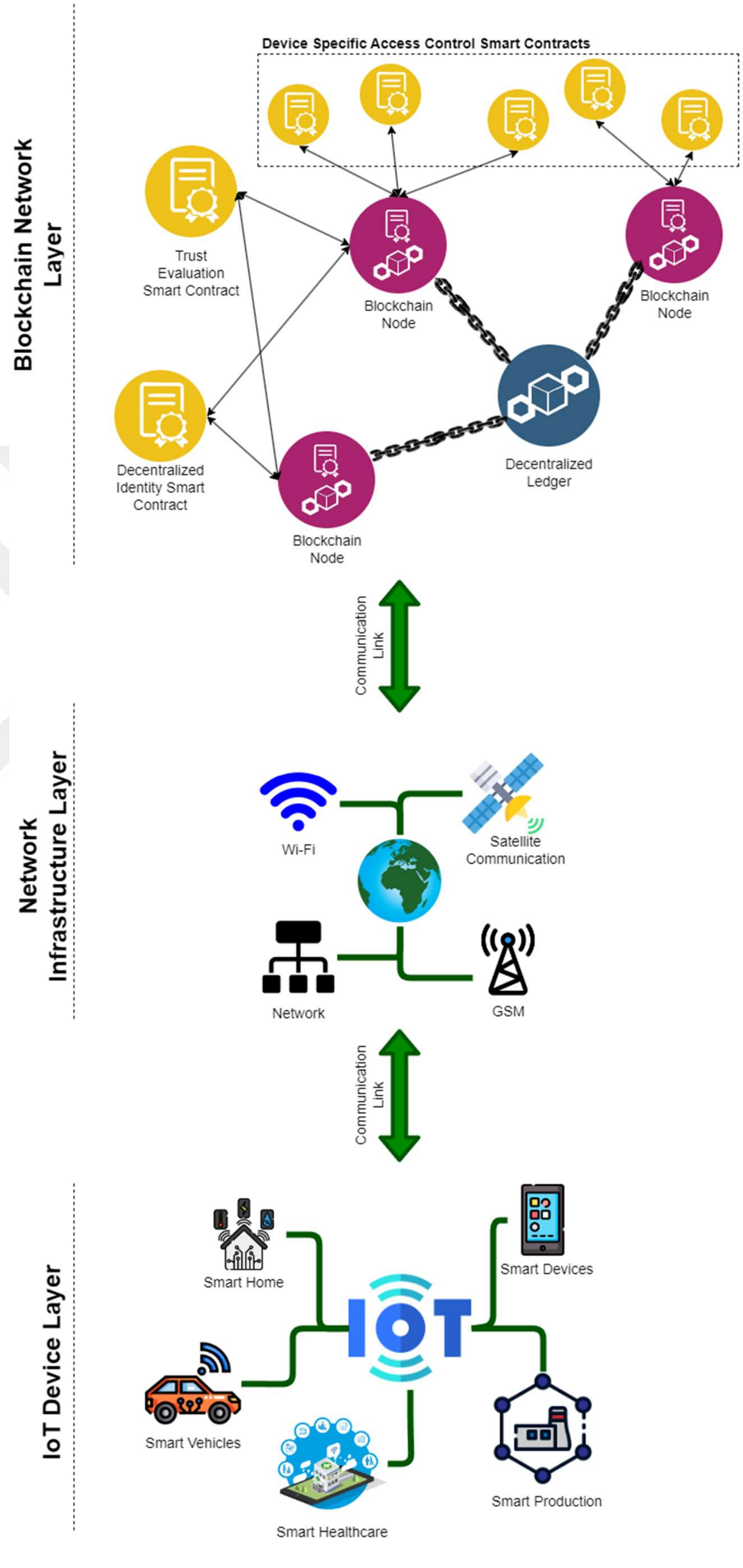


Figure 3.1 Architecture Overview of Proposed Framework

Functionality of the IoT Device Layer and Network Infrastructure Layer within the overall framework is summarized below.

- **IoT Device Layer:** This layer is the foundational level where the actual IoT devices reside. It consists of various "things" or endpoints, such as sensors, actuators, smart appliances, wearable devices, and more, each potentially with computing and communication capabilities. These devices are responsible for collecting data from their environment or performing specific actions based on commands received from the network. The IoT Device Layer is crucial as it serves as the point of interaction with the physical world and generates the data that fuels IoT applications.
- **Network Infrastructure Layer:** The Network Infrastructure Layer is responsible for maintaining connections between the applications, devices, and the proposed framework. It provides the communication pathways that allow IoT devices to connect and interact with each other and with systems or applications. This layer may include a variety of communication technologies and protocols, which may consist of NB-IoT and MQTT. The Network Infrastructure Layer ensures that data can be transmitted between the IoT devices and the systems that process and analyze this data, such as cloud services or blockchain nodes. It plays a crucial role in the scalability and performance of IoT applications, as it must handle potentially large volumes of data traffic and a wide variety of device types and communication needs.

The IoT Device Layer and Network Infrastructure Layer are integral to the framework, serving as the groundwork for implementing the blockchain-based solution. However, it is essential to note that this thesis study's main contribution and primary focus lies within the Blockchain Network Layer. The first two layers are only studied to show the audience that the proposed solution operates on top of these two layers.

In the Blockchain Network Layer, we've set up three unique smart contracts, each serving a critical function in the proposed IAM and trust evaluation framework. The cornerstone of the framework can be identified as the Decentralized Identity SC. This contract acts as a storage and verifier for registered device identities in the blockchain network, serving as a base of trust. It allows devices to verify and get reliable information about each other in a decentralized manner, thus facilitating a secure and reliable IoT environment. Another key element is the Trust Evaluation SC

deployed within the proposed framework. It is responsible for monitoring device interactions and assigning a trust score to each device in the network. This trust score is initialized and maintained for each device following the device registration on the "Decentralized Identity Smart Contract." These scores are then used by devices to validate the trust of other devices within the network. Furthermore, trust scores allow devices to manage their access levels against other devices, enabling them to interact independently within the IoT ecosystem.

Finally, we have Device Specific Access Control SCs. These are used to create and enforce access rules for either individual devices or groups, as set by their owners. These contracts use trust score limits to define access rules. When a device seeks to establish a connection with another, the contract evaluates if the requesting device meets the required trust level. If it does, a unique access token for that device is created. These tokens, which are predefined data structures, contain the necessary data for maintaining secure connections with other devices. They allow devices to safely access resources provided by other devices or applications in the IoT ecosystem.

In short, the proposed framework represents a paradigm shift in how identity-related processes are secured and managed within global-scale IoT networks. By leveraging the decentralized and immutable nature of blockchain technology, coupled with the dynamic capabilities of smart contracts, we offer a solution that addresses current security, interoperability, and scalability challenges in IoT and lays a foundation for the future evolution of IoT ecosystems. In order to provide decentralized IAM and trust evaluation functions to IoT devices, the proposed framework defines several methods, such as decentralized device identity modelling, device registration, authorization and access control, mutual authentication, and trust evaluation. These methods will be explained in detail in the upcoming sections.

3.1. Decentralized Device Identity Model

A verifiable, reliable, globally accessible, and unique identifier is needed for the proposed framework to correctly identify and verify the identity of an IoT device within the network. This identifier is referred to as decentralized device identity, and it serves as a digital fingerprint of individual devices inside the framework. This identifier is used by other devices to access public information about another and enables the verification of the device's identity throughout the network. The decentralized device identity is a data structure containing specific data fields related to the device it corresponds to. This data structure is maintained and stored in the “Decentralized Identity SC” in a decentralized manner, and it is accessible to all the devices operating within the proposed framework. In the current scheme of the framework, each device can have a single decentralized device identity registered by the device's owner into the Decentralized Identity SC. The decentralized device identity data structure and its data fields are shown in Table 3.1.

Table 3.1 Decentralized Device Identity Data Structure [84]

Data Field	Definition
Device Address	Indicates the device’s public blockchain address. It is mainly used in mutual authentication scheme.
Owner Address	Indicates the device owner’s public blockchain address. Only a transaction signed by the owner is authorized to update the specific data fields within the decentralized device identity structure.
Device Description	The device’s type, creation date, and modification dates can be seen in the device description field. For instance, with each modification on the data structure, modification dates are logged in the device description field.
Hashed Attributes	The hash value of the device attributes is contained in this data field. The device can

	use this value to prove that it has these features when it shares its identity with another entity. Examples of device attributes include a serial number, manufacturing date, firmware version, and hardware specifications.
Discovery-Specific Data	The devices within the proposed framework may search for other devices using the device discovery-specific data metadata. This field may include publicly available data regarding the device's resources and services. Based on this data field, other devices may make informed decisions on whether to communicate with a device or not.
Device Specific Access Control SC Address	Access and authorization are controlled for devices via Device Access Control SCs. This data field contains the necessary data (public blockchain address) to establish a connection with the device's access control SC. The device-specific access control SC enforces the access control policies set forth by the owner of the device. Additionally, device-specific access tokens are also generated by this contract.

Adding more data fields to the decentralized device identity data structure is possible to enable use-case-specific mechanisms or different methods. However, it is essential to remember that this data structure is stored publicly on the blockchain network and managed by the Decentralized Identity SC, which is publicly accessible by other devices within the framework.

3.2. Device Registration

The device manufacturer initiates the creation of the decentralized device identity at the time of the device's production - a stage we term the device's 'birth.' At this stage, the manufacturer generates a unique public-private key pair linked to the device. This key pair is used as the blockchain credentials of the device, forming the cornerstone of its digital identity and ensuring secure interactions within the blockchain network. After creating a key pair, the manufacturer proceeds to construct a decentralized device identity data structure that is uniquely mapped to the device. Following the creation of identity, the next step involves the registration of this identity within the “Decentralized Identity SC.” The manufacturer undertakes to register the device's identity onto the blockchain. This registration is considered a critical event, marking the device's formal introduction into the blockchain network and, hence, the proposed framework. Once the registration is successfully concluded, the blockchain credentials and the newly minted digital identity are written onto the device itself. This action acts as the enabler that allows the device to be recognized and authenticated by other participants in the network. It assures that the device can be trusted, having an identity that is not only unique and globally accessible but also anchored in the security and immutability offered by the blockchain. Whole sequence of the device registration process is summarized in Figure 3.2.

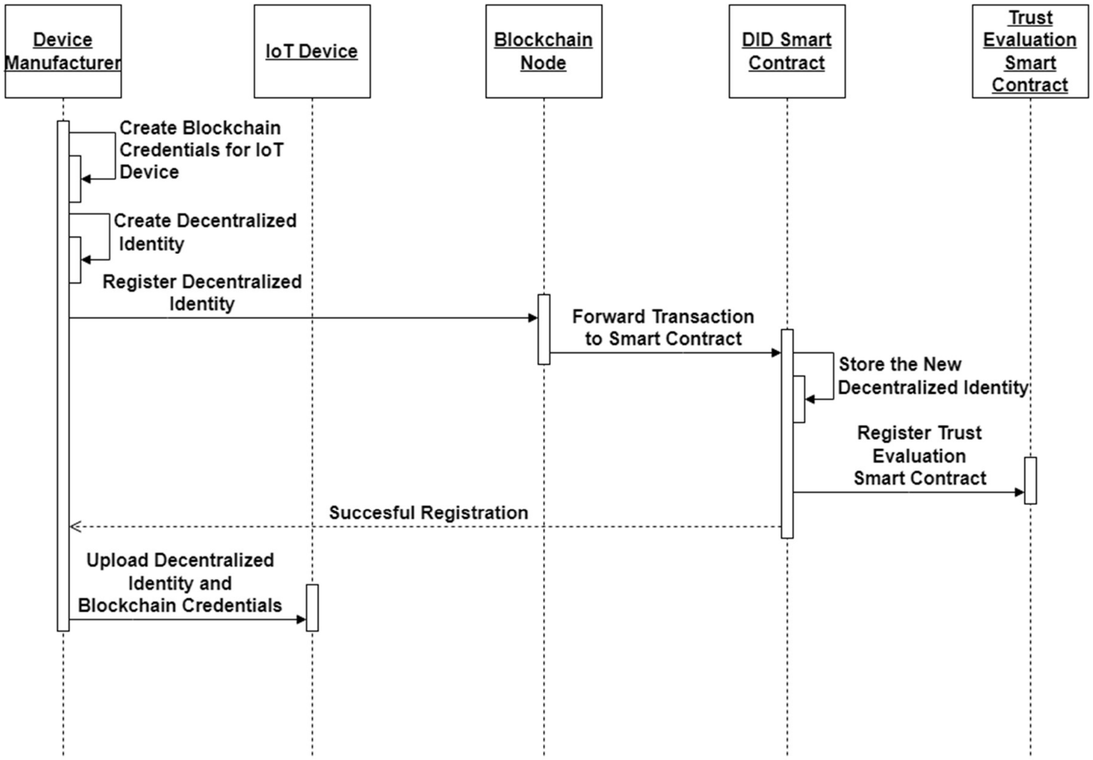

Device Manufacturer
IoT Device
Blockchain Node
DID Smart Contract
Trust Evaluation Smart Contract
Create Blockchain Credentials for IoT Device
Create Decentralized Identity
Register Decentralized Identity
Forward Transaction to Smart Contract
Store the New Decentralized Identity
Register Trust Evaluation Smart Contract
Succesful Registration
Upload Decentralized Identity and Blockchain Credentials

Figure 3.2 Sequence Diagram of Device Registration Process

Upon registration, the device identity becomes an immutable certificate of the device's credibility and legitimacy within the blockchain network. It ensures that as the network scales and diversifies, the integrity of device interactions remains intact. This approach not only solidifies trust within the network but also establishes a harmonized ecosystem where devices, despite their disparate origins and functions, can interact and collaborate seamlessly under a unified identity framework.

3.3. Device Lifecycle Management

Lifecycle management of a device identity is a process that ensures that the digital representations of IoT devices remain accurate and up-to-date throughout their operational lifetime. The lifecycle management contains several critical functions: identity creation, updating, ownership transfer, and eventual decommissioning of device identities. At the inception of a device's lifecycle, the manufacturer initiates the creation of the device identity. This process involves generating a unique digital identity containing the device's essential attributes and its blockchain credentials within the Decentralized Identity SC. As devices may undergo changes or require new configurations during their lifetime, the proposed framework includes a method for updating device identities. This method allows for altering the device's data fields within its decentralized identity, reflecting any changes in its status, capabilities, or ownership. These updates are transactions that are securely logged on the blockchain, maintaining an immutable history of the device's evolution. Additionally, the Decentralized Identity SC ensures that only the device owner can modify the identity of a device within the framework.

Transferring ownership of a device is a common occurrence in the lifecycle of an IoT device. The proposed framework accommodates this through a 'change-ownership' method within the Decentralized Identity SC. This method enables the current owner to transfer the rights and control over the device's identity to a new owner. The transaction is recorded on the blockchain, providing a verifiable trail of ownership, and the new owner is granted the ability to update the device identity as needed. When a device reaches the end of its service life or is otherwise retired from the network, it is crucial to have a secure method for decommissioning. The "delete-device" method within the Decentralized Identity SC is designed for this purpose. It allows for removing the device's identity from the active registry within the smart contract, effectively revoking its access and authentication within the network. This action helps maintain the network's integrity, ensuring that only active and valid devices participate in the IoT ecosystem.

Security and integrity are of great concern throughout each phase of the device identity lifecycle. All interactions with the smart contract are cryptographically secure, and the blockchain's inherent properties ensure that the entire lifecycle is transparent and tamper-proof. Whether a device is undergoing an update, changing hands, or being decommissioned, each transaction is an immutable entry on the blockchain, providing transparent and trustworthy records. In short, the proposed framework provides a comprehensive scheme for device identity lifecycle management within the IoT space. It is a robust system that facilitates the secure and efficient management of device identities and fosters a trusted environment that fits the dynamic nature of IoT networks. Blockchain technology ensures that each stage of a device's lifecycle is transparent, immutable, and under the owner's control, paving the way for a secure and interoperable IoT future.

3.4. Device Discovery

A method is needed to avoid unnecessary communications within the IoT network and define a way for devices to find and learn about each other. This process is known as "Device Discovery". It allows a device to figure out if it should start the steps needed to connect to another device by checking if the other device has the right kind of resources available. To make this possible, necessary methods were implemented within the proposed framework that help devices identify each other using decentralized device identities. A sequence diagram of the device discovery method is given in Figure 3.3.

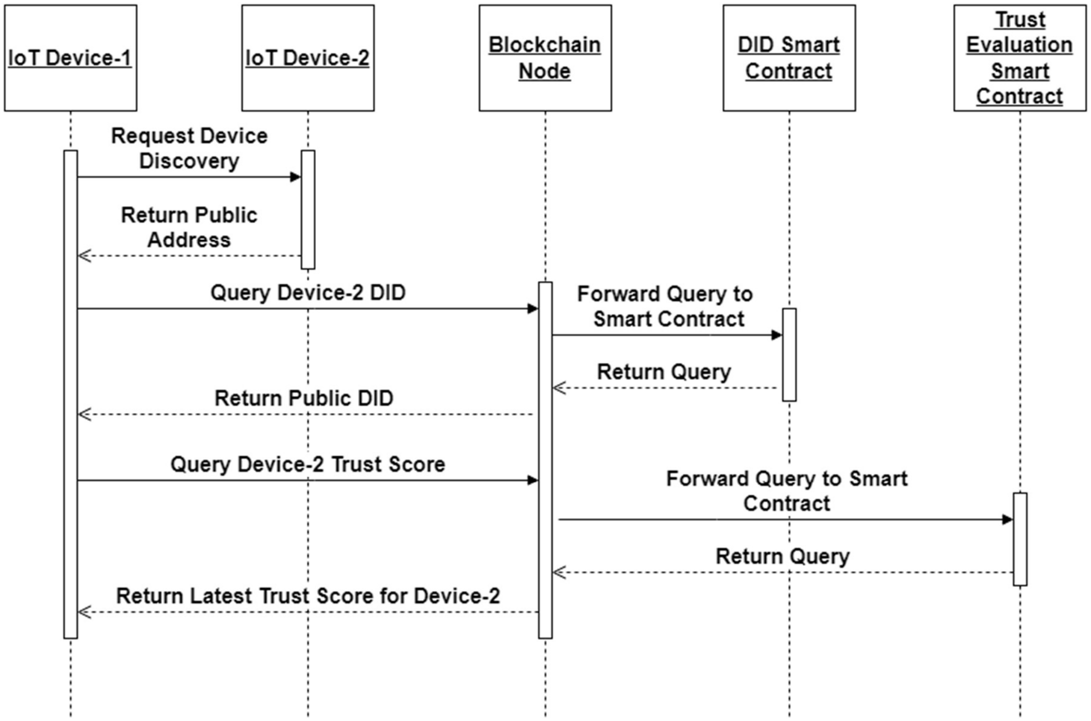


Figure 3.3 Sequence Diagram of Device Discovery Process

When a device wants to find other devices in the network, it first asks for their public blockchain addresses. Once it has the address, it can go to the Decentralized Identity SC and look up the identity information that matches the address. Each device has a specific piece of data in its identity information that tells other devices what it can do. Using this info, the device that's looking can determine whether it wants to connect.

3.5. Device Authorization and Access Control

In a global-scale IoT network, devices need a secure way to determine who gets to use their services and resources. This decision-making process is known as authorization and is typically guided by a set of rules or access control policies that an organization or network puts in place. To address this need within the proposed framework, a unique authorization protocol was designed to allow decentralized access control. The decentralized access control approach is built into the framework, allowing devices to make their own decisions on access based on rules set by their owners. These rules are directly linked to trust scores, a reliability rating for each device that the Trust Evaluation SC calculates within the framework. The access control policies use these trust scores to set thresholds or cut-off points. When one device asks to connect with another, the system checks if the asking device's trust score meets these thresholds. If a device meets the requirements and is allowed to connect, it gets a data structure referred to as an access token. This token is like a digital key, created by the Device Specific Access Control SC, that lets the device use the resources and services it's asking for. This token is specifically structured with different data fields, each carrying specific information, as outlined in Table 3.2.

Table 3.2 Data Fields of Access Token [84]

Data Field	**Definition**
Token Identifier	Specifies the token’s unique ID. It can be used to both query and validate the access token obtained from the Device Specific Access Control SC.
Public Blockchain Address of the Owner	Indicates the token owner’s public blockchain address. This token can only be used by the owner. The process of mutual authentication uses this field.
Public Blockchain Address of the Resource Owner	Indicates the IoT device’s public blockchain address, which is responsible for hosting the requested resources and services. Only the resources hosted by this

	address’s owner may be accessed with the relevant token.
Requested Resources	Only the resources and services listed under this data field are accessible using this token.

Every time a token is successfully created, and a connection is made, the Device Specific Access Control SC sends a signal to the Trust Evaluation SC. This message is crucial in influencing future trust score calculations for the device that asked to connect. Details of this process will be further discussed in Section 3.8. Whole flow of device authorization and interactions between entities during the process is given in Figure 3.4.

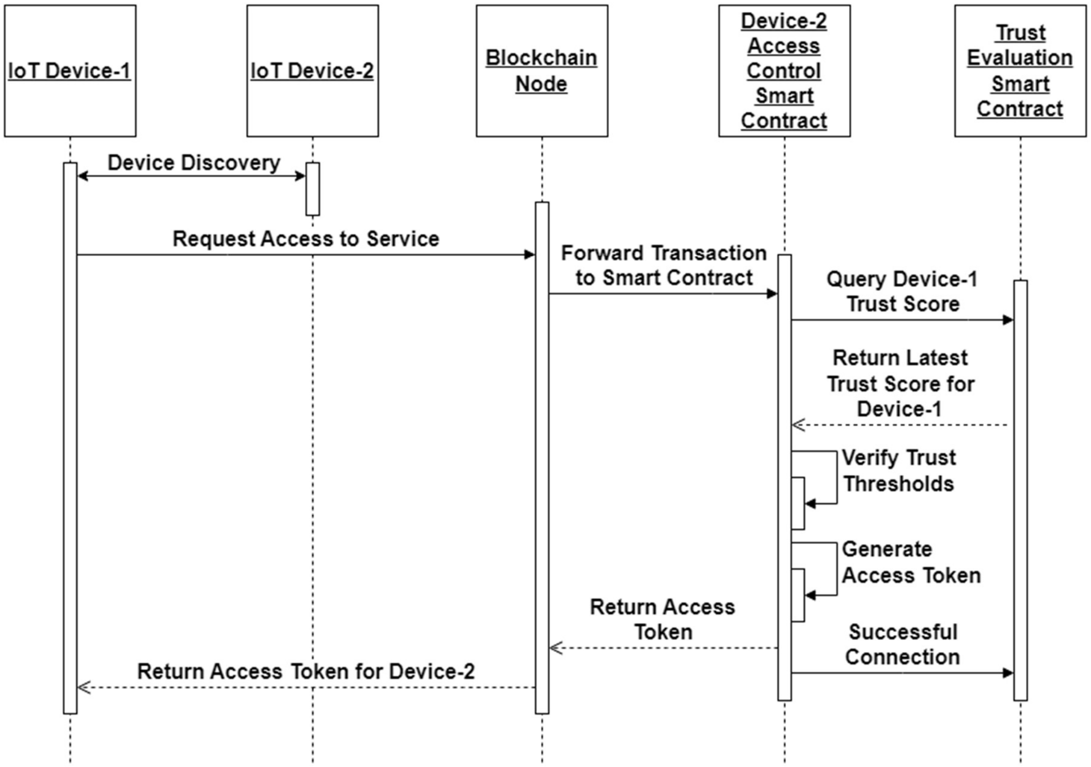

IoT Device-1
IoT Device-2
Blockchain Node
Device-2 Access Control Smart Contract
Trust Evaluation Smart Contract
Device Discovery
Request Access to Service
Forward Transaction to Smart Contract
Query Device-1 Trust Score
Return Latest Trust Score for Device-1
Verify Trust Thresholds
Generate Access Token
Return Access Token
Successful Connection
Return Access Token for Device-2

Figure 3.4 Decentralized Access Control and Sequence of Device Authorization

When a device wishes to access another device's resources, it presents its access token to the resource owner device. The owner device examines the token's fields to ensure the request is valid. It checks the token identifier to ensure the token is genuine, uses the owner's address to authenticate the requesting device, and verifies that the token allows the requested resources. This access token mechanism within the framework not only facilitates secure and efficient access to IoT resources but also allows for fine-grained control over what each device is permitted to do. It ensures that only authorized devices can interact with each other, enhancing the overall security and integrity of the IoT ecosystem.

3.6. Device Mutual Authentication

In the context of the proposed framework, mutual authentication of devices plays a vital role in establishing secure communications within the IoT environment. This chapter presents an authentication protocol that enables devices to verify each other's identity in a decentralized manner. The protocol makes use of the access tokens described in the previous section and various cryptographic methods. Access tokens act as digital keys, allowing a device to prove its identity and gain access to services or other devices inside the network. These tokens are secured through cryptography, ensuring they cannot be forged or tampered with. The approach to mutual authentication within the framework is designed to operate in a decentralized manner. This means each device inside the network can verify another device's identity independently. By enabling devices to mutually authenticate, the framework significantly reduces the risk of unauthorized access and ensures that communications between devices are secure and reliable. The flow of the mutual authentication scheme is given in Figure 3.5.

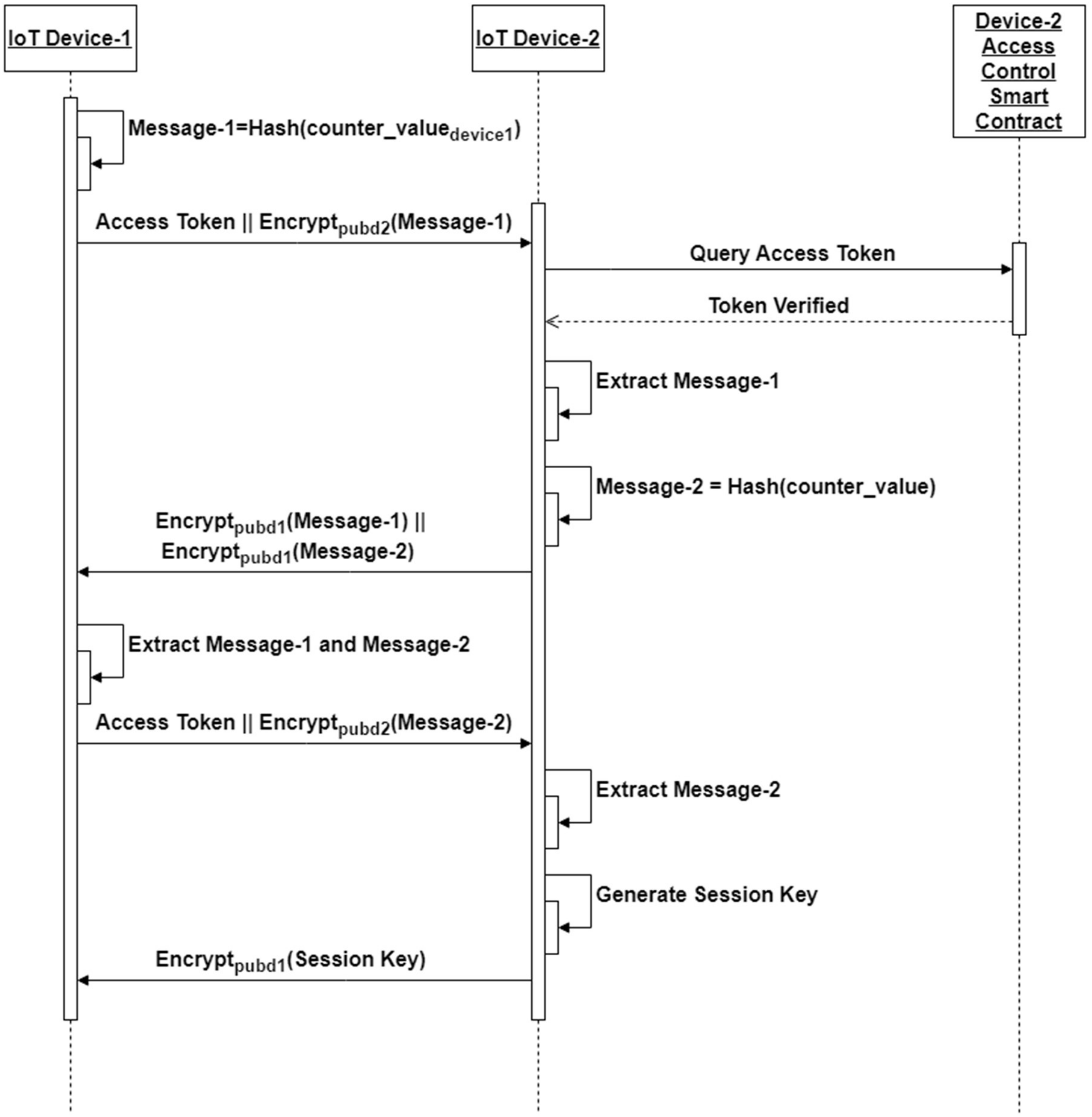

IoT Device-1
IoT Device-2
Device-2 Access Control Smart Contract
Message-1=Hash(counter_value$_{device1}$)
Access Token || Encrypt$_{pubd2}$(Message-1)
Query Access Token
Token Verified
Extract Message-1
Message-2 = Hash(counter_value)
Encrypt$_{pubd1}$(Message-1) ||
Encrypt$_{pubd1}$(Message-2)
Extract Message-1 and Message-2
Access Token || Encrypt$_{pubd2}$(Message-2)
Extract Message-2
Generate Session Key
Encrypt$_{pubd1}$(Session Key)

Figure 3.5 Mutual Authentication Protocol

Mutual authentication protocol uses various cryptographic primitives such as hashes and PKI. These primitives are actively used in flow diagrams; however, it is essential to note that these primitives can be changed with other cryptographic protocols. The proposed framework can be adapted to use different protocols as the cryptography technology evolves. The main idea of the Mutual Authentication scheme is that it uses decentralized access tokens to verify each device's identity without relying on a central third party.

3.7. Auditability of the Framework

Auditability in the IoT ecosystem is vital for several key reasons. Firstly, it enhances security. As IoT devices are often connected to critical infrastructure and personal data, the ability to audit these systems ensures that any security breaches or vulnerabilities can be quickly identified and addressed. This is essential in preventing data theft and unauthorized access and maintaining the integrity of the network. Secondly, auditability aids in compliance and regulatory adherence. Many industries are subject to strict regulations regarding data handling and privacy. The ability to audit IoT systems helps organizations demonstrate compliance with these regulations, avoiding legal penalties and maintaining public trust.

Integration of blockchain technology is essential to maintain auditability of the proposed framework. Blockchain acts as an immutable ledger, recording all transactions and interactions between IoT devices. Once recorded on the blockchain, each transaction cannot be altered or deleted. This immutability provides a trustworthy audit trail. Every interaction related to the framework, such as authentication requests and access control decisions, is recorded on the blockchain. These records are time-stamped and linked to previous transactions, creating a chronological and unalterable history. The blockchain ledger is transparent. While maintaining the confidentiality of sensitive data, the framework allows for the verification of transaction histories by authorized entities. This transparency aids in maintaining accountability across the network. The benefits of auditability within the proposed framework are listed below.

- **Security:** A robust audit trail helps in identifying and mitigating security breaches promptly. It also deters malicious activities, as actors know their actions are recorded.
- **Regulatory Compliance:** The framework's auditability ensures compliance with regulatory standards, which often require detailed records of data access and processing activities in IoT environments.
- **Operational Transparency:** Stakeholders, including device manufacturers, service providers, and end-users, can verify the integrity of their devices and data, fostering trust in the IoT ecosystem.

- **Forensic Analysis:** In the event of security incidents, the immutable logs serve as a reliable source for forensic analysis, aiding in understanding the incident's scope and impact.

Auditability in the proposed framework is not just a feature but a cornerstone that upholds the integrity and trustworthiness of the entire system. By leveraging blockchain’s inherent properties, the framework ensures that interaction and transactions within the IoT environment are transparent, traceable, and accountable.

3.8. Trust Evaluation Framework

Trust evaluation in an IoT ecosystem is a process that may involve the autonomous determination of trust through the calculation of trust scores for various devices and data exchanges within the network. This approach is essential in an IoT environment, where numerous devices with diverse security levels and capabilities are interconnected. Trust scores calculated based on factors such as past behavior, security credentials, and interaction history, provide a quantifiable measure of a device's reliability and integrity. This automated and dynamic assessment allows for real-time identification and mitigation of potential risks from compromised or malicious devices.

This chapter introduces the trust evaluation mechanism within the proposed framework, designed to enhance security and credibility. The core idea is to measure and evaluate how trustworthy each device is in a decentralized manner. Framework evaluates the trustworthiness of devices using a trust score calculation. This score, referred to as Ω, is derived from two key parameters:

- **ITS (α):** This score represents the initial level of trust placed in a device. It considers factors like the reputation of the device's manufacturer and adherence to security standards. The calculation of α is based on information stored in the device's DID. It is estimated that the Trust Evaluation SC calculates the ITS upon the device's birth. This score calculation may be based on the device description field of the decentralized identity data structure, and based on this information, the contract may automatically calculate the ITS.

- **InTS (β):** This score evolves according to the device's interactions within the network. It considers the number of successful interactions (X) and the amount of negative feedback received from other entities (Y).

Calculation of InTS is given in Eq. 1.

$$\beta = w_x . X - w_y . Y \quad (1)$$

β: InTS
w_x: Adjustable weight constant for X
X: Amount of positive feedback
w_y: Adjustable weight constant for Y
Y: Amount of negative feedback

In this equation, w_x and w_y are weight values assigned to positive and negative feedback, respectively. These values introduce the ability to adjust the InTS calculation based on the consensus of the participants of the framework. A mechanism to adjust these values can be added to the Trust Evaluation SC where the adjustment would require a consensus between the IoT device manufacturers or the stakeholders of the proposed framework.

3.8.1. Positive Feedback Mechanism

The positive feedback mechanism allows devices to acquire higher trust scores within the framework based on their successful interactions with other devices. After a Device-Specific Access Control SC generates access tokens, it sends a confirmation message to the Trust Evaluation SC. This message acts as positive feedback for the requesting device, indicating successful and secure interactions. This process helps incrementally increase the InTS of the device, as it demonstrates its reliability and compliance with the framework's policies. Additionally, the positive feedback calculation is based on the trust score of the device that gives the feedback. For instance, a device with higher trust scores may weigh more on their feedback than devices with lower scores. This mechanism allows reputable devices to weigh more on the overall framework than inactive or less trustable ones. However, this could potentially lead to a bias on the device's age as older devices would eventually have more interactions and higher trust scores. In order to prevent this, an aging factor was introduced to the framework. Based on this aging factor, the Trust Evaluation SC decreases an aging factor from each device's trust score on predetermined periods, forcing devices to stay active within the framework.

3.8.2. Negative Feedback Mechanism

Unlike positive feedback, collecting negative feedback from devices may present unique challenges, as relying solely on devices for negative feedback can be risky in cases where cyber-attacks might compromise a device and force it to generate fraudulent feedback. This could lead to incorrect trust assessments and potentially destabilize the trust framework. To address this issue, a manufacturer-based voting system was proposed where, instead of relying solely on device-to-device feedback, a consensus between the major stakeholders would be reached before processing the negative feedback. Whenever a negative feedback transaction reaches the framework, a voting mechanism, similar to consensus on the blockchain technology, would commence, and based on the result of this voting, the device would receive the negative feedback. This process is further automatized via the event feature of smart contracts, where each major manufacturer would have a central node that scans the network for negative feedback events. With each event trigger, voting would begin between the stakeholders, resulting in the acceptance or decline of the negative feedback transaction. Additionally, votes from different stakeholders can have different weights based on their credibility and role in the ecosystem. The framework can achieve a more balanced and accurate trust evaluation by involving manufacturers as major stakeholders in the negative feedback mechanism and setting clear criteria for negative behavior. This approach enhances the overall security and reliability of the IoT ecosystem, ensuring that trust scores reflect the true behavior and status of the devices in the network.

4. RESULTS AND DISCUSSION

This section outlines the empirical findings derived from implementing the proposed framework on a test network on the Hyperledger Fabric platform. The test network serves as a miniature version of the proposed system for IoT devices, offering a controlled environment to investigate the system's performance, security, and scalability aspects. Deriving the results from the test network starts with the network deployment step, which marks the transition from theoretical design to execution within the thesis study. Here, we discuss the network configuration, the rationale behind the choice of parameters, and the setup process. After the successful deployment of the network, a performance evaluation was conducted to demonstrate the feasibility of the proposed system. Various testing tools have been employed to measure the network's throughput, latency, and resource utilization. Additionally, monitoring tools were used to observe the system's behavior under different conditions. The stability and robustness of the network and the design were assessed through continuous monitoring, allowing for the detection of anomalies and performance bottlenecks. The implications of the test results were explored in depth in the discussion part. The performance and monitoring outcomes are examined in the context of the network's design goals and the broader field of IoT IAM. The upcoming sub-sections will give in-depth information about the implementation, performance evaluation, cyber-security analysis, possible use cases, and future work of the proposed framework.

4.1. Implementation

This thesis proposed a blockchain-based decentralized identity and trust evaluation framework for use in global-scale IoT networks. In order to analyze the performance, functionality, and cyber resilience of the proposed framework, an implementation was made using the Hyperledger Fabric blockchain platform. This platform was selected for its ability to operate within a permissioned environment, a feature prohibiting participants from joining the network without specific permission. Unlike public blockchain implementations, the permissioned nature of Hyperledger Fabric allows for a network configuration, defining explicit participant permissions and roles, which is essential for maintaining the integrity and confidentiality of IoT device interactions. Moreover, the permissioned framework of Hyperledger Fabric provides a performance advantage due to its efficient consensus mechanism that eliminates the computational overhead seen in public blockchains. This efficiency is crucial in scenarios where IoT devices require rapid transaction processing and identity-related functions. The architecture of the implementation, as depicted in Figure 4.1., was methodically constructed to harness these advantages, ensuring a secure, scalable, and high-performing decentralized IoT IAM system that could be used in the dynamic IoT ecosystem. The source codes of the implementation discussed under this chapter can be accessed via [85].

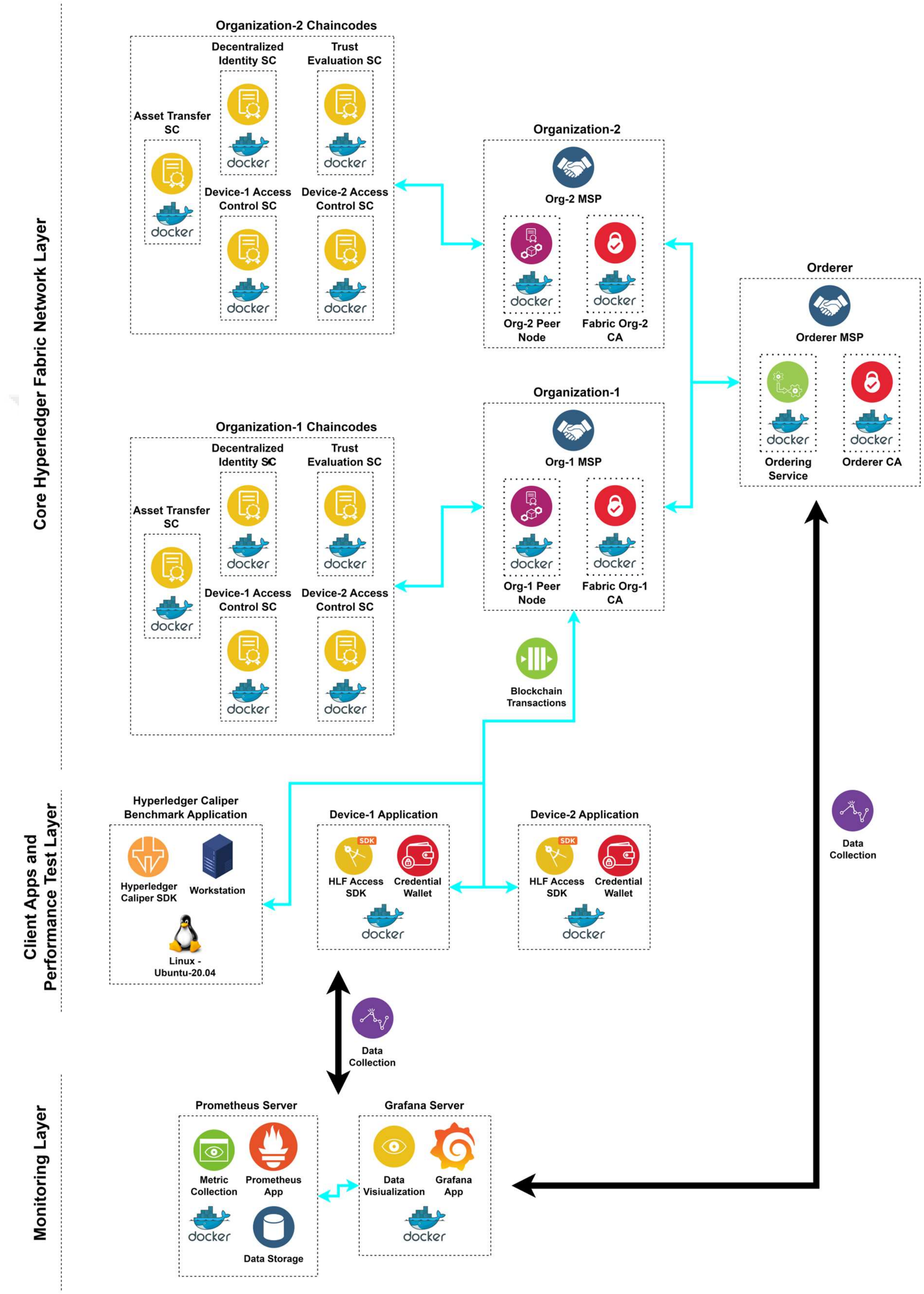


Figure 4.1 Architectural Overview of the Implementation

The architecture of the implementation was designed to enable each function of the proposed framework, therefore facilitating identity management, access management, and trust evaluation across multiple IoT devices and organizational boundaries. The implementation consists of three layers, each containing different applications related to core architecture, performance evaluation, and monitoring. Each unit within the layer was designed to operate within a containerized environment, allowing us to orchestrate the whole test environment efficiently. Additionally, the integration of Docker containers ensured isolated and consistent deployment of components of the network.

4.1.1. Core Hyperledger Fabric Network Layer

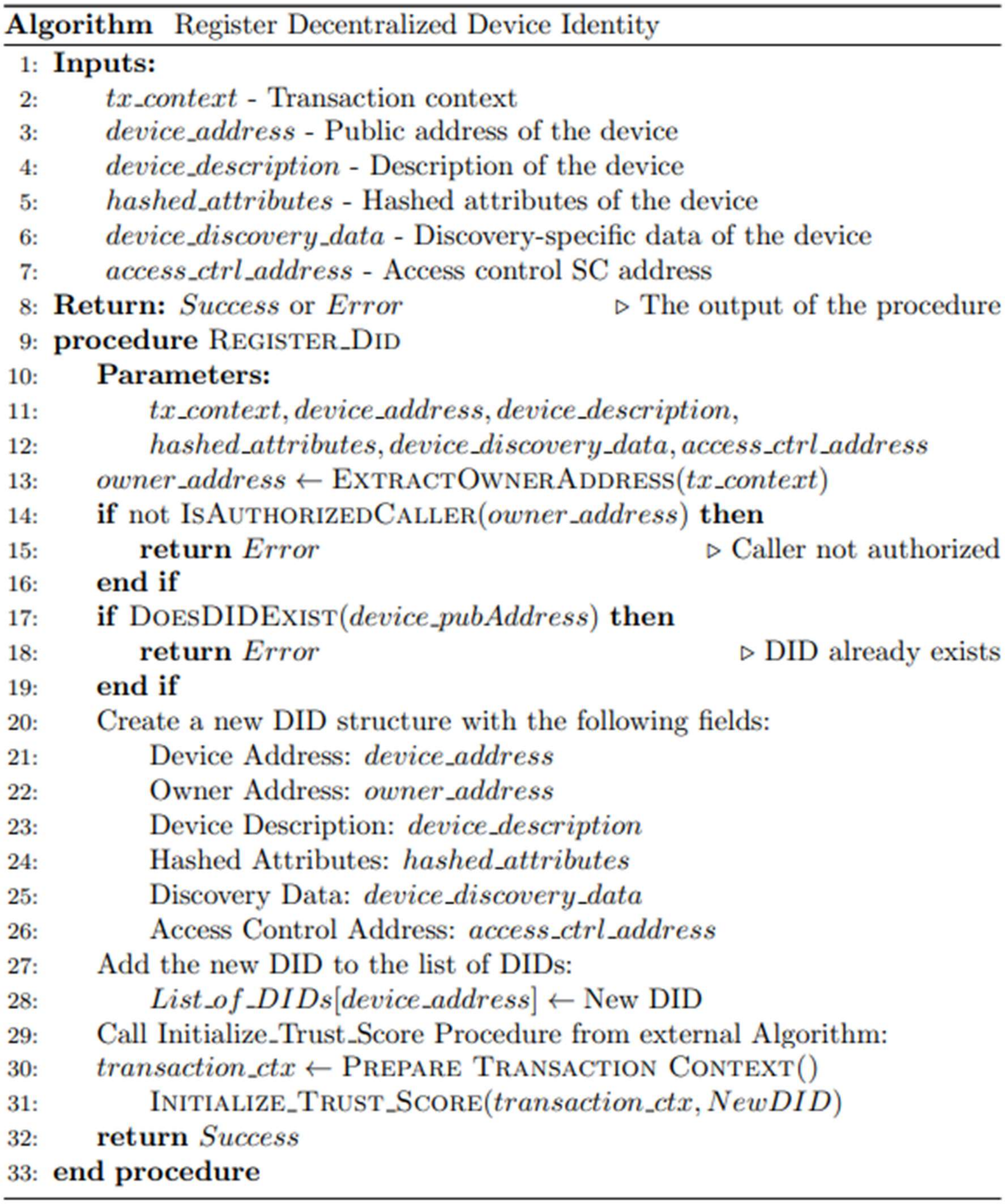
The Core Hyperledger Fabric Network Layer serves as the backbone of the proposed system, consisting of a series of interconnected nodes and services that orchestrate the blockchain's operations. Hyperledger Fabric's architecture is inherently modular, allowing it to be finely tuned to specific requirements such as those needed for IoT devices. This layer comprises several components, including peer nodes, orderers, MSPs, CAs, and chaincodes. The network architecture was designed with two peer organizations, each acting as a separate administrative domain within the blockchain. This dual setup enables the simulation of a decentralized ecosystem where multiple stakeholders manage their IoT devices independently. Each organization was equipped with its own set of peer nodes, enabling the execution and storage of transactions related to its domain. The peer nodes within each organization endorsed transactions, maintained the ledger, and ran smart contracts autonomously. Central to the network's architecture was the ordering service, which utilized the Raft consensus protocol. The Raft-based orderer ensured a high-performance and fault-tolerant ordering mechanism for transactions. The ordering service was singular across the network, aggregating transactions from both organizations, sequencing them consistently, and distributing blocks to all peers, thus maintaining a cohesive and synchronized ledger.

Each organization, including the orderer, had its own CA, which was responsible for issuing and revoking digital certificates that authenticate the identities of nodes and users, forming the basis of a trust structure within the network. These certificates, essential for participating nodes and applications in the network, were validated through the MSP. The MSP delineated the rights and privileges of the participants, enforcing access control policies at an organizational level. The architectural integration of the components of the layer facilitated a robust environment for secure transactions among IoT devices. The peer organizations operated independently yet cohesively, with the single orderer providing a centralized point for transaction ordering without compromising the decentralized nature of the network. This setup reinforced security through division and isolation and optimized network performance through the Raft consensus, meeting the critical demands of IoT environments for rapid IoT IAM and trust management.

Within the Core HLF Network Layer, each organization has its own instance of chaincodes to enable the functions of the proposed framework. These chaincodes comprise the Decentralized Identity SC, the Trust Evaluation SC, the Device-1 Access Control SC, the Device-2 Access Control SC, and the Asset Transfer SC. By maintaining separate instances for each SC, autonomy for each organization was enabled, allowing them to manage and update their contracts as per their operational requirements while still participating in a shared ledger system.

4.1.1.1. The Decentralized Identity Smart Contract

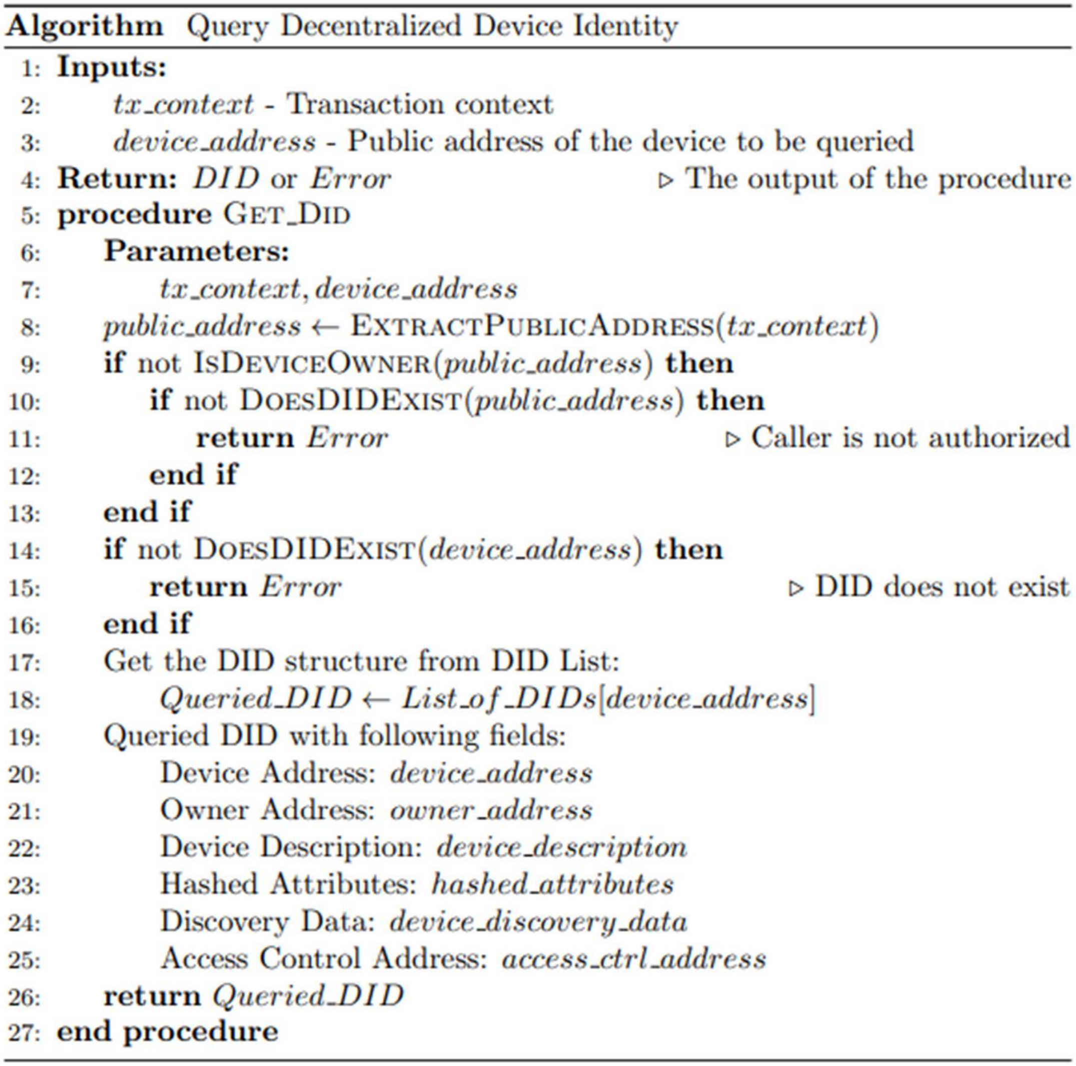
The Decentralized Identity SC was implemented in Go and consisted of various methods. The Register Decentralized Device Identity Method is considered the entry point for adding new identities to the framework. It ensures that no duplicate entries are created and associates the device with the identity of the transaction submitter, effectively establishing device ownership. The pseudo code of this method can be seen in Figure 4.2.



```
Algorithm Register Decentralized Device Identity
Inputs:
    tx_context - Transaction context
    device_address - Public address of the device
    device_description - Description of the device
    hashed_attributes - Hashed attributes of the device
    device_discovery_data - Discovery-specific data of the device
    access_ctrl_address - Access control SC address
Return: Success or Error                    ▷ The output of the procedure
procedure REGISTER_DID
    Parameters:
        tx_context, device_address, device_description,
        hashed_attributes, device_discovery_data, access_ctrl_address
    owner_address ← EXTRACTOWNERADDRESS(tx_context)
    if not ISAUTHORIZEDCALLER(owner_address) then
        return Error                        ▷ Caller not authorized
    end if
    if DOESDIDEXIST(device_pubAddress) then
        return Error                        ▷ DID already exists
    end if
    Create a new DID structure with the following fields:
        Device Address: device_address
        Owner Address: owner_address
        Device Description: device_description
        Hashed Attributes: hashed_attributes
        Discovery Data: device_discovery_data
        Access Control Address: access_ctrl_address
    Add the new DID to the list of DIDs:
        List_of_DIDs[device_address] ← New DID
    Call Initialize_Trust_Score Procedure from external Algorithm:
    transaction_ctx ← PREPARE TRANSACTION CONTEXT()
        INITIALIZE_TRUST_SCORE(transaction_ctx, NewDID)
    return Success
end procedure
```

Figure 4.2 Pseudo Code for Identity Registration

Query Decentralized Device Identity Method was used to query an identity based on the public blockchain address of the identity owner. It serves as a verification method between devices and is used in mutual authentication process. The pseudo code for this method can be seen in Figure 4.3.



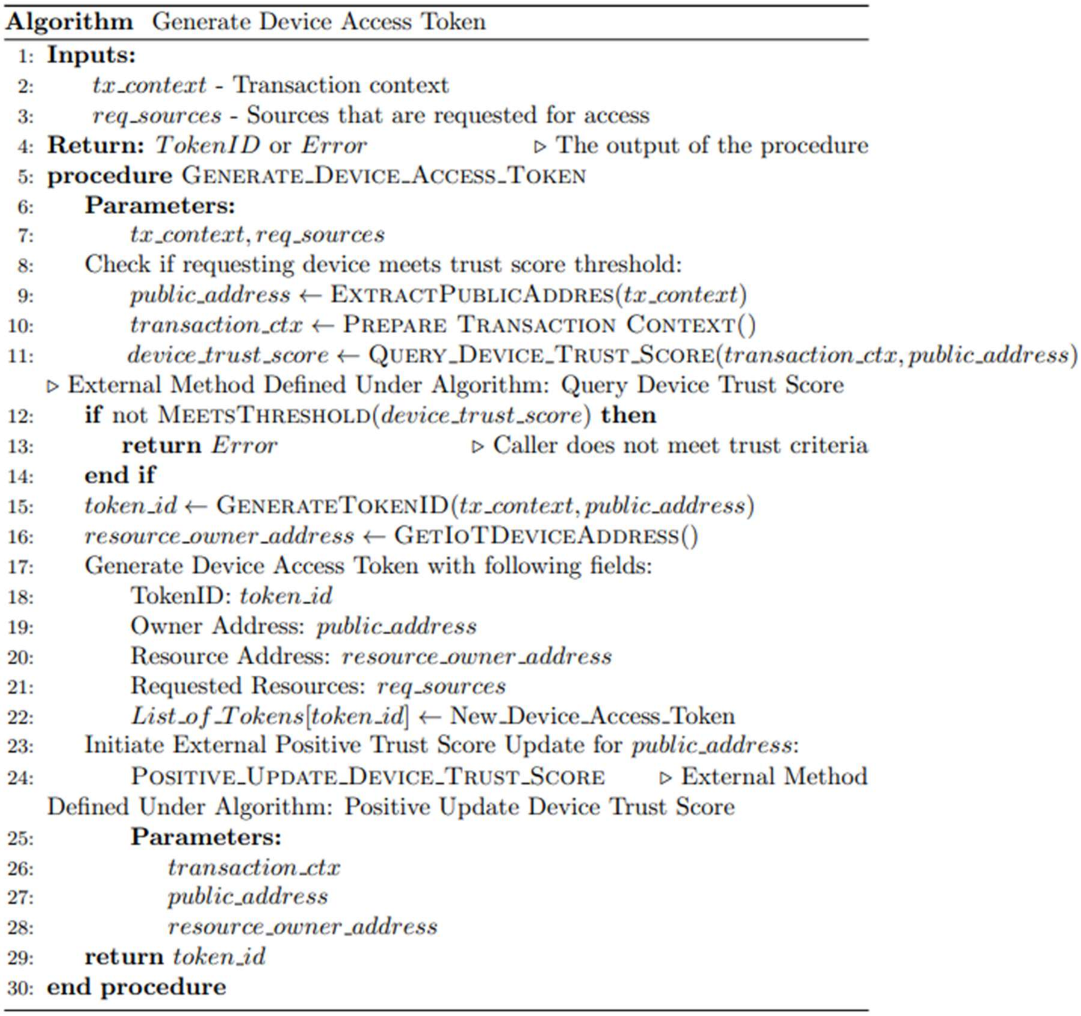
```
Algorithm Query Decentralized Device Identity
Inputs:
    tx_context - Transaction context
    device_address - Public address of the device to be queried
Return: DID or Error                                  ▷ The output of the procedure
procedure GET_DID
    Parameters:
        tx_context, device_address
    public_address ← EXTRACTPUBLICADDRESS(tx_context)
    if not ISDEVICEOWNER(public_address) then
        if not DOESDIDEXIST(public_address) then
            return Error                              ▷ Caller is not authorized
        end if
    end if
    if not DOESDIDEXIST(device_address) then
        return Error                                  ▷ DID does not exist
    end if
    Get the DID structure from DID List:
        Queried_DID ← List_of_DIDs[device_address]
    Queried DID with following fields:
        Device Address: device_address
        Owner Address: owner_address
        Device Description: device_description
        Hashed Attributes: hashed_attributes
        Discovery Data: device_discovery_data
        Access Control Address: access_ctrl_address
    return Queried_DID
end procedure
```

Figure 4.3 Pseudo Code for Identity Query

The Decentralized Identity SC also contains Change of Ownership and Decommission Device methods which allow the change of ownership between devices and end-of-lifecycle management of devices. Since these methods are straightforward and have little effect on the performance and security aspects of the framework, they were not implemented in the current state of the implementation.

4.1.1.2. The Device Access Control Smart Contract

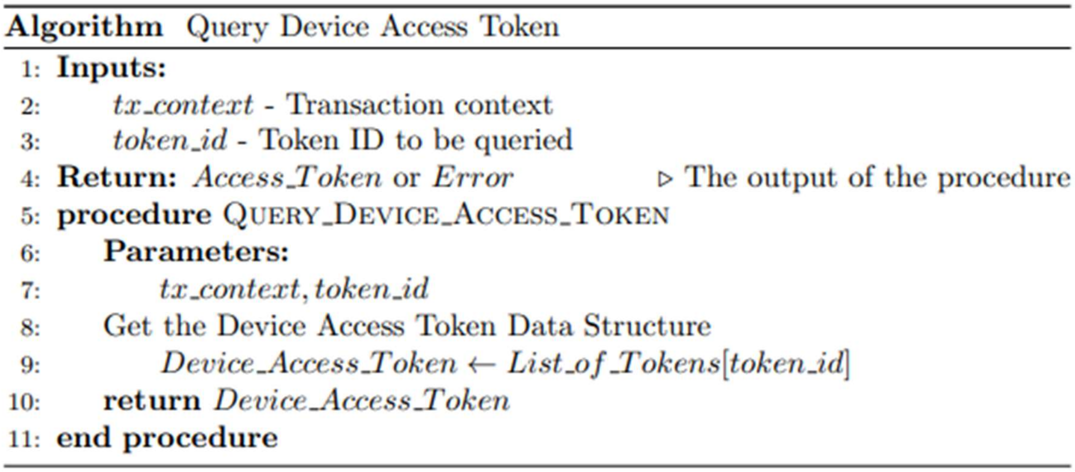
The Device Access Control SC is responsible for managing access control by issuing and verifying access tokens for devices. It is written in Go and represents an essential security feature within the proposed system by enforcing trust-based resource access. The main functionality of the contract is the generation of device access tokens. This process begins by querying the trust score of the requesting device from the Trust Evaluation SC. If the device's trust score meets the predefined threshold, a unique token ID is generated, and its respective token is minted. Pseudo code of this process is given in Figure 4.4.



```
Algorithm Generate Device Access Token
Inputs:
    tx_context - Transaction context
    req_sources - Sources that are requested for access
Return: TokenID or Error                    ▷ The output of the procedure
procedure GENERATE_DEVICE_ACCESS_TOKEN
    Parameters:
        tx_context, req_sources
    Check if requesting device meets trust score threshold:
        public_address ← EXTRACTPUBLICADDRES(tx_context)
        transaction_ctx ← PREPARE TRANSACTION CONTEXT()
        device_trust_score ← QUERY_DEVICE_TRUST_SCORE(transaction_ctx, public_address)
    ▷ External Method Defined Under Algorithm: Query Device Trust Score
    if not MEETSTHRESHOLD(device_trust_score) then
        return Error                        ▷ Caller does not meet trust criteria
    end if
    token_id ← GENERATETOKENID(tx_context, public_address)
    resource_owner_address ← GETIOTDEVICEADDRESS()
    Generate Device Access Token with following fields:
        TokenID: token_id
        Owner Address: public_address
        Resource Address: resource_owner_address
        Requested Resources: req_sources
        List_of_Tokens[token_id] ← New_Device_Access_Token
    Initiate External Positive Trust Score Update for public_address:
        POSITIVE_UPDATE_DEVICE_TRUST_SCORE       ▷ External Method
    Defined Under Algorithm: Positive Update Device Trust Score
        Parameters:
            transaction_ctx
            public_address
            resource_owner_address
    return token_id
end procedure
```

Figure 4.4 Pseudo Code for Access Token Generation

The Device Access Control SC also allows querying of specific access tokens using various mechanisms of the HLF platform, using the unique token ID. This mechanism ensures that access tokens can be efficiently retrieved and validated, enabling secure and streamlined access to resources. The pseudo code of this process is given in Figure 4.5.

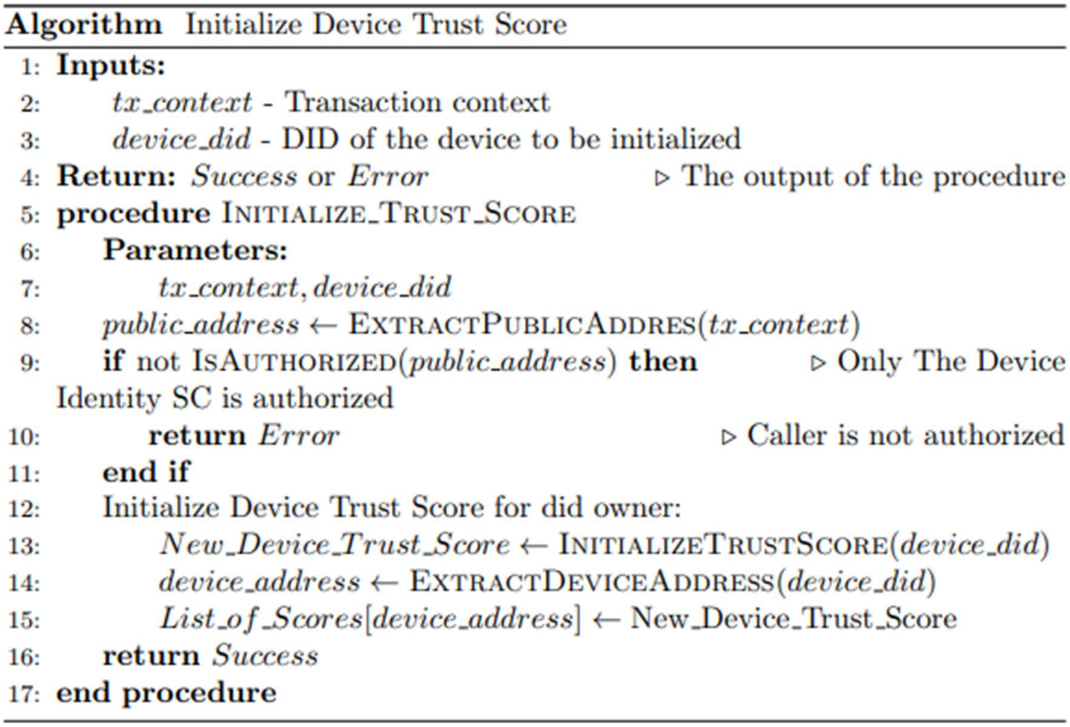


```
Algorithm Query Device Access Token
Inputs:
    $tx\_context$ - Transaction context
    $token\_id$ - Token ID to be queried
Return: $Access\_Token$ or $Error$                ▷ The output of the procedure
procedure QUERY_DEVICE_ACCESS_TOKEN
    Parameters:
        $tx\_context, token\_id$
    Get the Device Access Token Data Structure
        $Device\_Access\_Token \leftarrow List\_of\_Tokens[token\_id]$
    return $Device\_Access\_Token$
end procedure
```

Figure 4.5 Pseudo Code for Access Token Query

4.1.1.3. The Trust Evaluation Smart Contract

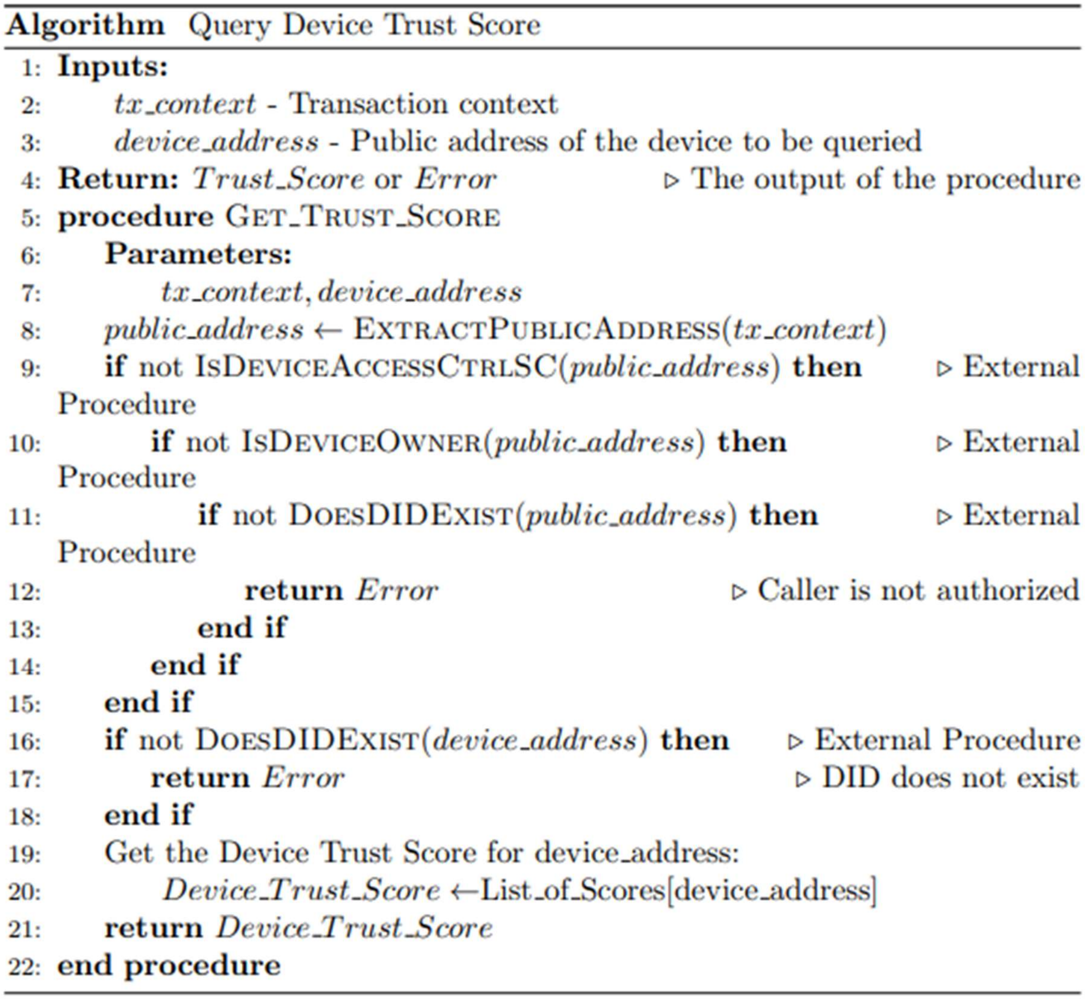
The Trust Evaluation SC is designed to dynamically assess and manage the trust scores of IoT devices within the proposed framework. This Go-based contract operates within the HLF network, performing critical trust-related operations. A key method of the contract is the Initialize Device Trust Score, which sets the ITS for a device upon its registration in the system. This initial score is a crucial starting point for the device's interactions within the IoT ecosystem and is based on predefined criteria that could be adapted to specific IoT applications. The pseudo code for this process is shown in Figure 4.6.



```
Algorithm Initialize Device Trust Score
Inputs:
    tx_context - Transaction context
    device_did - DID of the device to be initialized
Return: Success or Error                    ▷ The output of the procedure
procedure INITIALIZE_TRUST_SCORE
    Parameters:
        tx_context, device_did
    public_address ← EXTRACTPUBLICADDRES(tx_context)
    if not ISAUTHORIZED(public_address) then          ▷ Only The Device Identity SC is authorized
        return Error                        ▷ Caller is not authorized
    end if
    Initialize Device Trust Score for did owner:
        New_Device_Trust_Score ← INITIALIZETRUSTSCORE(device_did)
        device_address ← EXTRACTDEVICEADDRESS(device_did)
        List_of_Scores[device_address] ← New_Device_Trust_Score
    return Success
end procedure
```

Figure 4.6 Pseudo Code for Trust Score Initialization

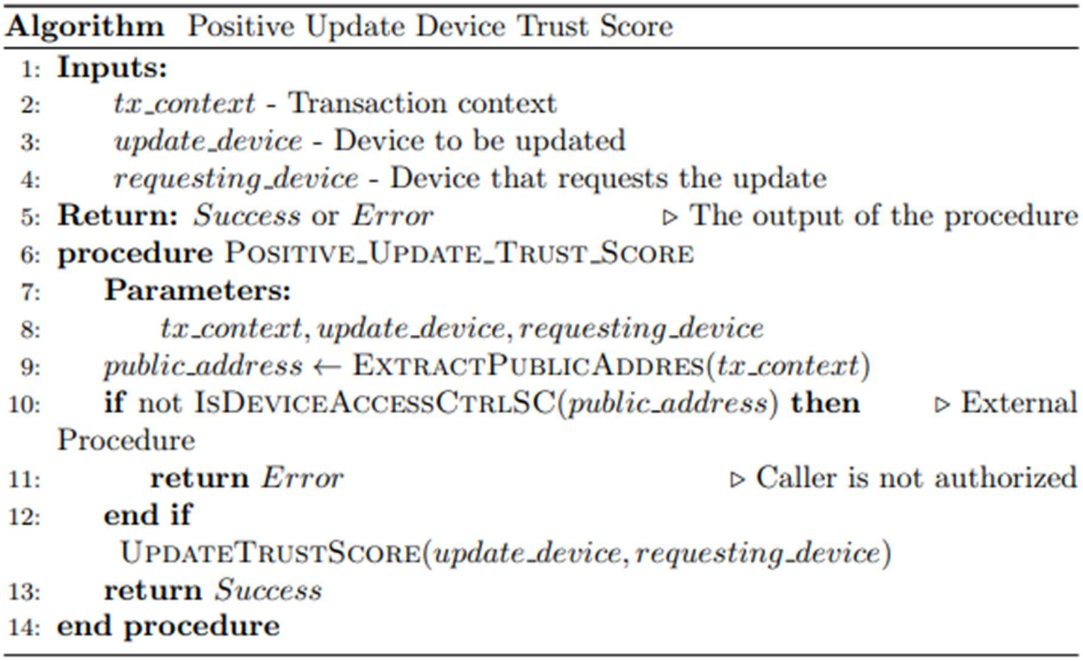
Additionally, the contract allows the retrieval of current trust score of a given device through the Query Device Trust Score method. This feature is crucial for other components of the framework to make informed decisions based on the trustworthiness of different devices. In order to call this method, the caller must either be a Device Access Control SC or a device owner or must have a valid identity registered within the Decentralized Identity SC. The pseudo code for the Query Device Trust Score method is shown in Figure 4.7.



```
Algorithm Query Device Trust Score
Inputs:
    tx_context - Transaction context
    device_address - Public address of the device to be queried
Return: Trust_Score or Error                    ▷ The output of the procedure
procedure GET_TRUST_SCORE
  Parameters:
      tx_context, device_address
  public_address ← EXTRACTPUBLICADDRESS(tx_context)
  if not ISDEVICEACCESSCTRLSC(public_address) then    ▷ External Procedure
      if not ISDEVICEOWNER(public_address) then       ▷ External Procedure
          if not DOESDIDEXIST(public_address) then    ▷ External Procedure
              return Error                       ▷ Caller is not authorized
          end if
      end if
  end if
  if not DOESDIDEXIST(device_address) then     ▷ External Procedure
      return Error                              ▷ DID does not exist
  end if
  Get the Device Trust Score for device_address:
      Device_Trust_Score ←List_of_Scores[device_address]
  return Device_Trust_Score
end procedure
```

Figure 4.7 Pseudo Code for Query Trust Score

The contract also introduces a mechanism for updating trust scores positively through the Positive Update Trust Score method. It receives a request from a device where, based on the request, it updates the trust score of a device positively. The pseudo code of the process is given in Figure 4.8.



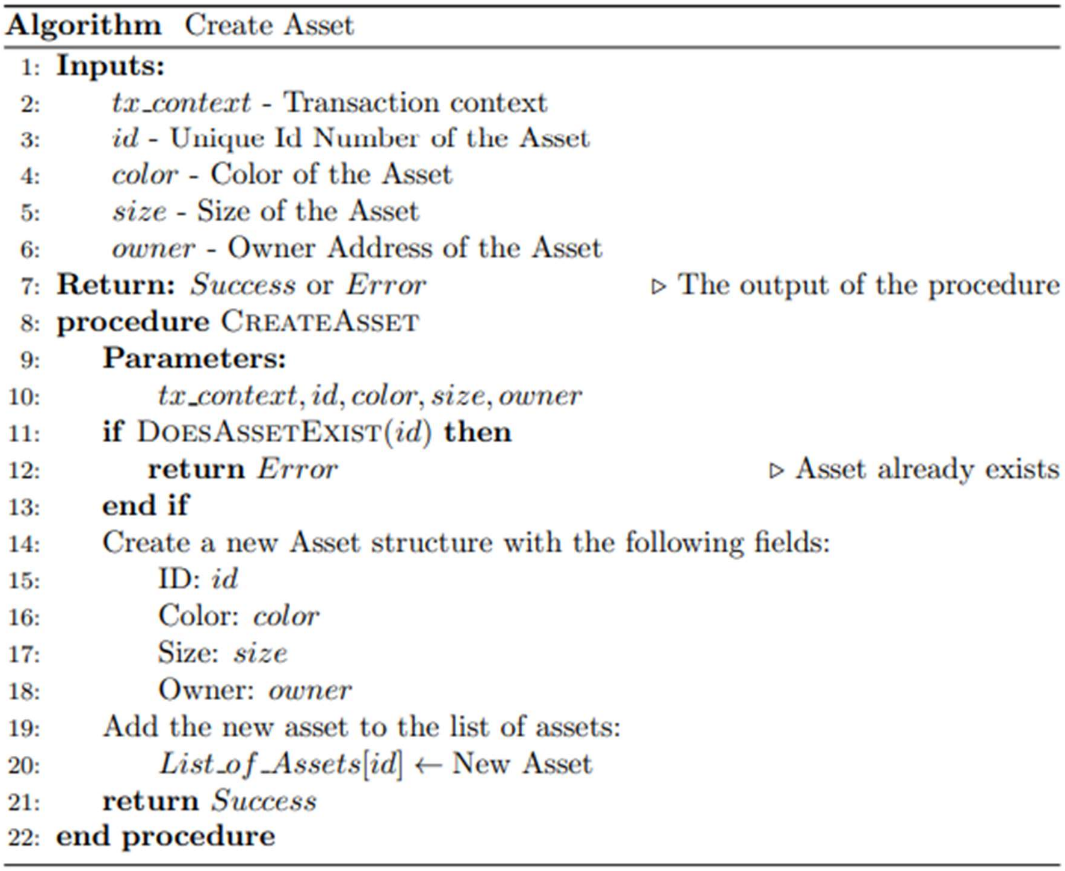
```
Algorithm Positive Update Device Trust Score
Inputs:
    tx_context - Transaction context
    update_device - Device to be updated
    requesting_device - Device that requests the update
Return: Success or Error                          ▷ The output of the procedure
procedure POSITIVE_UPDATE_TRUST_SCORE
    Parameters:
        tx_context, update_device, requesting_device
    public_address ← EXTRACTPUBLICADDRES(tx_context)
    if not ISDEVICEACCESSCTRLSC(public_address) then        ▷ External
Procedure
        return Error                              ▷ Caller is not authorized
    end if
     UPDATETRUSTSCORE(update_device, requesting_device)
    return Success
end procedure
```

Figure 4.8 Pseudo Code for Positive Update Trust Score

In the development of the Trust Evaluation SC, a deliberate decision was made to focus on implementing positive trust score updates, forgoing the negative trust score update method. This decision was made to streamline the initial version of the contract, simplifying its operational framework to ensure robustness and reliability in its core functionalities. By prioritizing the positive trust score updates, the contract effectively captures the dynamics of increasing trust based on positive device interactions and performance within the IoT ecosystem. In the Future Work section of the thesis, the importance of the negative trust score update method is thoroughly discussed. This method is essential for a fully rounded trust management system, as it would enable the network to respond dynamically to potentially harmful or untrustworthy behaviors by IoT devices.

4.1.1.4. The Asset Transfer Smart Contract

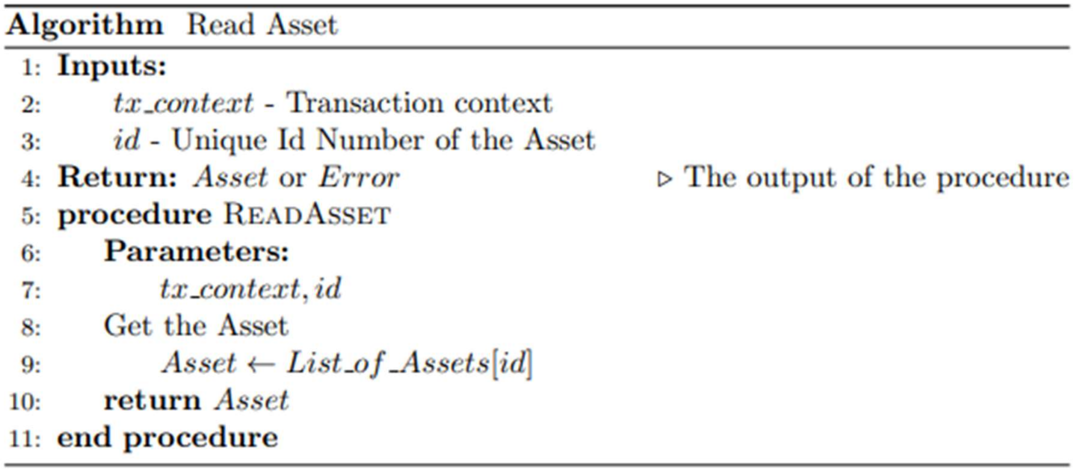
In order to establish a benchmark for evaluating the performance of the proposed framework for IoT, Asset Transfer SC, a standard, basic reference implementation from the Hyperledger Fabric Samples repository, was deployed in the testing environment [86]. This contract, designed for basic asset management operations, serves as a reliable point of comparison to measure the efficiency and scalability of our custom-developed smart contracts. Asset Transfer SC provides fundamental functionalities such as creating assets and reading the asset details. The simplicity and standardization of this contract make it an ideal candidate for performance benchmarking. The main functionality of the Asset Transfer SC is the Create Asset method, as given in Figure 4.9.



```
Algorithm Create Asset
Inputs:
    tx_context - Transaction context
    id - Unique Id Number of the Asset
    color - Color of the Asset
    size - Size of the Asset
    owner - Owner Address of the Asset
Return: Success or Error                    ▷ The output of the procedure
procedure CREATEASSET
    Parameters:
        tx_context, id, color, size, owner
    if DOESASSETEXIST(id) then
        return Error                        ▷ Asset already exists
    end if
    Create a new Asset structure with the following fields:
        ID: id
        Color: color
        Size: size
        Owner: owner
    Add the new asset to the list of assets:
        List_of_Assets[id] ← New Asset
    return Success
end procedure
```

Figure 4.9 Pseudo Code for Create Asset

The pseudo code for the Read Asset method is also given in Figure 4.10. These pseudo codes are crucial to understand to benchmark them against the SCs of the proposed framework correctly.



```
Algorithm Read Asset
Inputs:
        $tx\_context$ - Transaction context
        $id$ - Unique Id Number of the Asset
Return: $Asset$ or $Error$                        ▷ The output of the procedure
procedure READASSET
    Parameters:
        $tx\_context, id$
    Get the Asset
        $Asset \leftarrow List\_of\_Assets[id]$
    return $Asset$
end procedure
```

Figure 4.10 Pseudo Code for Read Asset

4.1.2. Client Apps and Performance Test Layer

The performance testing of the proposed IoT IAM and trust evaluation framework was conducted using Hyperledger Caliper, a blockchain benchmarking tool, to evaluate the performance of individual smart contract methods specifically. This approach allowed the opportunity to gain detailed insights into the efficiency of each transaction type, focusing on transaction throughput and latency, which are critical metrics for IoT operations. For a more practical and representative test environment, a Linux workstation running Ubuntu 20.04 was employed. The setup enabled the emulation of the conditions of a real-world deployment, ensuring that performance metrics were as realistic as possible. In the testing architecture, two separate Docker containers were deployed, each acting as a simulated IoT device (device-1 and device-2). These containers interacted with the framework through the HLF Access SDK [87]. An embedded Credential Wallet within each container was used for managing identity credentials securely, a crucial aspect of IoT device interaction in a blockchain network. This setup was used in the testing of the mutual authentication scheme. Moreover, hardware resources available to each IoT container were limited to mirror the capabilities of actual IoT devices, which often operate with constrained computational power and memory. The combination of Hyperledger Caliper for smart contract method testing and the constrained resource environment of the Docker containers provided us with a comprehensive view of the system's performance.

4.1.3. Monitoring Layer

The monitoring layer can be considered to play a critical role in the ongoing health assessment and performance valuation of the blockchain-based proposed framework. Tools such as Prometheus [88] and Grafana [89], coupled with Hyperledger Caliper, were used to achieve comprehensive monitoring. Prometheus application was configured to collect a wide array of metrics from the network. This included detailed resource usage data, critical for understanding the system's performance under various load conditions. By monitoring metrics such as CPU usage, memory consumption, and network I/O, we gained insights into how the system responds to different operational demands. To complement Prometheus' data collection capabilities, Grafana provided a user-friendly and intuitive interface for data visualization. Grafana enabled us to create dynamic dashboards that displayed real-time metrics, offering an immediate visual representation of the system's performance and health. Integrating Prometheus, Grafana, and Hyperledger Caliper's visualization tools created a robust monitoring framework. This framework supported the efficient observation of system performance and resource utilization and facilitated a deeper understanding of how different components interacted within the proposed framework.

4.2. Results and Discussion on Performance

This chapter conducts a critical assessment of the performance results obtained from implementing the proposed framework. This evaluation is considered pivotal in demonstrating not only the feasibility but also the efficiency and robustness of the proposed system. Throughout the analysis, it was aimed to bridge the gap between conceptualization and practical application. Performance evaluation encompasses a comprehensive analysis of various key metrics, including transaction throughput, resource utilization, latency, and system scalability. Each working scheme (detailed under Chapter 3) of the framework was tested against performance. For instance, results for the device registration scheme can be seen in Table 4.1.

Table 4.1 Performance Results for Device Registration

Tested Scenario	Success	Fail	Send Rate (TPS)	Max Latency (s)	Min Latency (s)	Average Latency (s)	Throughput (TPS)

Device Registration	30002	6	1000.0	11.34	0.05	7.21	730.7

It can be seen from the results that out of total device registration attempts, 30002 were successful, while only six failed. A 99.98% success rate suggests that the system is capable of handling registration requests with a high degree of reliability. Additionally, the send rate of 1000 TPS indicates that the implemented system can generate a high volume of transactions. The latency results show a maximum latency of 11.34 seconds and a minimum of 0.05 seconds, with an average latency of 7.21 seconds. While the average and maximum latencies are on the higher side, it's not unusual for blockchain systems, especially when handling a high volume of transactions. The throughput of 730.7 TPS is slightly lower than the send rate, which is expected due to network and processing overheads.

Overall performance results for Create Asset, Read Asset, Device Registration, Device Authorization, and Device Discovery schemes can be seen in Table 4.2. Additionally, resource usage metrics for each scheme can be viewed in Table 4.3.

Table 4.2 Overall Performance Results

Tested Scenario	Create Asset	Read Asset	Device Registration	Device Authorization	Device Discovery
Sequence Diagram	-	-	Figure 3.2.	Figure 3.4.	Figure 3.3.
Algorithms Used	Figure 4.9.	Figure 4.10.	Figure 4.2.	Figure 4.4.	Figure 4.3. Figure 4.7.
Success Transaction Count	30002	218261	30002	29999	108692
Failed Transaction Count	8	0	6	11	0

Send Rate (TPS)	999.9	7222.6	1000.0	995.4	7218.2
Max Latency (s)	5.21	0.53	11.34	7.54	0.13
Min Latency (s)	0.03	0.00	0.05	0.03	0.00
Average Latency (s)	2.89	0.10	7.21	4.39	0.03
Throughput (TPS)	861.0	7222.5	730.7	806.5	7217.3

Table 4.3 Resource Usage Results

Tested Scenario	Create Asset	Read Asset	Device Registration	Device Authorization	Device Discovery
CPU Usage - Host	%62	%100	%80	%71	%100
CPU Usage - Peer0: Organization-1	%231.46	%628.43	%366.07	%317.21	%692.08
Memory Usage - Peer0: Organization-1	478.7 MB	963.6 MB	1.056 GB	1.277 GB	1.623 GB
CPU Usage - Peer0: Organization-2	%183.49	%1.60	%280.77	%232.84	%1.55

Memory Usage - Peer0: Organization-2	358.7 MB	489.6 MB	489.1 MB	551.3 MB	581.5 MB
CPU Usage - Orderer	%84.31	%0.39	%82.63	%78.26	%0.28
Memory Usage - Orderer	184.8 MB	381.1 MB	426.7 MB	607.1 MB	723.6 MB

As we dissect the performance results of each operational scheme within our proposed framework, it is crucial to acknowledge the capabilities of the host machine that facilitated these tests. The host machine was equipped with an AMD Ryzen 9 7950X 16-core Processor, operating at 4501 MHz. Additionally host environment was equipped with 32 GB of installed RAM. A notable observation from the results is the stark contrast between read and write operations in TPS. Read operations like those seen in the Asset Read scenario and Device Discovery exhibited a significantly higher throughput than write operations like Asset Creation and Device Registration. The inherent difference between read and write operations in blockchain systems can account for this dissimilarity. Write operations are generally more resource intensive as they involve executing transaction logic, reaching consensus across the network, and committing new data to the ledger. Conversely, read operations are typically less demanding as they only require retrieving data from the ledger without the need for consensus or ledger updates. In our case, the Asset Read and Device Discovery operations demonstrated higher throughput than other schemes because reads are not subject to the consensus mechanism and only involve local data retrieval, which can be serviced rapidly compared to ledger writes. An observation from our performance testing is the upper limit of 7222.6 TPS for read requests, which were unable to surpass with our current host machine setup. This limitation was evident despite the substantial processing power of the AMD Ryzen 9 7950X 16-core processor. The CPU and memory utilization metrics during read request tests approached the maximum capacity, indicating that we reached the performance ceiling of our testing environment rather than that of the proposed framework itself. This observation suggests that the tested throughput for read operations may not reflect the full potential of the proposed framework. Instead, it represents the limit imposed by our specific hardware configuration. Given that the resource usage during these read operations was at its peak, with close to %100 CPU usage recorded, the framework could achieve higher throughputs on a more capable or specialized hardware setup. It is also worth considering the theoretical read limits of HLF networks. While HLF does not have a hard-coded limit on the read TPS, the practical limit is often dictated by factors such as network architecture, consensus mechanism efficiency, the complexity of chaincode operations, and the hardware specifications of the nodes involved. The read TPS can be considerably higher in enterprise-grade deployments, where resources are scaled to match the demand.

The HLF documentation and community benchmarks often cite much higher theoretical limits. Still, these are under ideal conditions with highly optimized chaincodes, network configurations, and possibly more powerful hardware than what was available in our test setup. Hence, it's reasonable to infer that our proposed framework's actual limits on read TPS could be higher when deployed in an optimized production environment.

In analyzing the performance results for write operations such as Create Asset, Device Registration, and Device Authorization, we observe that these transactions engage with most of the network's components, including both peer organizations and the orderer. The resource utilization table reflects the involvement of these parties, as evidenced by the increase in CPU and memory usage across the network during these operations. Write operations in HLF are inherently more resource-intensive than read operations. This is because they require endorsement from peers, ordering of transactions, and commitment to the ledger, processes that collectively involve all network parties. Such operations invoke a consensus mechanism, which in our case includes the orderer services, to ensure that transactions are validated and consistently recorded across the distributed ledger. Our performance metrics for write operations demonstrate that while the throughput for these transactions is lower than for read operations, the success rates remain high, which is indicative of a stable and reliable network. Device Registration and Device Authorization processes, while unique to our framework, show throughputs of 730.7 TPS and 806.5 TPS, respectively, suggesting that our system performs relatively close to a more basic Asset Transfer SC. Asset Transfer SC related schemes are included as benchmarks to provide a standard comparison against typical HLF operations. The fact that our specialized framework is close to these benchmarks strongly indicates its efficiency.

Results for the last scheme of the proposed framework, mutual authentication, can be seen in Table 4.3.

Table 4.4 Performance Metrics for Mutual Authentication

Device Name	Device-1	Device-2
Sequence Diagram	Figure 3.5.	Figure 3.5.
Algorithms Used	Figure 4.5.	Figure 4.5.
Peak Throughput (TPS)	372.9	372.9
Total CPU	1 Core	1 Core
Total Memory	1 GB	1 GB
Peak CPU Usage	%100	%100
Peak Memory Usage	120 - 130 MB	120 - 130 MB

Throughput around 350-400 TPS is compatible with our previous findings, considering the context of mutual authentication, which is typically more complex due to the cryptographic processes involved. The total CPU and memory allocation for each device was constrained to 1 Core and 1 GB, respectively, to mimic real IoT devices, which are often limited in resources. The average CPU usage reaching %100 for both devices indicates that the mutual authentication process is CPU-intensive. This is expected due to the cryptographical computations required to establish trust between the devices. Average memory usage is relatively low, which suggests that the mutual authentication scheme is not as memory intensive. This low memory footprint benefits IoT environments where devices may have limited resources. Since mutual authentication involves direct interaction between Device-1 and Device-2, using Jmeter [90] to trigger the authentication process via their APIs, instead of using Hyperledger Caliper, is considered a better solution. Jmeter is a versatile tool that can effectively simulate API requests and measure the performance of these interactions. The testing approach, involving the triggering of connection and authentication sequences between the devices, simulates a realistic scenario where devices frequently need to establish secure connections in an IoT network.

4.2.1. Contextualizing for IoT Environments

The mutual authentication and token generation schemes demonstrate throughput levels that are well within the acceptable range for most single-device operations in an IoT context. In practical scenarios, it is unlikely that individual IoT devices would need to generate requests at a higher rate than what our system can handle. However, when we consider the device registration and trust score-related functions, the achieved throughput may not suffice for a full-scale IoT network where potentially thousands of devices might attempt to register or update trust scores concurrently. While promising, the maximum throughput observed under our test conditions suggests that further optimization might need to accommodate the vast number of transactions that a complete IoT ecosystem would demand. An enterprise-level HLF solution may offer higher throughputs, leveraging more powerful hardware and optimized network configurations. Nonetheless, it is important to recognize that blockchain technology, by its nature, imposes certain constraints on transaction processing due to the requirements of consensus protocols and data immutability, which can affect throughput. Blockchain technology is still considered to be in its infancy, and ongoing research is aimed at enhancing its performance. Innovations in consensus algorithms, network sharding, and off-chain processing are among the many areas being explored to increase transaction processing speeds. As the field of blockchain matures, and these research efforts bear fruit, we can anticipate significant improvements in throughput and efficiency. These advancements will undoubtedly make frameworks like ours more viable for real-world applications.

4.3. Discussion on Security

In this sub-section, we discuss the security aspects of our “Blockchain-Based Decentralized Identity and Trust Evaluation Framework” for IoT, mainly focusing on its resilience against various cyber-attacks. The decentralized nature of blockchain provides a foundation for security; however, no system is fully protected against threats. By examining potential attack vectors such as Man-in-the-Middle (MitM), Sybil, Endpoint Compromise, and blockchain-based attacks, we aim to assess the resilience of our framework and identify areas for further strengthening.

4.3.1. Man in the Middle Attacks

A MitM attack involves an attacker secretly relaying and possibly altering the communication between two parties [91]. In the context of our proposed framework, this attack can typically target device mutual authentication and access token generation schemes. For the device authorization and access control function, an attacker may try to intervene between a device and the Device Specific Access Control SC and pretend to be the device that makes the requests. However, for this, the attacker would need to have acquired the blockchain credentials of the requesting device, as SC would look for a verified transaction from the requesting identity. Also, SC would look for a valid trust score and, hence, a valid decentralized device identity registered on the Decentralized Identity SC. Without acquiring the private credentials of a valid entity, an attacker cannot perform a MitM attack on the device authorization, as it would be forced to prove its identity at some point to continue the communication. Upon hijacking a valid device inside the framework, an attacker can perform a MitM attack if other devices do not detect it. However, this is considered a different class of attack, and it depends on the security features of the specific device. If other devices detect the hijacked device, these devices may report the incident through the trust evaluation mechanism by sending appropriate responses to the framework. Based on these incidents, the hijacked device may become decommissioned.

In the case of device mutual authentication, where two devices try to verify their identity and create a shared session key, an attacker may try to intervene and assume the identity of one device. The attack can gain the access token explicitly generated for a device while listening to the communication and then use this token to authenticate itself to the other device. However, the access tokens, by their nature, are bound to a device's identity, hence their secure private-public key pair. Although the attacker would get the access token, it would need to possess the private key respective to that token to verify its identity to the receiving device. Without having the private key of a device, a MitM attacker cannot assume the identity of a device and, hence, cannot complete the mutual authentication scheme of the proposed framework.

4.3.2. Sybil Attacks

Sybil attacks involve creating numerous fake identities to subvert the network's reputation system [92]. In a framework like ours, this typically means generating many fake nodes or devices to gain influence across the system. The attacker may target several working schemes in our framework: the device registration process, the trust evaluation method, and the overall control of the blockchain network. The attacker would first need the authority to register devices within the framework for the device registration process. Without the respective access, an attacker wouldn't be allowed to register device identities within the permissioned blockchain framework and typically would be required to become a verified IoT manufacturer, verified by the other network participants. An attacker may try to hijack devices within the framework and use these devices as nodes to perform a Sybil attack. However, since a global-scale IoT ecosystem is considered within the framework, an attacker would be required to acquire numerous devices within the network to assume control of the system. This would require enormous time and resources, which may be considered unfeasible and unrealistic on a fully decentralized extensive network. The attacker may try to disrupt the trust evaluation process by gaining control over reputable devices. This attack is our framework's most likely cyber threat and must be dealt with accordingly. In order to have a proper response to such attacks, the framework must undergo extensive cybersecurity testing and identify further weaknesses. A hijacking of a highly reputable device within the framework may indeed damage the framework. Response to such an attack within the proposed framework is identified as a research area for further strengthening.

4.3.3. Endpoint Compromise

Endpoint compromise refers to unauthorized access to a device, allowing an attacker to manipulate a device and extract sensitive information from the system via this device. Endpoint compromise refers to unauthorized access to a device, allowing an attacker to manipulate a device and extract sensitive information from the system via this device. While the blockchain layer can be secure, IoT devices are often the most vulnerable. An attacker may hijack an IoT device and use this device to disrupt the operation of the system. The proposed framework's decentralized nature offers several barriers to mitigate the effect of hacked devices on the system. First, on a large-scale network with millions of devices, an attacker may need to possess many devices to launch a significant cyber-attack on the framework. This attack can be considered unfeasible since the attacker cannot just gain control over a singular node and compromise the whole network due to the decentralized nature of the system. Additionally, there are mechanisms like device decommissioning, negative feedback, and audit trailing within the framework. As soon as a device is identified as hacked, that device is immediately removed from the framework, and all transactions issued by that device can be traced backward to analyze the attack and generate the appropriate response. In short, an attacker would need to compromise numerous endpoints to make a difference within the framework.

4.3.4. Blockchain-Based Attacks

Attacks specific to blockchain technology include %51 attacks, where an attacker gains control of most of the network's mining hash rate, and smart contract vulnerabilities, where flaws in the contract code can be exploited. Given that our framework operates on a permissioned blockchain, the risk of a %51 attack is greatly reduced due to the controlled nature of the consensus process. As for smart contract vulnerabilities, rigorous testing, code audits, and adopting best practices in smart contract development are critical for minimizing these risks. The proposed framework in this thesis is independent of the underlying blockchain technology. It can be adapted to any blockchain platform that allows the execution of smart contracts. Therefore, should any vulnerabilities arise within the underlying blockchain platform, the framework may be moved to another platform with more dense security features. Additionally, it is important to note that blockchain technology is still in its infancy. As the technology matures, the security concerns will also decrease.

4.4. Discussion on Possible Use Cases

We explore the potential use cases for our “Blockchain-Based Decentralized Identity and Trust Evaluation Framework” for IoT under this sub-section. The application of this framework extends to various cases where secure and decentralized identity management and trust evaluation are paramount. A particular focus will be on a network of trusted IoT manufacturers and the integration with the emerging technology of digital twins. In a consortium of IoT device manufacturers, each manufacturer is an independent entity and, while there is a mutual interest in interoperability, there isn't a complete trust among them, as they are rivals in the economy of IoT. The goal is to ensure that devices from different manufacturers can seamlessly communicate and operate with each other, enhancing market appeal and consumer convenience. Our proposed framework can serve as the foundation for a unique and global IoT IAM and trust evaluation network among these manufacturers. Utilizing blockchain technology offers a decentralized approach where each manufacturer maintains control over their devices while ensuring compatibility and secure interaction with devices from other manufacturers. Each device registered in this network receives a unique decentralized identity, managed, and verified within the blockchain framework. This setup ensures device authenticity and facilitates access control management, allowing devices from different manufacturers to interact securely based on predefined policies. In this environment, mutual authentication is crucial for secure device interaction. Our framework's mutual authentication scheme ensures that devices verify each other's identities before interaction, preventing unauthorized access and data breaches. Additionally, the trust evaluation mechanism continuously assesses device behavior, contributing to a dynamic and responsive network where trust levels are adjusted based on real-time interactions.

4.4.1. Integration with Digital Twins

Digital twins, virtual representations of physical devices, are becoming increasingly prevalent, especially in complex IoT ecosystems. These digital counterparts can optimize device performance, predict maintenance needs, and enhance overall system efficiency [93]. Our proposed framework can extend to authenticate digital twins, allowing them to interact securely with their physical counterparts and other digital twins. This seamless authentication is vital in scenarios where digital twins must exchange data or perform operations directly impacting the physical devices they represent. In a network where digital twins communicate with each other and with physical IoT devices, ensuring the integrity and security of the exchanged data is essential. The blockchain-based identity and trust evaluation mechanisms ensure that data exchanges occur only between verified entities and that the interactions are recorded immutably, enhancing transparency and accountability. In short, our framework's application to the digital twin ecosystem opens avenues for secure and efficient IoT operations, bridging the gap between the physical and digital worlds.

4.4.2. Integration with Mobile Phones

In the context of mobile phone manufacturers, the application of our framework can address several challenges, particularly in the areas of device identity management and inter-device communication. A critical area of focus would be managing International Mobile Equipment Identity (IMEI) numbers and enhancing interoperability and security between different mobile products.

In the context of mobile phone manufacturers, the application of our framework can address several challenges, particularly in the areas of device identity management and inter-device communication. A critical area of focus would be managing International Mobile Equipment Identity (IMEI) numbers and enhancing interoperability and security between different mobile products. IMEI numbers are unique identifiers for mobile phones, but issues like duplication, fraud, and manipulation pose significant challenges. These problems can lead to security vulnerabilities, complicate tracking of stolen devices, and disrupt network integrity. By integrating our framework's decentralized identity management capabilities, each mobile device can be assigned a unique, blockchain-verified digital identity linked to its IMEI number. This approach would significantly reduce the risks associated with IMEI duplication or fraud, as the blockchain ledger provides a tamper-proof record. Moreover, it would facilitate the tracking and verification of devices across different networks and regions.

As the mobile industry evolves, there's a growing emphasis on seamless interaction between devices within a single manufacturer's ecosystem and across different manufacturers. This interoperability requires a robust IAM system to ensure secure and efficient communication. Our framework can be adapted to serve as a basis for IAM and trust evaluation between mobile devices. By leveraging blockchain technology, the framework can authenticate devices, manage access controls, and continuously evaluate the trustworthiness of devices or other entities, such as applications, based on their interaction patterns. This system ensures that only authenticated devices can interact, thereby enhancing security and user privacy.

In an ecosystem where mobile phones from different manufacturers need to communicate, for instance, in smart home setups or for cross-platform applications, ensuring secure communication is paramount. The blockchain-based identity and access management facilitated by our framework can serve as a common standard for manufacturers, fostering a secure and interoperable ecosystem. The framework also upholds data privacy and integrity during inter-device communication. Blockchain's inherent characteristics, like data immutability and encryption, ensure that the data exchanged between devices remains secure and unaltered.

The application of our “Blockchain-Based Decentralized Identity and Trust Evaluation Framework” to the mobile phone industry presents an innovative solution to existing challenges around IMEI management and inter-device communication. It offers a way to strengthen mobile device security, integrity, and interoperability across different manufacturers. As the mobile industry continues to grow and integrate more deeply with IoT, adopting such a framework could be a strategic step toward creating a more secure, efficient, interoperable, and trustworthy digital environment. This could pave the way for new levels of collaboration and innovation among mobile phone manufacturers, enhancing the overall user experience and privacy.

4.5. Discussion on Possible Benefits and Drawbacks of the Framework

This chapter examines the advantages and disadvantages of the “Blockchain-Based Decentralized Identity and Trust Evaluation Framework” for IoT, analyzing its performance within the broader landscape of IAM solutions. The objective is to provide a balanced perspective, highlighting where the framework shows strengths and areas where it faces challenges.

Possible benefits of the proposed framework are listed below.

- **Decentralized Trust and Security:** The decentralized architecture of the framework enhances security and trust. The underlying blockchain technology's immutable ledger characteristics provide a transparent and tamper-proof system for managing identities and transactions.

- **Scalability Potential:** Despite limitations in the current testing environment, blockchain technology holds the potential for high scalability. Future advancements in the blockchain frameworks related to performance are expected to enable the proposed framework to handle an increasing number of IoT devices and transactions effectively.
- **Interoperability:** The framework facilitates interoperability between IoT providers and manufacturers, allowing devices from varied entities and networks to communicate seamlessly. This capability is considered vital for the expansion of IoT ecosystems.
- **Enhanced Privacy Control:** Blockchain technology inherently supports enhanced privacy, granting users more control over their data. In the context of IoT, this translates into improved control over device data and interactions.

Possible drawbacks of the proposed framework are listed below.

- **Performance Limitations:** Currently, the framework's transaction throughput, particularly for write operations like device registration and trust score updates, may not meet the demands of large-scale IoT networks. Continuous research in blockchain technology is likely to enhance these aspects. However, the query-based methods may meet the requirements for a large network. Additionally, Device Specific Access Control SCs can be defined for each device within the network. This allows SC to operate for a single device, therefore meeting the performance requirements of a single device.
- **Dependency on Blockchain Technology:** The framework's performance and security are closely linked to the underlying blockchain technology. Any inherent vulnerabilities or limitations in the blockchain could affect the framework's efficacy.

The proposed framework demonstrates considerable potential, particularly regarding security, decentralization, and interoperability. Current performance limitations and resource intensity are recognized challenges, yet these are active research areas within the blockchain field. As the technology evolves, it is anticipated that frameworks like ours will become more practical for real-world IoT applications, leading to more secure, efficient, and interconnected device networks. The continued development of blockchain technology is expected to address many of the current limitations, unlocking its full potential in the context of IoT IAM.

4.5.1. Comparative Analysis of IAM Systems

A comparison between the proposed framework and conventional centralized and federated IAM systems is shown in Table 4.5. This comparison spans across various Key Performance Indicators (KPIs), such as scalability, interoperability, security, and user-centric approaches. The table aims to highlight how these different architectures fare in terms of their capabilities and limitations, offering insights into each approach's relative strengths and potential drawbacks in the context of IoT environments.

Table 4.5 Comparison Table

KPI	Centralized IAM Systems	Federated IAM Systems	Proposed Solution
Scalability	Moderate (May struggle with very large networks)	High (Better suited for large user bases)	High (Designed for scalability in decentralized networks)
Performance	Varies (Depends on the resources available to the system)	Varies (Depends on the resources available to the system)	Low (Current blockchain implementations cannot cope with centralized solutions in terms of performance)
Interoperability	Moderate (Depends on proprietary standards)	High (Federation supports multiple domains)	Very High (Flexible integration with various IoT systems and possibility of being deployed as globally unique)

Mobility	Moderate (Depends on proprietary standards)	High (Federated approach is designed to enhance mobility)	High (Inherently supports ubiquitous IoT device mobility)
Security	High (Centralized control can offer security however, zero-day attacks can be devastating for the system [94])	High (Federated systems provide secure inter-domain transactions)	Very High (Enhanced by decentralized, cryptographic methods)
Privacy	Moderate (Centralized data storage can be a concern)	Moderate (Depends on federation agreements)	High (Blockchain enhances user privacy control)
Decentralization	Low (Single administrative control)	Moderate (Distributed across federated domains)	Very High (Inherent in blockchain architecture)
User-Centric Approach	Low (User control is limited)	Moderate (Varies with implementation)	High (Empowers users with control over their identity data)
Flexibility and Extensibility	Moderate (Can be limited by central infrastructure)	High (Federation allows for extensible identity solutions)	High (Adaptable to various IoT scenarios)
Ease of Management	High (Centralized management can be simpler)	Moderate (Complex federated agreement)	Moderate (Depends on blockchain complexity)
Cost-Efficiency	Moderate (Can be costly for large-scale operations)	High (Cost-effective for managing across domains)	Varies (Depends on blockchain implementation and scale)
User Experience	Moderate (Can be impacted by centralization)	High (SSO and federated access improve experience)	High (Designed for user-friendliness and control)

Compliance and Standards	High (Easier to enforce compliance centrally)	Moderate (Depends on federation standards)	Varies (Blockchain must align with evolving IoT standards)

Centralized IAM systems have traditionally been the backbone of identity management, offering robust security and ease of management due to their singular control. However, they often fall short in scalability, flexibility, and user-centricity, particularly in large and diverse networks like those in IoT environments. For the federated IAMs, they mark a significant advancement, particularly in interoperability and mobility, functioning well in scenarios involving multiple domains. They offer a more user-friendly experience, primarily due to the features like SSO. Blockchain-based IAM solutions emerge as a transformative approach, excelling in decentralization, user privacy, and security fueled by cryptographic methods inherent in blockchain technology. They are designed for high interoperability and flexibility, adapting seamlessly to various IoT scenarios. However, the cost-efficiency and ease of management of these systems can vary, influenced by the specific blockchain implementation and scale of deployment. While blockchain-based systems offer enhanced user privacy and control, challenges exist in aligning them with evolving IoT standards and managing the intricacies of blockchain technology. Their success and effectiveness will largely depend on how these challenges are addressed as the technology matures.

In summary, the shift towards Blockchain-Based IAM Solutions indicates the growing need for more secure, scalable, and user-centric identity management systems, especially in the increasingly interconnected world of IoT. This shift also underscores the need for continuous innovation and adaptation in IAM solutions to meet the evolving demands of technology and user expectations. As blockchain technology advances, it is expected to play a key role in redefining IAM for IoT and beyond, offering a balance between security, scalability, and user empowerment.

5. CONCLUSION AND FUTURE WORK

This thesis study proposes a comprehensive blockchain-based IAM, and trust evaluation system designed for the growing IoT landscape. It addresses the critical challenges of identity management, security, interoperability, and scalability that are inherent in the rapidly expanding IoT networks. The blockchain-based solution proposed in this study is particularly relevant given the limitations of existing server-client-based models, which struggle with IoT networks' increasing complexity and size. The system aims to create a secure, interoperable, and reliable ecosystem for globally connected IoT devices within a decentralized environment. Emphasizing features like transparency, security, and decentralization, the framework lays the foundation for a globally unique identity management system designed specifically for IoT devices. It incorporates specific authentication, authorization, access control, and trust evaluation methods, all underpinned by the decentralized identities maintained within the blockchain network. Additionally, the framework leverages blockchain technology's transparency and immutability features and creates a secure audit trail of the events inside the network. The utility of this framework extends to real-world scenarios, including supply chain management, digital twins, mobile phone industry, smart cities, healthcare systems, and IoT-based smart home appliances.

Key to the framework is its decentralized approach, leveraging blockchain's inherent properties like immutability, cryptographic security, and distributed ledger technology. This enhances the security and reliability of IoT networks and eliminates the single points of failure common in centralized systems. Moreover, the framework supports the increasing mobility and interoperability demands of diverse IoT devices, making it a promising solution for future IoT ecosystems.

This study introduces a trust evaluation protocol as a key element and as an extension of the proposed blockchain-based IAM framework. The protocol is designed to work smoothly with the IAM solution, but this study only outlines a basic concept. It shows how the system might operate with the IAM solution using blockchain technology's capabilities. However, the complete development of this system is a task for future research. This is because it requires the creation of complex algorithms and thorough cybersecurity testing. Another critical aspect of future work involves enhancing the framework's performance to manage potential bottlenecks caused by the blockchain network. This is essential to meet the performance demands of a global-scale IoT network. As blockchain technology continues to evolve, particularly in addressing scalability issues, the proposed framework in this study is expected to become more efficient and suitable for widespread IoT applications.

Further research will also explore the integration of the framework with emerging IoT technologies and standards. This includes expanding its applicability to various IoT scenarios, ensuring compliance with evolving IoT standards, and enhancing user experience. The aim is to create a flexible, user-centric framework that can adapt to the dynamic needs of the IoT landscape. Although the proposed framework operates independently of any specific blockchain platform, the study used Hyperledger Fabric, a permissioned blockchain platform. This choice is based on the need for trusted stakeholders within the blockchain network to handle device registrations. If not managed properly, the registration process could be misused, threatening the entire system's stability. A future extension of this study could involve creating a device registration method for public use, thereby applying the proposed framework to a public blockchain network.

In summary, this study contributes to the field of IoT security and identity management, showcasing the potential of blockchain technology in this domain. As the IoT world grows more interconnected, the need for decentralized, secure, and efficient identity and access management solutions becomes increasingly evident. The proposed framework, with its focus on decentralization, transparency, and user empowerment, paves the way for the realization of a more secure, interconnected IoT ecosystem. In conclusion, the research presented here marks an important step towards addressing the challenges in the IoT landscape. It highlights the potential of blockchain as a key technology in shaping the future of IoT, offering a framework that balances security, efficiency, and interoperability. As the technology matures and the framework evolves, it holds the promise of transforming IoT identity management and paving the way for its broader adoption in various digital realms.

6. REFERENCES

[1] Global IoT and non-IoT connections 2010-2025 | Statista, (n.d.). https://www.statista.com/statistics/1101442/iot-number-of-connected-devices-worldwide/ (accessed November 22, 2023).

[2] M. Di Pierro, What Is the Blockchain?, Comput Sci Eng 19 (2017) 92–95. https://doi.org/10.1109/MCSE.2017.3421554.

[3] How Many Blockchains are there? (2023 Guide), (n.d.). https://watcher.guru/news/how-many-blockchains-are-there (accessed November 22, 2023).

[4] F.A. Hadi, A.R. Hussein, J.R. Rashed, N. SAAD, H.A. ALSEELAWI, A vision of blockchain technology and its integration with IOT: Applications, challenges, and opportunities; from the authentication perspective, J Theor Appl Inf Technol 97 (2019) 4048.

[5] Number of IoT Devices (2023-2030), (n.d.). https://explodingtopics.com/blog/number-of-iot-devices (accessed November 22, 2023).

[6] A. Erdem, S.Ö. Yildirim, P. Angin, Blockchain for ensuring security, privacy, and trust in IoT environments: the state of the art, Security, Privacy and Trust in the IoT Environment (2019) 97–122.

[7] W. Zou, D. Lo, P.S. Kochhar, X.-B.D. Le, X. Xia, Y. Feng, Z. Chen, B. Xu, Smart contract development: Challenges and opportunities, IEEE Transactions on Software Engineering 47 (2019) 2084–2106.

[8] P.R. Sousa, J.S. Resende, R. Martins, L. Antunes, The case for blockchain in IoT identity management, Journal of Enterprise Information Management 35 (2020) 1477–1505.

[9] M. Nofer, P. Gomber, O. Hinz, D. Schiereck, Blockchain, Business & Information Systems Engineering 59 (2017) 183–187.

[10] Get Started - The Go Programming Language, (n.d.). https://go.dev/learn/ (accessed November 23, 2023).

[11] W. Python, Python, Python Releases for Windows 24 (2021).

[12] M. Cantelon, M. Harter, T.J. Holowaychuk, N. Rajlich, Node. js in Action, Manning Greenwich, 2014.

[13] T. Kubernetes, Kubernetes, Kubernetes. Retrieved May 24 (2019) 2019.

[14] K. Jangla, K. Jangla, Docker compose, Accelerating Development Velocity Using Docker: Docker Across Microservices (2018) 77–98.

[15] M. Kuzlu, M. Pipattanasomporn, L. Gurses, S. Rahman, Performance analysis of a hyperledger fabric blockchain framework: throughput, latency and scalability, in: 2019 IEEE International Conference on Blockchain (Blockchain), 2019: pp. 536–540.

[16] F. Khelifi, A. Bradai, A. Benslimane, P. Rawat, M. Atri, A survey of localization systems in internet of things, Mobile Networks and Applications 24 (2019) 761–785.

[17] W. Lv, F. Meng, C. Zhang, Y. Lv, N. Cao, J. Jiang, A general architecture of IoT system, in: 2017 IEEE International Conference on Computational Science and Engineering (CSE) and IEEE International Conference on Embedded and Ubiquitous Computing (EUC), 2017: pp. 659–664.

[18] M.A. Jabraeil Jamali, B. Bahrami, A. Heidari, P. Allahverdizadeh, F. Norouzi, M.A. Jabraeil Jamali, B. Bahrami, A. Heidari, P. Allahverdizadeh, F. Norouzi, IoT architecture, Towards the Internet of Things: Architectures, Security, and Applications (2020) 9–31.

[19] S.J. Pokorni, Reliability and availability of the Internet of things, Vojnotehnicki Glasnik/Military Technical Courier 67 (2019) 588–600.

[20] F. Piccialli, S. Cuomo, F. Giampaolo, G. Casolla, V.S. Di Cola, Path prediction in IoT systems through Markov Chain algorithm, Future Generation Computer Systems 109 (2020) 210–217.

[21] C.E. Perkins, Mobile ip, IEEE Communications Magazine 35 (1997) 84–99.

[22] A. Luntovskyy, L. Globa, Performance, reliability and scalability for IoT, in: 2019 International Conference on Information and Digital Technologies (IDT), 2019: pp. 316–321.

[23] M. Noura, M. Atiquzzaman, M. Gaedke, Interoperability in internet of things: Taxonomies and open challenges, Mobile Networks and Applications 24 (2019) 796–809.

[24] Essential Cybersecurity Measures for Healthcare IoT, (n.d.). https://www.linkedin.com/pulse/essential-cybersecurity-measures-healthcare-iot-sidra-zafar (accessed November 29, 2023).

[25] N.M.M. Banu, C. Sujatha, IoT architecture a comparative study, Int J Pur Appl Math 117 (2017) 45–49.

[26] J. Haxhibeqiri, E. De Poorter, I. Moerman, J. Hoebeke, A survey of LoRaWAN for IoT: From technology to application, Sensors 18 (2018) 3995.

[27] C.M. Ramya, M. Shanmugaraj, R. Prabakaran, Study on ZigBee technology, in: 2011 3rd International Conference on Electronics Computer Technology, 2011: pp. 297–301.

[28] M.B. Yassein, W. Mardini, A. Khalil, Smart homes automation using Z-wave protocol, in: 2016 International Conference on Engineering & MIS (ICEMIS), 2016: pp. 1–6.

[29] G. Mulligan, The 6LoWPAN architecture, in: Proceedings of the 4th Workshop on Embedded Networked Sensors, 2007: pp. 78–82.

[30] D. Soni, A. Makwana, A survey on mqtt: a protocol of internet of things (iot), in: International Conference on Telecommunication, Power Analysis and Computing Techniques (ICTPACT-2017), 2017: pp. 173–177.

[31] C. Bormann, A.P. Castellani, Z. Shelby, Coap: An application protocol for billions of tiny internet nodes, IEEE Internet Comput 16 (2012) 62–67.

[32] I. Fette, A. Melnikov, The websocket protocol, 2011.

[33] Z.D. Patel, A review on service oriented architectures for internet of things (iot), in: 2018 2nd International Conference on Trends in Electronics and Informatics (ICOEI), 2018: pp. 466–470.

[34] I. Ungurean, N.C. Gaitan, V.G. Gaitan, A Middleware Based Architecture for the Industrial Internet of Things., KSII Transactions on Internet & Information Systems 10 (2016).

[35] K.A. Alaghbari, M.H.M. Saad, A. Hussain, M.R. Alam, Complex event processing for physical and cyber security in datacentres-recent progress, challenges and recommendations, Journal of Cloud Computing 11 (2022) 65.

[36] H. Mrabet, S. Belguith, A. Alhomoud, A. Jemai, A survey of IoT security based on a layered architecture of sensing and data analysis, Sensors 20 (2020) 3625.

[37] J.L. Camp, Digital identity, IEEE Technology and Society Magazine 23 (2004) 34–41.

[38] J. Chen, Y. Liu, Y. Chai, An identity management framework for internet of things, in: 2015 IEEE 12th International Conference on E-Business Engineering, 2015: pp. 360–364.

[39] D. Pöhn, W. Hommel, An overview of limitations and approaches in identity management, in: Proceedings of the 15th International Conference on Availability, Reliability and Security, 2020: pp. 1–10.

[40] H. Lockhart, B. Campbell, Security assertion markup language (saml) v2. 0 technical overview, OASIS Committee Draft 2 (2008) 94–106.

[41] D. Hardt, The OAuth 2.0 authorization framework, 2012.

[42] D. Fett, R. Küsters, G. Schmitz, The web sso standard openid connect: In-depth formal security analysis and security guidelines, in: 2017 IEEE 30th Computer Security Foundations Symposium (CSF), 2017: pp. 189–202.

[43] S.S.Y. Shim, G. Bhalla, V. Pendyala, Federated identity management, Computer (Long Beach Calif) 38 (2005) 120–122.

[44] A. Josang, M. AlZomai, S. Suriadi, Usability and privacy in identity management architectures, in: ACSW Frontiers 2007: Proceedings of 5th Australasian Symposium on Grid Computing and e-Research, 5th Australasian Information Security Workshop (Privacy Enhancing Technologies), and Australasian Workshop on Health Knowledge Management and Discovery, 2007: pp. 143–152.

[45] U. Der, S. Jähnichen, J. Sürmeli, Self-sovereign identity - opportunities and challenges for the digital revolution, ArXiv Preprint ArXiv:1712.01767 (2017).

[46] U. Habiba, R. Masood, M.A. Shibli, M.A. Niazi, Cloud identity management security issues & solutions: a taxonomy, Complex Adaptive Systems Modeling 2 (2014) 1–37.

[47] A. Ometov, S. Bezzateev, N. Mäkitalo, S. Andreev, T. Mikkonen, Y. Koucheryavy, Multi-factor authentication: A survey, Cryptography 2 (2018) 1.

[48] Okta Identity Engine | Okta, (n.d.). https://help.okta.com/oie/en-us/content/topics/identity-engine/oie-index.htm (accessed December 6, 2023).

[49] D. Subbarao, B. Raju, F. Anjum, C. venkateswara Rao, B.M. Reddy, Microsoft Azure active directory for next level authentication to provide a seamless single sign-on experience, Appl Nanosci 13 (2023) 1655–1664.

[50] Cloud Identity | Google Cloud, (n.d.). https://cloud.google.com/identity# (accessed December 6, 2023).

[51] G.J. Simmons, A survey of information authentication, Proceedings of the IEEE 76 (1988) 603–620.

[52] S. di Vimercati, S. Foresti, P. Samarati, Authorization and access control, Security, Privacy, and Trust in Modern Data Management (2007) 39–53.

[53] R.S. Sandhu, Role-based access control, in: Advances in Computers, Elsevier, 1998: pp. 237–286.

[54] V.C. Hu, D.R. Kuhn, D.F. Ferraiolo, J. Voas, Attribute-based access control, Computer (Long Beach Calif) 48 (2015) 85–88.

[55] What Is an IT Audit? A Definitive Guide to Safeguard Your Data, (n.d.). https://track.g2.com/resources/it-audit (accessed December 6, 2023).

[56] Y. Cao, L. Yang, A survey of identity management technology, in: 2010 IEEE International Conference on Information Theory and Information Security, 2010: pp. 287–293.

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

[58] C. Dannen, Introducing Ethereum and solidity, Springer, 2017.

[59] Blockchain Structure - GeeksforGeeks, (n.d.). https://www.geeksforgeeks.org/blockchain-structure/ (accessed December 8, 2023).

[60] K.S. Garewal, K.S. Garewal, Merkle trees, Practical Blockchains and Cryptocurrencies: Speed Up Your Application Development Process and Develop Distributed Applications with Confidence (2020) 137–148.

[61] J. Bhosale, S. Mavale, Volatility of select crypto-currencies: A comparison of Bitcoin, Ethereum and Litecoin, Annu. Res. J. SCMS, Pune 6 (2018) 132–141.

[62] A. Gervais, G.O. Karame, K. Wüst, V. Glykantzis, H. Ritzdorf, S. Capkun, On the security and performance of proof of work blockchains, in: Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, 2016: pp. 3–16.

[63] Cardano Docs, (n.d.). https://docs.cardano.org/ (accessed December 9, 2023).

[64] J. Burdges, A. Cevallos, P. Czaban, R. Habermeier, S. Hosseini, F. Lama, H.K. Alper, X. Luo, F. Shirazi, A. Stewart, others, Overview of polkadot and its design considerations, ArXiv Preprint ArXiv:2005.13456 (2020).

[65] A. Gervais, G.O. Karame, K. Wüst, V. Glykantzis, H. Ritzdorf, S. Capkun, On the security and performance of proof of work blockchains, in: Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, 2016: pp. 3–16.

[66] D. Huang, X. Ma, S. Zhang, Performance analysis of the raft consensus algorithm for private blockchains, IEEE Trans Syst Man Cybern Syst 50 (2019) 172–181.

[67] Smart Contract Development | Hack, (n.d.). https://hack.bg/dlt-blockchain-development-services/smart-contracts-development/ (accessed December 9, 2023).

[68] Corda Permissioned Distributed Ledger Technology (DLT) | R3, (n.d.). https://r3.com/products/corda/ (accessed December 9, 2023).

[69] E. Androulaki, A. Barger, V. Bortnikov, C. Cachin, K. Christidis, A. De Caro, D. Enyeart, C. Ferris, G. Laventman, Y. Manevich, others, Hyperledger fabric: a distributed operating system for permissioned blockchains, in: Proceedings of the Thirteenth EuroSys Conference, 2018: pp. 1–15.

[70] A. CouchDB, Apache couchdb, URL Https://Couchdb. Apache. Org (n.d.).

[71] LevelDB - Wikipedia, (n.d.). https://en.wikipedia.org/wiki/LevelDB (accessed December 9, 2023).

[72] What is Hyperledger Fabric? - Hyperledger Fabric Explained - AWS, (n.d.). https://aws.amazon.com/tr/blockchain/what-is-hyperledger-fabric/ (accessed December 9, 2023).

[73] M. Kuzlu, M. Pipattanasomporn, L. Gurses, S. Rahman, Performance analysis of a hyperledger fabric blockchain framework: throughput, latency and scalability, in: 2019 IEEE International Conference on Blockchain (Blockchain), 2019: pp. 536–540.

[74] Architecture | Hyperledger Caliper, (n.d.). https://hyperledger.github.io/caliper/v0.2/architecture/ (accessed December 9, 2023).

[75] Understanding Decentralized IDs (DIDs) | by Adam Powers | Medium, (n.d.). https://medium.com/@adam_14796/understanding-decentralized-ids-dids-839798b91809 (accessed December 10, 2023).

[76] Decentralized Identity: The Ultimate Guide 2023, (n.d.). https://www.dock.io/post/decentralized-identity (accessed December 10, 2023).

[77] A.S. Omar, O. Basir, Identity management in IoT networks using blockchain and smart contracts, in: 2018 IEEE International Conference on Internet of Things (IThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData), 2018: pp. 994–1000.

[78] X. Fan, Q. Chai, L. Xu, D. Guo, Diam-iot: A decentralized identity and access management framework for internet of things, in: Proceedings of the 2nd

ACM International Symposium on Blockchain and Secure Critical Infrastructure, 2020: pp. 186–191.

[79] M.T. Hammi, B. Hammi, P. Bellot, A. Serhrouchni, Bubbles of Trust: A decentralized blockchain-based authentication system for IoT, Comput Secur 78 (2018) 126–142.

[80] P. Angin, M.B. Mert, O. Mete, A. Ramazanli, K. Sarica, B. Gungoren, A blockchain-based decentralized security architecture for IoT, in: Internet of Things–ICIOT 2018: Third International Conference, Held as Part of the Services Conference Federation, SCF 2018, Seattle, WA, USA, June 25-30, 2018, Proceedings 3, 2018: pp. 3–18.

[81] A. Dixit, M. Smith-Creasey, M. Rajarajan, A Decentralized IIoT Identity Framework based on Self-Sovereign Identity using Blockchain, in: 2022 IEEE 47th Conference on Local Computer Networks (LCN), 2022: pp. 335–338.

[82] T. Ranathunga, R. Marfievici, A. McGibney, S. Rea, A DLT-based trust framework for IoT ecosystems, in: 2020 International Conference on Cyber Security and Protection of Digital Services (Cyber Security), 2020: pp. 1–8.

[83] A. Sghaier Omar, Decentralized identity and access management framework for Internet of Things devices, (2020).

[84] S.B. Öztürk, M. Aydos, A Blockchain Based Decentralized Identity, Access Management, and Trust Evaluation Framework for IoT, in: 2023 16th International Conference on Information Security and Cryptology (ISCTürkiye), 2023: pp. 1–6.

[85] BugrahanOzturk/Decentralized-Identity-Platform-for-IoT: A decentralized identity platform specific for IoT devices. Built with hyperledger fabric., (n.d.). https://github.com/BugrahanOzturk/Decentralized-Identity-Platform-for-IoT (accessed December 23, 2023).

[86] fabric-samples/asset-transfer-basic/chaincode-go at main · hyperledger/fabric-samples, (n.d.). https://github.com/hyperledger/fabric-samples/tree/main/asset-transfer-basic/chaincode-go (accessed December 23, 2023).

[87] Hyperledger Fabric SDKs — hyperledger-fabricdocs main documentation, (n.d.). https://hyperledger-fabric.readthedocs.io/en/release-2.2/fabric-sdks.html (accessed December 23, 2023).

[88] Prometheus - Monitoring system & time series database, (n.d.). https://prometheus.io/ (accessed December 23, 2023).

[89] Install Grafana | Grafana documentation, (n.d.). https://grafana.com/docs/grafana/latest/setup-grafana/installation/ (accessed December 23, 2023).

[90] Apache JMeter - Apache JMeter™, (n.d.). https://jmeter.apache.org/ (accessed December 25, 2023).

[91] A. Mallik, Man-in-the-middle-attack: Understanding in simple words, Cyberspace: Jurnal Pendidikan Teknologi Informasi 2 (2019) 109–134.

[92] J.R. Douceur, The sybil attack, in: International Workshop on Peer-to-Peer Systems, 2002: pp. 251–260.

[93] R. Minerva, G.M. Lee, N. Crespi, Digital twin in the IoT context: A survey on technical features, scenarios, and architectural models, Proceedings of the IEEE 108 (2020) 1785–1824.

[94] L. Bilge, T. Dumitraş, Before we knew it: an empirical study of zero-day attacks in the real world, in: Proceedings of the 2012 ACM Conference on Computer and Communications Security, 2012: pp. 833–844.