Decentralized Identity and Access Management Framework for Internet of Things Devices

by

Ahmad Sghaier Omar

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Doctor of Philosophy
in
Electrical and Computer Engineering

Waterloo, Ontario, Canada, 2020

Examining Committee Membership

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

External Examiner:	Reza Parizi Assistant Professor, Kennesaw State University
Supervisor:	Otman Basir Professor, University of Waterloo
Internal Member:	Ladan Tahvildari Professor, University of Waterloo
Internal Member:	Mark Crowley Assistant Professor, University of Waterloo
Internal-External Member:	Eihab Abdel-Rahman Associate Professor, University of Waterloo

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

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

Abstract

The emerging Internet of Things (IoT) domain is about connecting people and devices and systems together via sensors and actuators, to collect meaningful information from the devices surrounding environment and take actions to enhance productivity and efficiency.

The proliferation of IoT devices from around few billion devices today to over 25 billion in the next few years spanning over heterogeneous networks defines a new paradigm shift for many industrial and smart connectivity applications. The existing IoT networks faces a number of operational challenges linked to devices management and the capability of devices' mutual authentication and authorization.

While significant progress has been made in adopting existing connectivity and management frameworks, most of these frameworks are designed to work for unconstrained devices connected in centralized networks. On the other hand, IoT devices are constrained devices with tendency to work and operate in decentralized and peer-to-peer arrangement. This tendency towards peer-to-peer service exchange resulted that many of the existing frameworks fails to address the main challenges faced by the need to offer ownership of devices and the generated data to the actual users. Moreover, the diversified list of devices and offered services impose that more granular access control mechanisms are required to limit the exposure of the devices to external threats and provide finer access control policies under control of the device owner without the need for a middleman.

This work addresses these challenges by utilizing the concepts of decentralization introduced in Distributed Ledger (DLT) technologies and capability of automating business flows through smart contracts. The proposed work utilizes the concepts of decentralized identifiers (DIDs) for establishing a decentralized devices identity management framework and exploits Blockchain tokenization through both fungible and non-fungible

tokens (NFTs) to build a self-controlled and self-contained access control policy based on capability-based access control model (CapBAC). The defined framework provides a layered approach that builds on identity management as the foundation to enable authentication and authorization processes and establish a mechanism for accounting through the adoption of standardized DLT tokenization structure.

The proposed framework is demonstrated through implementing a number of use cases that addresses issues related identity management in industries that suffer losses in billions of dollars due to counterfeiting and lack of global and immutable identity records. The framework extension to support applications for building verifiable data paths in the application layer were addressed through two simple examples.

The system has been analyzed in the case of issuing authorization tokens where it is expected that DLT consensus mechanisms will introduce major performance hurdles. A proof of concept emulating establishing concurrent connections to a single device presented no timed-out requests at 200 concurrent connections and a rise in the timed-out requests ratio to 5% at 600 connections. The analysis showed also that a considerable overhead in data link budget of 10.4% is recorded due to the use of self-contained policy token which is trade-off between building self-contained access tokens with no middleman and link cost.

Acknowledgements

All praise is to Allah for giving me the ability and knowledge to reach this stage.

I would like to express my sincere gratitude to my supervisor, Prof. Otman Basir, for his profound supervision and his guidance, support, encouragement and the long hours of discussions. I would like also to extend my appreciation and thanks to the thesis committee members for taking the time in reviewing my thesis and providing their valuable remarks.

I would like to offer a special acknowledgment to my role model Prof. Mohammed Samir Elbuni for his support and guidance throughout my entire career.

Last and by far not least, I am indebted to my mother, Fatima, for her support throughout my life with patience, and prayers. I owe a great deal of gratitude to my wife, Halima, and my kids, Kenda, Adam, Nawfel, and Layan, for their continued understanding and encouragement, and all the support and love during the years of my study.

Dedication

To my wife and kids.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

The emerging Internet of Things (IoT) domain is about connecting people and devices and systems together via sensors and actuators, to collect meaningful information from the devices' surrounding environment and take actions to enhance productivity and efficiency. The proliferation of IoT devices from around a few billion devices today to over 25 billion in the next few years defines a new paradigm shift for many industrial and smart connectivity applications. Major telecommunication equipment vendors such as Cisco and Ericsson, [1, 2], reported in early 2019 that there are around 7-8 billion IoT devices connected with an estimate of 20-25 billion devices to be connected by the year 2023. An earlier report by Ericsson published in 2016, [3], showed a compound annual growth rate (CAGR) of 22-27% for IoT devices surpassing the growth rate of mobile phones, a trend starting from 2018, as shown in Figure 1.1.

The increased usage of IoT devices among a wide spectrum of domains and its proliferation from 2-3 devices per person today to an expected number of 10-15 devices per person in the coming years, has already opened up a lot of new opportunities on how data and

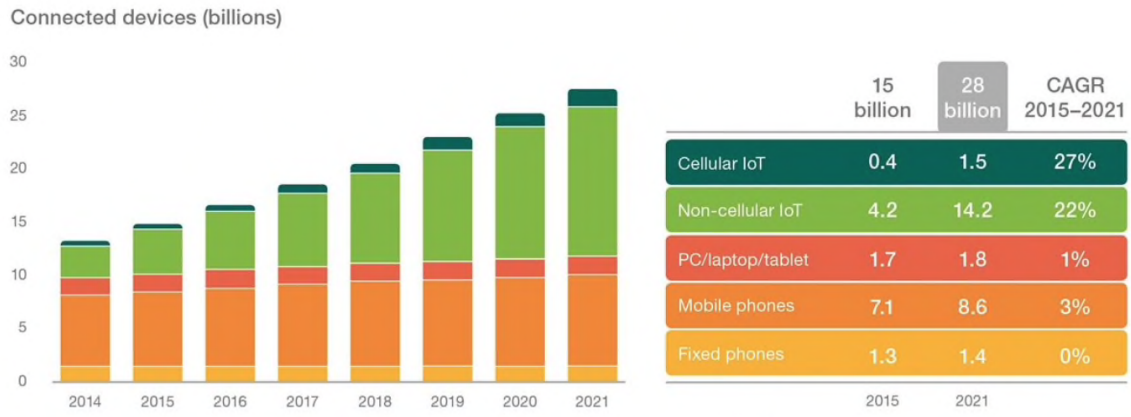

Figure 1.1: Number of IoT Devices and Growth Rate (2104-2021). Source: Ericsson.

services offered by IoT devices can be monetized [4]. A number of startups and also big corporations are defining new business models for IoT devices' data exchange including IOTA as an IoT transaction settlement and data transfer layer, Grid+ by ConsenSys offering consumers direct access to wholesale energy markets, and Streamr offering a marketplace for monetizing real-time stream of IoT data [5, 6, 7].

Examples of current IoT products with real monetization concepts include the Nest thermostat form Google. Nest device is among the early entrants into the IoT world used to control temperature schedules for the potential to save 10-15% of the money spent on home energy [8]. Another example is smart street lights which helped the city of Los Angeles to save 63% on energy costs in the first year using connected street lights [9]. Farmobile is enabling another real-life example of monetizing IoT data while giving the actual owners more value. Farmobile is an IoT data aggregator that enables farmers to collect, share and sell their agronomic and machine data to equipment manufacturers, agronomists, insurers, and other interested parties. To date, the Farmobile IoT data collection devices have collected data from various sources in more than one million acres of farmland [10].

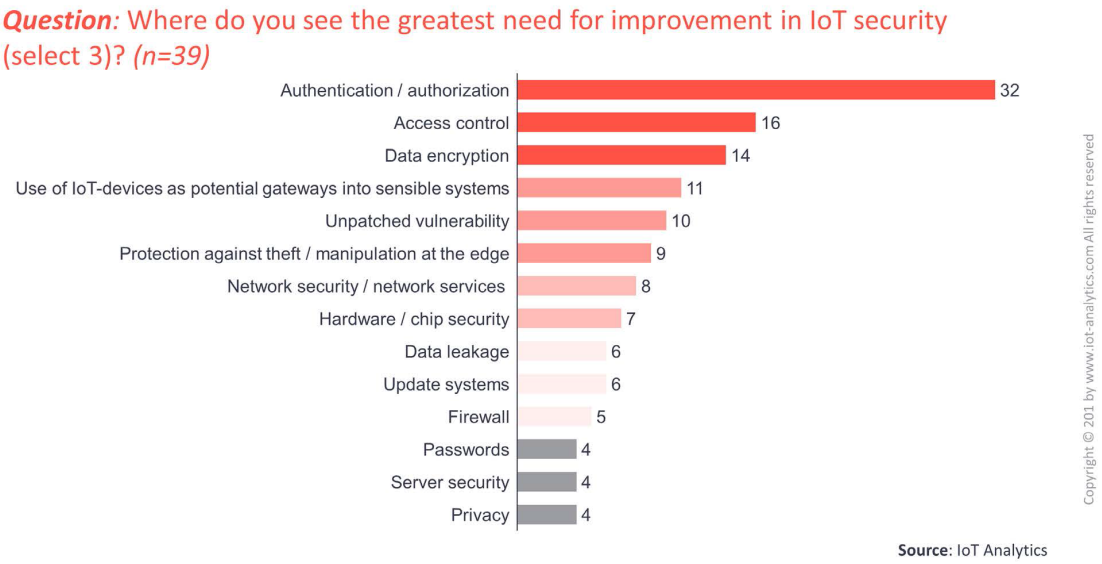

Figure 1.2: Ranking of IoT Security Weaknesses. Source: IoT Analytics

However, along all the IoT networks benefits many challenges arise, which include operational challenges linked to devices management and security and privacy concerns. The device management functionality is considered a crucial task in IoT operations due to the large number of devices managed and the heterogeneity of the network. A recent report on IoT platforms market, by Kenneth Research agency, showed that the IoT platforms market would reach approximately USD 74.74 billion by 2023. It presented that the growth of IoT device management platforms by year is expected to hit a CAGR of 25.1% in market size, falling only second to connectivity platforms, which are anticipated to grow at a CAGR of 26.3% [11].

Whereas smart meters improve usage monitoring and enable remote control and shut down of energy-consuming devices, however access to records of telemetry data can result in a security compromise. The concerns about securing authentication, authorization, and access control for IoT devices, and the information gathered is driving the industry for

finer control on who has access to devices and how those devices are used. An industry report by IoT Analytics covering the period 2017-2022 gathered input from technologists about the greatest need for improvement in IoT security, where respondents ranked authentication/authorization and access control on top of the list [12], as in Figure 1.2.

To date, one of the most known examples of an IoT network attack is a Distributed Denial of Service (DDoS) attack on the domain registration service, Dyn. This attack utilized several distributed vulnerable IoT devices through the Mirai botnet and managed to take down the DNS service of Dyn, which resulted in interruption of services of Amazon, Github, Twitter and Netflix, and others [13]. One of the main reasons behind the success of such an attack is the ill-defined security and access control roles and reliance on coarse-grained authorization rules. Coarse-grained roles have been proven as an ineffective mechanism against many attacks on IoT networks. Therefore, IoT networks require a mechanism that facilitates operations and still secure communication among the devices by building a foundation for devices identities linked to an authentication and authorization mechanism under control of device owners with fine-grained access control rules.

1.1 Motivation

As the number of IoT devices increases in proliferation in factories, on roads, at homes, and connected to our bodies, the challenges and concerns of unauthorized access to those devices and the confidential data gathered arise. One of the most profound challenges is linked to identity management, which pertains to how devices′ identities are authenticated and verified in addition to how devices establish the means for authorizing and controlling access to data and services offered.

Furthermore, the access control and authorization mechanism are even more compli-

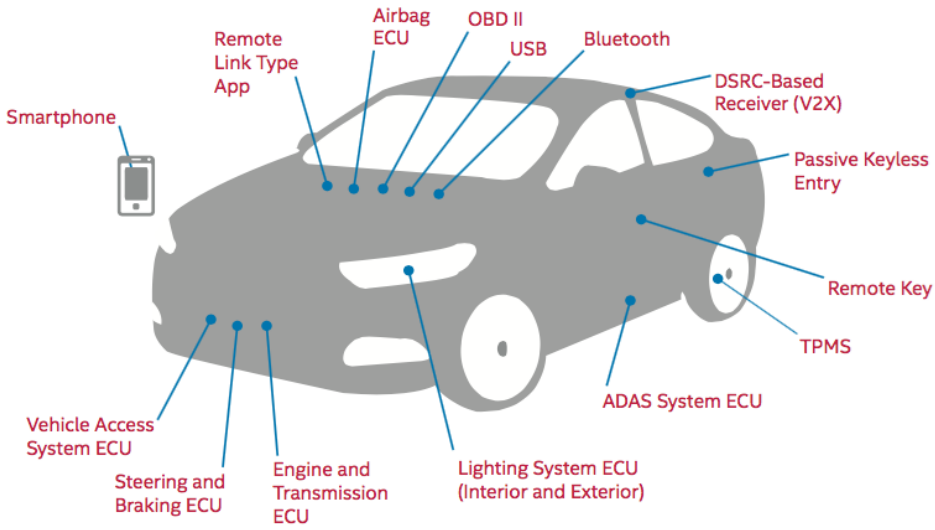

Figure 1.3: A Diagram showing different Systems in a connected vehicle that are accessed by different entities. Source: CSIdentity

cated when considering having a number of 10-15 devices/user. That can translate to 30-50 devices per smart home or smart asset, and the fact that the setup in many situations is based on devices procured from different vendors and connected through various connectivity options. For example, the de-facto standard for managing access to many of the home-connected devices is performed at the home gateway router with coarse-grained access rules usually managed and controlled by the communication service provider (CSP). Another example highlighting the sophisticated devices identity and access management that defines "who" can access "what" is a vehicle. In a connected vehicle that is equipped with tens of sensors, few parties play the role of either managing or obtaining access, such as fleet managers, insurance providers, roadside units, and anyone else using a connected vehicle.

Thus, providing secure and fine-granularity authentication and authorization frame-

work for IoT networks is an essential topic in the industry [12]. In this regard, a number of studies looked at applying the existing authentication and authorization mechanisms in constrained environments [14, 15, 16]. However, the constrained IoT device environment presents its own challenges. These challenges are related to suitability of use of current security protocols, and the demand for designing lightweight security protocols that adapt to power, processing, and memory limitations. The magnitude of the challenges is magnified by the need of mechanisms that define user-centric and fine-grained authentication and authorization models, in which devices' owners are in full control of who access what, when and how.

However, the vast majority of IoT platforms are based on a centralized model where an authority handles the interaction between its devices and with other devices belonging to other authorities [17]. Subsequently, many issues are mounting, especially in the scenarios in which devices need to exchange data in a peer-to-peer fashion. This specific requirement has led to the introduction of the decentralized approach since decentralization provides devices owners the ability to exert direct control over the services and data generated by the devices.

The distributed ledger technology (also known as Blockchain) can facilitate the implementation of decentralized IoT platforms, that can maintain a secure and trusted data exchange as well as the historical records. In such decentralized architecture, the Blockchain serves as the general ledger, keeping authentic records of all the transactions and events exchanged between smart devices in a decentralized topology [18].

Blockchain – a shared and distributed ledger technology – is a suitable candidate to address the challenges of building a decentralized IoT device management framework. That is mainly attributed to Blockchain use of cryptographic identifiers, records immutability, provenance, and tokenization. These features, combined, provide a platform to implement

Table 1.1: Blockchain Main Features.

Feature	**Benefit**
Near-real time	It enables near real-time settlement of transactions with less friction.
No Intermediary	Building on the cryptogrphic proofs it allows any two parties to transact directly without the need for a trusted third party.
Distributed Ledger	Making transactions and records synchronization across networks happen for multi-organizational business networks such as supply chains or other consortia.
Immutability and Irreversibility	Blockchain contains recorded and verified blocks of transactions that are chained and linked to disallow any manipulation of previous records.
Provenance	Having immutable records provide the capability to represent assets ownership and trace of origin throughout the entire life cycle.
Smart Contracts	Stored procedures to process pre-defined business rules to execute commercial and legal transactions that gets enforced without the involvement of an intermediary.

the functions of IoT devices identity and authentication and authorization management that ensures a global and unique identity for the devices, and also provide the mechanism to maintain it throughout the device life cycle. Applying Blockchain technology for building a decentralized management framework enables IoT devices to authenticate and authorize access without requiring a centralized authority to authenticate and authorize access to the services offered. Besides, such a decentralized architecture, having the Blockchain serve as the general ledger, enables organizations to keep authentic records of all the transactions and events exchanged between smart devices in a decentralized topology for purposes of auditing and accounting [18].

1.2 Problem Statement

The increased adoption of IoT devices in every day life activity makes these devices among the most vulnerable targets for attackers. IoT networks' dynamic and heterogeneous nature, with intermittent device connectivity, makes it possible for an adversary to target manipulating devices identities (Sybil attack) to take down the reputation of the system.

In case of identity manipulation, it is essential that once the device is affected and get detected to maintain the history of the reputation incident to prevent it from affecting other devices. Therefore, having a global and unique device identity with an immutable record is an approach that can boost confidence in devices' reputation while roaming heterogeneous networks and changing ownership throughout the life cycle. The system needs to hold a transparent and tamper-proof record with access to all historical transactions.

On the other hand, coarse-grained access rules for authorization in IoT networks based on regular IP networks access control mechanisms, including packet filters and firewalls are not addressing access rules specific to IoT data and services. In these systems, access rules are either based on network identifiers such as IP addresses and ports or packet content. However, IoT devices tend to offer particular and diverse data and services that are more user-centric. Thus, self-controlled access control rules are more favorable, especially in peer-to-peer applications, which contribute to a large segment of IoT applications.

Therefore, applying self-controlled and fine-grained access control rules supported by a decentralized authorization mechanism can ensure that owners have more control over access to devices' data and services [12]. This approach requires the capability to map access control rules into unique and non-interchangeable access tokens under the sole control of the device owner. The research addresses some of the main limitations in current IoT device management platforms with a focus on:

- Fragmented device identity management systems: The proliferation of IoT devices in heterogeneous networks, built by different vendors and deployed by multiple operators, created silos of identity management domains that limits portability and identity life cycle management.

- Lightweight framework in constrained environments: The need for a lightweight framework is a demand in most IoT applications. These devices are characterized by their constrained environment and by being deployed in heterogeneous networks, which require the design of an authentication and authorization framework that provides a more user and device-centric approach with lower computation requirements.

- Peer-to-peer service exchange: IoT devices are foreseen to operate in a p2p model, which is not feasible in a central authority coarse-grained authentication and authorization policy. Therefore, self-contained and self-controlled access mechanisms are required.

1.3 Research Goals

The main goal of this work is to design a decentralized framework for the management of IoT devices and develop smart contracts that enhance the security, privacy, and authenticity of data exchange and enable peer-to-peer IoT applications. A decentralized IoT device management framework is sought as the primary goal of this work by achieving the following objectives:

- Define a layered decentralized IoT device management framework for identity, authentication, authorization, and accounting management based on smart contracts and distributed ledger.

- Map the IoT device identity functionality to a decentralized framework that provides a global and self-owned device identity with features of portability and traceability throughout the device life cycle.

- Designing an authentication and authorization framework that is more user and device-centric with lower computation requirement and that enables peer-to-peer IoT applications.

- Demonstrating the generality and applicability of the proposed framework through the implementation of several use cases in different domains.

1.4 Contribution

This work proposes a decentralized IoT device management framework for enabling peer-to-peer and user-centric IoT services exchange. The proposed decentralized framework significantly reduces the reliance of an IoT device(s) owner on a centralized authority to control access and perform the service provisioning by building a foundation of decentralized self-owned digital identities and access and authorization tokens in an open verifiable environment. The proposed framework addresses the problem of IoT device management from the angles of organizational design, technological foundation, as well as possible business models and use cases. The main contributions of this research include:

- The research work produced a decentralized IoT device management framework that is based on the layers of identity management, authentication, authorization, and accounting management while highlighting some non-functional requirements, e.g., portability and provenance.

- Defined a decentralized self-owned IoT device identity method based on the W3C specification, with defined methods for identity transfer and delegation.
- Building a novel concept for authorization and access control model utilizing Blockhcian tokenization to create a self-contained capability-based access token with features of revocation, delegation, and accounting. The designed model is based on industry standards for tokenization to ensure portability and extensibility.
- Building two use cases that demonstrate how decentralized identity and Blockchain tokenization are used to solve problems related to smartphone devices theft and counterfeiting and to build a system that enhances the adherence of pharmaceuticals supply chain systems to the latest regulations.
- Demonstrating the extensibility of the decentralized device identity management framework to support building verifiable and provenant data paths. Ensuring that the consumers of IoT devices data can verify its authenticity and trace it to the origin.

1.5 Organization

This thesis is organized as follows: Chapter 2 provides a comprehensive background to the IoT device management problem from the perspective of identity management and authentication and authorization functionality. A generic design of the proposed decentralized framework is devised in Chapter 3. Chapter 4 defines a detailed tokenization approach for a decentralized IoT devices authorization and accounting mechanism. An anti-counterfeiting system for smartphones using the designed decentralized identity management framework is presented in Chapter 5. In Chapter 6, a composable non-fungible blockchain-hosted

token as a mechanism for securing the pharmaceuticals supply chain is presented. The concept of extending the proposed framework to enable provable and verifiable data paths in IoT networks is introduced in Chapter 7. Finally, the conclusion and future work is presented in Chapter 8.

Chapter 2

Background and Literature Review

The MIT Auto-ID Center firstly introduced the concept of the Internet of Things (IoT) through Kevin Ashton and David L. Brock back in 1999. Their vision is to establish an ecosystem that can connect physical objects and real-time service platforms to improve the lives of human beings. IoT, since its inception, is envisioned of being ubiquitous in all its aspects, intended to accommodate an interdisciplinary framework to enable connecting sensing and actuating devices through communication and networking and thus bringing information and systems in a seamless way to the use of everyone.

IoT applications are expected to include billions of everyday objects in various domains, namely: smart homes, smart cities, industrial applications, wearables, healthcare, and many others. A simple illustration included in Figure 2.1 is from my first reading on IoT in the article "The Internet of Things" by Neil Gershenfeld, Raffi Krikorian, and Danny Cohen. The picture explains their vision on how IoT works in a smart home scenario in the Scientific American October 2004 edition [19].

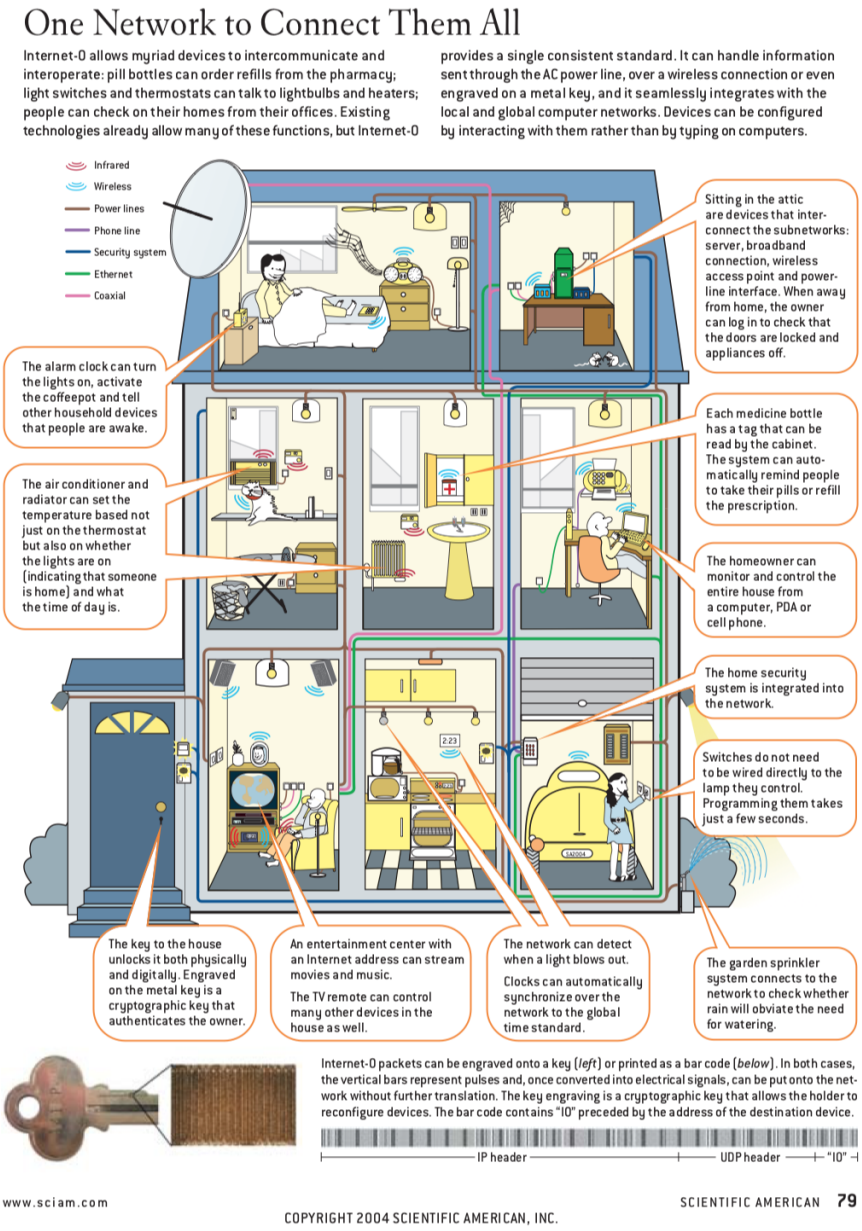

Figure 2.1: An IoT Smart Home Example. Source: Scientific American

2.1 IoT Architecture

IoT is an ecosystem that is building on several technologies and frameworks to enable gathering information from the surrounding environment and connecting devices with humans and platforms. Enabling this in a heterogeneous setup resulted in the existence of different IoT architectures. These architectures each describe the ecosystem from a different perspective, however, a very simplified representation is composed of three main layers:

- the perception layer is the main interface to the object environment to sense and collect data.
- the network layer has the functionality of transferring the collected data from the perception layer to the application layer and deliver telemetry commands to actuators in the perception layer.
- the application layer contains the intelligence required to carry out the analysis procedures on collected data and to perform the necessary management functions.

The three layers comprise a set of protocols and frameworks to connect IoT networks to the Internet and other business processes. For example, the network layer includes routers and gateways to connect IoT devices to the cloud and to provide an interface to send data to the application layer. This is achieved using different networking protocols; such as the Constrained Application Protocol (CoAP) [20], Message Queuing Telemetry Transportation (MQTT) [21], and Lightweight Machine to Machine (LwM2M) [22].

The application layer includes the analytics and processing applications in addition to the management functions such as device management, which manages the device life

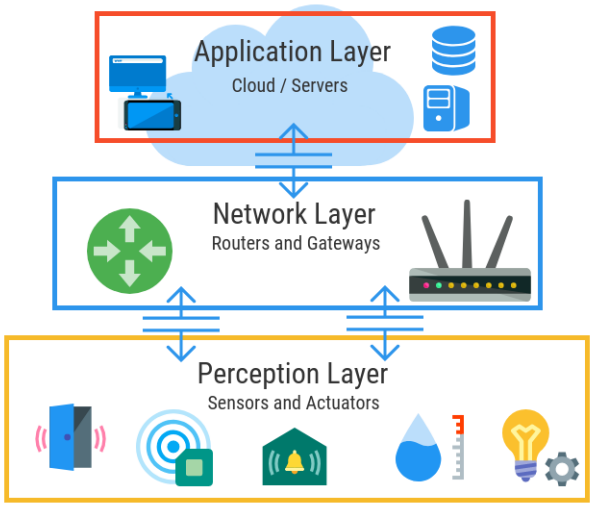

Figure 2.2: Three Layer IoT Architecture.

cycle including configuration, inventory, device identity management and authorization and access control.

Those device management functions pertain to how devices are identified in the different layers and the processes used to maintain the device identity during its life cycle. Devices' identities are a core component in IoT protocols, and they are used in all layers to enable information routing, service delivery, authentication, and authorization of access. Building decentralized IoT devices management relies on devices identities as a foundation layer; thus, the next section introduces the concepts related to identity management and the work conducted in building device identity management frameworks for IoT.

2.2 Identity Management in IoT

In general, identity management (IdM) is an administrative domain that deals with identifying individual entities in a system (such as a user or a device) to enable binding entity information to identities and thus access to the resources within that system by combining access rights and constraints with the established identity [23]. The definition also includes identity creation and management throughout the device life cycle.

According to the ITU-T NGN (Next Generation Network) identity management framework, IdM is defined as the "set of functions and capabilities (e.g. administration, management and maintenance, discovery, communication exchanges, correlation and binding, policy enforcement, authentication and assertions)" [24]. IdM primary purpose is to enable business and security applications and to provide assurance mechanisms of identity information that is the core of authentication and authorization schemes. According to the ITU-T NGN framework, typical functionalities of IdM framework include:

- Identity Life Cycle Management
- Identity Information Correlation and Binding
- Identity Information Authentication, Assurance, and Assertions
- Discovery and Exchange of Identity Information

Fulfilling these functionalities is achieved through three object identity components.

- **Identifiers:** an identifier is a series of digits, characters, and symbols used to identify a subject. For IoT devices, this can include Uniform Resource Identifier (URIs), RFID Tag ID (TID), IP addresses, and other identification schemes.

- **Credentials:** which is the set of data providing evidence for claims about parts of the objects' identities. Examples of a credential include digital certificates and access tokens.

- **Attributes:** which is the set of data that describes the characteristics of a subject. For IoT devices, attributes may include hardware specification, firmware version, power options, and other physical or logical attributes.

Identity management functions rely heavily on the device identifier that is used to address the device in all operations and services. Therefore, the text of next section extends and lists the major IoT device identification schemes.

2.2.1 IoT Devices Identification Schemes

Identification schemes for IoT devices are mainly driven by other identification specifications adopted by a number of organizations and standardization entities. These entities usually use their exclusive identification schemes. Most of the existing IoT devices utilize some of the these industrial identification schemes. Yet, there is no common identification scheme for IoT, which is attributed to the political and commercial drivers of the standard bodies and their sponsors [25].

Although there are various IoT identification schemes, these identification schemes are used in silos in different verticals. Thus, there is no single global unique identifier adopted in IoT, which makes interoperability challenging to achieve.

Ensuring interoperability is a key challenge in IoT due to the diverse set of hardware devices and heterogeneous IoT platforms. This issue is affecting services and applications that require device identity recognition prior to commencing any exchange of data [26].

The interoperability problem is aggravated when devices across different platforms start exchanging services and data, especially in peer-to-peer scenarios. Furthermore, IoT devices are also deployed in a mobile configuration where devices visit and connect to other networks that need to bind the device to its unique identifier in the home network. Among the major identification schemes used in IoT networks are:

IMEI

IMEI (International Mobile Equipment Identity) is a 15-digit number, each digit has a range of 0 to 9 coded as a binary coded decimal and is used to identify each device on mobile networks [27]. IMEIs are used to distinguish a mobile device among the billions of devices manufactured. The IMEI DB uses IMEIs to enforce operations related to combating devices' theft and counterfeit as devices can be labeled as either whitelisted, greylisted, or blacklisted. The access to theIMEI DB is offered to 3GPP (3rd Generation Partnership Project) members, including network operators, device manufacturers, and qualified industry parties. The IMEI identifier is defined by 3GPP TS 22.016 specification [28].

EPC

EPC (Electronic Product Code) is a universal identifier used as an identity for physical objects in supply chain. EPC specifications require that the id is unique amongst all objects in the system. It is main use is for tracking goods in supply chain systems that are part of the EPCglobal Network [29]. RFID tags are usually used interchangeably with EPC since most of the applications that uses EPC ids utilize RFID Tags as a carrier. In these applications, a specific Tag Data Standard (TDS) is describing the encoding of EPC codes onto RFID tags. The EPC standards are over sought by EPCglobal Inc^{TM}. EPC

identifiers can be represented either as a binary form or text form. The binary forms is used for storing in the RFID tags while the text form, also known as EPC URI, is used for identification in IT enterprise systems [29].

UUID

UUID (Universally Unique IDentifier) is class of identifiers that represents represents a 128-bit value. UUIDs are used to identify an entity or object uniquely in a system. The standard structure of the UUID includes 32 hexadecimal digits representing the sixteen octets of a UUID. Each UUID is displayed as 36 characters, four of which are hyphens, and decomposed into five groups of characters as 8-4-4-4-12. The Bluetooth Special Interest Group governs Bluetooth Low Eneregy (BLE) IoT devices uses UUID as the identification scheme and assignment of devices IDs (SIG) [30, 31, 25].

EUI

Extended Unique Identifiers (EUI), as per RFC2373 [32], are address formats specified by the IEEE for use in various layer-2 networks, e.g. ethernet. Two formats of EUI exist; 48-bit EUIs (EUI-48) for MAC addresses and 64-bit EUIs (EUI-64) for IPv6 addresses. In addition to Ethernet and IP-enabled IoT devices, EUI-64 is used by LoraWAN devices. LoraWAN is one of the IoT low power wide access connectivity protocols, and LoraWAN devices use DevEUI as a device ID following the IEEE EUI-64 address format [33].

The aforementioned existing IoT identification schemes share common proprieties in the current setup, as each has an assigned authority handling the policies and processes of identifiers assignments. Table 2.2 lists some of the existing mainstream schemes, including IMEI, Base-UUID, EUI-64 and EPC.

Table 2.1: List of Commonly Used IoT Devices Identification Schemes.

Scheme	**Registration Authority**	**Format**	**Devices Usage**
IMEI	GSMA appointed by 3GPP	15 digits - numeric	Smartphones and cellualr IoT Modems
EPC	EPCGlobal Initiative	Text format referred to as EPC URI (e.g. urn:epc:id:sgtin:061.012.628) and the binary format for storage inside physical objects (e.g. RFID)	Supply chain, RFID tags
EUI-48/EUI-64	IEEE	48 and 64 bits identifier - first 24 bits are the Organization Identifier and the rest is device specific	Etherent MAC and IPv6 addresses and LoraWAN Devices
Base-UUID	Bluetooth SIG	128-bits identifier divided into 5 sections with hyphens (e.g. 4A98xxxx-1CC4-E7C1-C757-F1267DD021E8) the x symbols represnt a 16bit unique manufacturer identifier	Bluetooth Devices (e.g. BLEs)

Many examples show how these identification schemes are utilized by different IoT frameworks and use them in an isolated way. An RFID tag that carries an EPC code is not identifiable in a LoraWAN network that is based solely on UUID and vice versa. Few organizations are working toward the abstraction of devices' identities since that is essential to build authentication and authorization frameworks that are agnostic to the other layers.

2.2.2 Identity and Access Management

A broader definition of IdM is IAM (Identity and Access Management), where in addition to identity management, the concept expands to include the description of the authentication and authorization flows. In this context, access is the set of privileges assigned to any entity for tasks to create or modify a data object or read it. Access is defined through a set of roles according to the entity attributes and authority.

In IoT, IAM is driven by factors related to the complex and heterogeneous architecture, change of ownership, and intermittent network association. Those factors require a framework that maintains a global and unique identity of the devices so that they can be transferred between networks and owners with less impact on its identification and optimally with no loss of track of the historical records.

In the IT and telecom domain, widely used IAM approaches include the Identity Relationship Management IRM, backed by the Kanatra Initiative [34]. While IRM defines interesting digital identity relations that apply to IoT devices, Yet particular challenges exist as it assumes a centralized approach that requires handling many relationship models that need to be built and maintained in case of IoT networks.

OAuth 2.0 and OpenID Connect also define two standardized frameworks for authenti-

cation and authorization [35, 36]. However, the main challenge is that OAuth and OpenID Connect have mainly been bound to HTTP protocol, which introduces a limitation in the case of IoT devices that usually due to the constraints in power and memory do not run on HTTP. New authentication and authorization schemes in IoT are needed for enabling devices to interact with each other in applications where devices exchange services in an autonomous and decentralized way. The Authentication and Authorization for Constrained Environments (ACE) is one of the standardized approaches that focus on authorization and authentication for constrained devices to ensure that miniature IoT devices can still maintain the required level of security [37].

Privileged identity management (PIM) is another IAM approach [38]. This approach also provides features of securely storage of the identity data and subject profile. PIM also offers functionality to control data governance which permits to specify how data is shared shared based on user privileges. Though while PIM works on reducing the service attack by ensuring that access is granted and assigned to a specific user, it concentrates and centralizes the access management policy and turns to be applicable only in enterprise IoT applications.

Though, due to the unique characteristics of IoT networks represented by heterogeneity, mobility, scarce resources, and interoperability, directly be applying the existing identity management systems in IoT networks cannot be satisfy the stringent requirements. These requirements are of more significant impact on the design of identity and access management systems for IoT, especially when considering the nature of applications and services offered by IoT devices and other concerns of privacy and security.

Some approaches opt for offloading those functionalities to gateways [39, 40], which does not fulfill the requirement of ensuring a more secure framework. Gateways vulnerability is addressed in many studies as a result of the shared usage of gateway representing a

single point for attack. One of the most notable examples is the "UDP hole punching" vulnerability, which is inherited in many IoT gateways and routers using the built-in features of the Universal Plug and Play (UPnP) specification [41].

This vulnerability allowed attackers to launch a man-in-the-middle attack by intercepting the connection between the devices, where attackers utilized the very primitive identification and authentication scheme used. The attack was reported to have affected more than 2 million IoT devices that connect in a p2p way, including security cameras, baby monitors, and smart doorbells [41].

Besides, the concentration of authorization rules verification at the gateways resulted in the usage of coarse-grained access control rules, which also contributes to more relaxed security measures and widens the surface attack. As an example, the IBM Watson platform defines two coarse-grained gateways access roles that govern the gateway ability to register devices to the Watson IoT Platform service, either standard or privileged [42].

2.3 The Distributed Ledger Technology

The distributed ledger technology (DLT), also known as Blockchain, has received in the last few years a lot of interest in different industries, especially in financial services. While the first application of Blockchain technology is cryptocurrencies, namely Bitcoin, over the past few years, an entire ecosystem of institutions and companies have developed different Blockchain applications. These applications featured new decentralized platforms that provide fast and secure means of handling domain-specific transactions; in health-care, postal service, IoT, supply chain, and others [43].

Blockchain is a distributed shared digital ledger that that adopts the concepts of public-

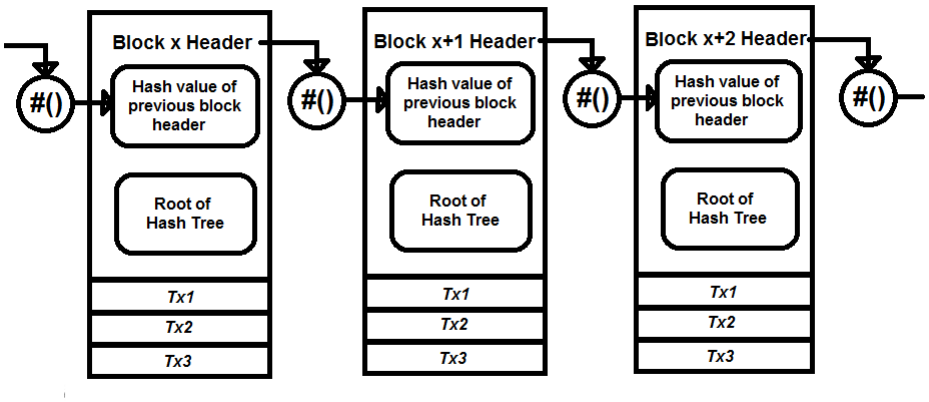

Figure 2.3: How Blockchain Works

key cryptography that provide the facility of pseudo-anonymity and identity. Blockchain operations maintain a distributed and immutable ledger through combining several transactions in a single block and hashing its content with the hash value of the previous block to construct the hash value of the new block, which turns the recorded transactions as irreversible and permanent. This mechanism enables all participants to verify the historical record down to the first block (the genesis block) [44].

2.3.1 The Chain Structure

Literally, Blockchain's name can be described by its two keywords; firstly that each (n) group of transactions form a single block, the second is the linkage of the blocks via cryptographic hashes into an immutable chain. This form of a cryptographically-based linked list makes it easy to verify the historical records up to the genesis block, and through this, it offers the feature of provenance [44].

The other intrinsic feature of Blockchain is its immutability, which makes recorded transactions irreversible and permanent. For example, the change in the balance of an

account is not done through updating a balance field. Instead, each new transaction combined with other transactions is used to form a block that will be validated and distributed as well as linked to the previous block.

The secure Blockchain workflow can be described in the following steps:

1. An external entity that owns a pair of private/public keys and a blockchain address, which is a short format of the public key, sends a transaction through a blockchain node.

2. The transaction is signed by the entity private key, and the blockchain will identify the entity through its address (public key) in a pseudo-anonymous way.

3. The transaction is validated, and with other transactions occurring in a period of time are ordered and grouped to form a block.

4. The generated block is used to challenge the mining nodes, the first to solve the challenge generates the so-called proof-of-work (PoW) and broadcast the block to the rest.

5. The rest of the networks confirm the validity of the transactions. If confirmed it inserts the block into the chain by linking it to the previous block, and all of this occurs according to the consensus algorithm.

6. A global image of the shared distributed ledger is updated, and in case of a financial transaction, the sender and recipient balances now reflects a change based on the value submitted in the transaction.

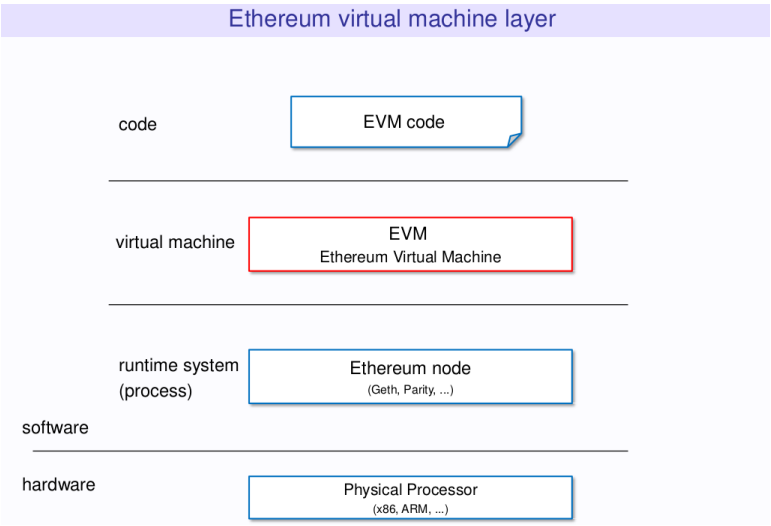

Figure 2.4: Ethereum Architecture

2.3.2 Ethereum Blockchain

Among the widely adopted Blockchain platforms is Ethereum, and it is a Blockchain architecture with an associated state database, capable of storing programs and their state. These programs are commonly referred to as Smart Contracts. A smart contract can be deployed by any Ethereum user, and it has a function-based interface or Application Binary Interface ABI. Once deployed, the smart contract can be referenced by its address, which is a cryptographic identifier. A user or an application interacts with a smart contract by sending a transaction to its address as the destination, and with the data payload of the transaction containing the function signature and input parameters. Calling a function causes the miners of the network to execute the program and update its state. A smart contract can hold and send the native value token Ether, and can furthermore call functions of other smart contracts [45].

Ethereum blockchain provides a "Turing complete" coding system, where theoretically

Table 2.2: Ethereum Smart Contract Terminology.

Term	Definition
Smart Contract	The code that is stored, verified and executed on a blockchain.
Contract Address	A blockchain address that is generated for each contract once created.
Constructor	A constructor is optional, and it is part of the code that executed once as the contract is created.
Function	The main part of the smart contract where the logic is built.
Modifier	A code used to change the behaviour of functions, for example to automatically check a condition prior to executing the function.
ABI	Application Binary Interface is the de facto method for encoding/decoding data into/out of the smart contract.

code can be put with any logic into an Ethereum smart contract, and it will be run by the whole network Ethereum Virtual Machine (EVM). The EVM computation capabilities are prevented from abuse, by allocating a token (Gas) against each computation, so computation power is paid for. The ecosystem of Ethereum blockchain-based solutions is made of the Ethereum network composed of number nodes (can be either public or private), smart contracts that are implemented in one of the supported languages (Serpent and Solidity) and user interface through other libraries [45].

2.3.3 Tokenization in Blockchain

Cryptocurrencies and tokens used in Blockchain applications represent a viable solution for establishing accounting and billing transactions. This trend has been largely attributed to the standardized and well-defined ERC20 token standard authored by Fabian Vogelsteller and Vitalik Buterin in 2015 [46]. In addition to ERC20 tokens, which is considered a

fungible (interchangeable) token, non-fungible tokens (NFTs) also started to become among the main assets exchanged on Blockchain platforms based on the ERC721 token standard [47]. That includes examples of game cards and other tradable collectible assets such as CryptoKitties, Decentraland and DotLicense [47].

The main difference between these two types of tokens is summarized by the fact that ERC20 tokens are the class of identical interchangeable tokens more of a fiat currency example, while ERC721 NFTs are the class of unique non-interchangeable token. The main feature of NFTs is that they represent ownership over digital or physical assets that include a diverse list of assets such as real-estate, unique artwork, virtual collectibles. This feature makes NFTs distinguishable and their ownership is trackable on an individual basis.

The thesis author demonstrates through the combination of these two token standards a foundation to enable a decentralized device access management framework. While the fungible tokens provide a simple, fast and frictionless transactional medium, NFTs offer the verifiable immutability and authenticity required in addition to other features such as delegation, transfer of ownership, and revocation.

2.3.4 Blockchain-Based Identity and Access Management Systems

A number of studies have proposed Blockchain-based IdM systems [48]. An Ethereum-based identity management solution was proposed in a work that defined few operations for decentralized public key infrastructure PKI . Those operations include adding attributes, signing attributes, and revoking signatures while running a performance test related to the Gas cost of the different operations [49]. Another identity management system based on Ethereum smart contracts was introduced by Liu et al., [50], building a virtual user

identity by binding a public key to user's entity information. The authors also defined a token to represent a reputation model for users.

In [51], Axon and Goldsmith introduced a privacy-preserving Blockchain-based decentralized PKI system. They proposed some IdM operations for identity registration and revocation while enhancing its privacy-preserving feature. Zero-Knowledge Proof based solutions also were studied, including the work of Hardjono and Pentland [52]. Their work is built on a permissioned blockchain setup and introduced a technique to anonymize the subject identity eliminating linking the entity to the performed transactions.

Besides, several companies are building decentralized blockchain-based identity systems, including Microsoft [53], Evernym [54], and others. For example, Chronicled Inc., is providing an IoT device registry. Their system uses encrypted microchips (mainly Bluetooth low Energy and NFC) to assign a secure digital identity to a physical object and linking it to a Blockchain record with a focus on the retail industry [55]. On the institutions and organizations level, Decentralized Identifiers (DIDs) maintained by W3C now considered as the main standard for decentralized identities [56]. Two principal streams for DID blockchain identity solutions exist permissionless and permissioned blockchain identity solutions.

A public blockchain identity platform is supported by Ethereum using ERC1056 and ERC 725/735 through Uport, one of the early startups in the Ethereum ecosystem [57]. Uport is building a decentralized identity solution for applications to solve the digital identity problem. Uport solution utilizes a set of smart contracts to provide decentralized identity management operations. On the other side, Sovrin Foundation and Linux Foundation, through the Hyperledger Indy project, offer a complete stack to manage DIDs and verifiable claims based on a selected and pre-authenticated validating and observer nodes [58]. Further discussion on DIDs follows in the next section.

2.4 Decentralized Identifiers and Verifiable Claims

2.4.1 Decentralized Identifiers

Decentralized identifier (DID) is a new paradigm for building global unique identifiers; those identifiers are built on top of distributed ledgers. DIDs are created as the cornerstone for decentralized digital identity and public key infrastructure (PKI) for the Internet. Among the main features is that DIDs offer full control of the DID subject by its owner without relying on a centralized authority.

The W3C Credentials Community Group maintains the DID specification, and those specifications describe the data model and syntax for decentralized identifiers [56]. DIDs specifications are designed to be compatible with any distributed ledger or network, where up to date there are more than 10 different Blockchain systems that support resolving DIDs. For example, the Ethereum Blockchain supports that through more than one implementation and among them the use of ERC1056 to enable a decentralized identity management system [59].

DIDs are defined by a generic format URL that is used to address DID documents and to be resolved by a DID resolver. DID generic format is a URI scheme conformant with the web standards as per RFC3986 [60]. DID is composed of three essential fields and can contain specific custom fields. The essential fields include the urn scheme in this case 'did' following that is the namespace or method name (e.g. ethr, sov ... etc), and following is the method-specific identifier which can be a combination of letters and numbers (e.g. "did:example:12345AbCDEfgh").

$$"did" : "method - name" : "method - specific - id" \tag{2.1}$$

2.4.2 DID Documents

In this decentralized scheme, DIDs are considered as a global key-value database where DID-compatible blockchains host the DID documents, where DIDs act as keys, and DID Documents are the values. DID Documents are constructed as JSON-LD objects describing the specified data model presented as a set of fields, including the public keys, authentication protocols, and service endpoints necessary to bootstrap cryptographically-verifiable interactions with the identified entity. These different fields are used either for proof purposes, verification methods, or describing service endpoints [56].

```
{
  "@context": "https://www.w3.org/ns/did/v1",
  "id": "did:example:123456789abcdefghi",
  "authentication": [{
    "id": "did:example:123456789abcdefghi#keys-1",
    "type": "RsaVerificationKey2018",
    "controller": "did:example:123456789abcdefghi",
    "publicKeyPem": "---BEGIN PUBLIC KEY...END---\r\n"
  }],
  "service":
  [{
    "id":"did:example:123456789abcdefghi#vcs",
    "type": "VerifiableCredentialService",
    "serviceEndpoint": "https://example.com/vc/"
  }]
}
```

According to W3C specification [56],the standard elements of DID Document include the following fields:

- **Context:** The value of this key MUST be the URL for the generic DID context: https://w3id.org/did/v1.
- **DID Subject:** The DID subject field contains a valid DID as the identifier of the DID Document.
- **Public Keys:** A DID document may include an array of public keys stored in different formats (e.g. PEM, base64, HEX). The public keys are used for authentication and authorization purposes as well as establishing secure communication with defined service endpoints.
- **Authentication:** It defines the mechanism used by a DID subject to prove that it is associated with a DID. This field is defined as an array of verification methods, where each method can be embedded or referenced to the listed public keys.
- **Authorization:** It is used to state the operations that can be performed on behalf of the DID subject. The most preferable approach is to utilize a capability-based approach to define details of authorization and delegation.
- **Service Endpoints**: This field specifies the methods of interaction with the DID subject to enable discovery and interaction.
- **Created/Updated:** These are optional fields to include a timestamp of the original creation and recent change used mainly for auditing.
- **Proof:** It is represented by a digital signature forming a cryptographic proof of the integrity of the DID Document.

The main feature of DIDs is their independence of a central party, the Identity Provider IdP. Instead, DIDs are managed by the identity owner or a guardian on behalf of the owner, which is known as self-sovereign identity (SSI). Finally, DIDs and DID Documents form the first layer of decentralized identity; however, the capability to tie DIDs to claims is achieved when combined with verifiable credentials.

2.4.3 Verifiable Credentials

A verifiable credential (VC) is the representation of physical credentials such as a driver's license, a university degree, or ownership of certain assets. The use of digital signatures makes verifiable credentials more tamper-resistant and credible than the physical credential. In practice, upon the request of a verifier, the holder of a verifiable credential can generate a presentation of this credential and share it to prove that the holder possesses this credential with the defined parameters [61].

According to the W3C specifications, specific fields are used to construct a verifiable credential object. Those fields include context field and identifier field which are URIs pointing to a readable and valid JSON-LD document about the field subject. The other field is the credential subject, which contains claims about the subject and referenced by the subject id, which is a DID. Another mandatory field is the issuer field that contains a URI (DID) pointing to a document containing information about the issuer identity. An issuance field based on ISO 8601 format is also required to express the time of issuing a verifiable credential. Furthermore, for a credential to be verifiable, a proof field represented by ,e.g., a digital signature, is required, where the method used for verification is defined in the type property [61].

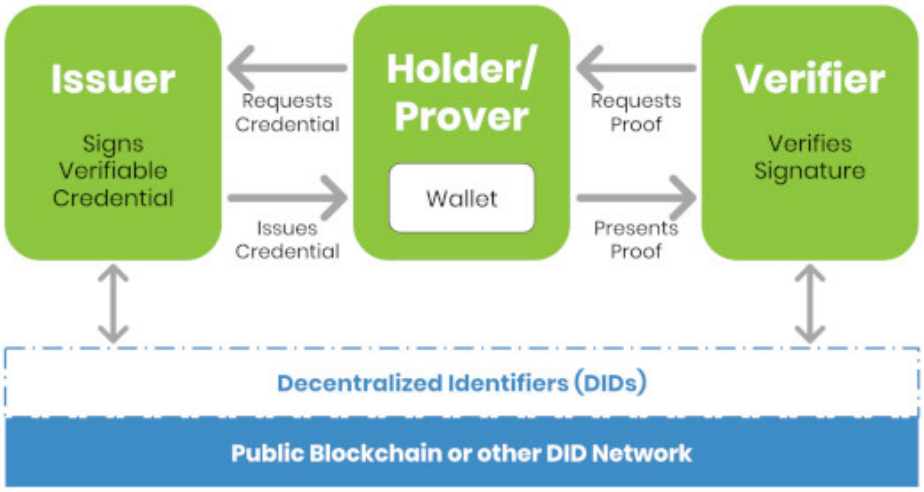

Figure 2.5: DID Infrastructure Roles

2.4.4 DID Infrastructure Roles

In a DID infrastructure, there are at least four roles need to exist [61], as follows:

- Holder — the individual or organization in control of the digital wallet used to control the use of a given credential. Important to note that the Holder may not always be the subject. The holder, in this case, is the one in control of the private keys associated with the subject DID (e.g. a phone may be the Subject of a digital identity, but the phone owner is Holder of that identity)
- Issuer — the individual or organization issuing credentials.
- Verifier — the individual or organization performing the verification of a given credential.
- Validators — the trusted parties validating identities stored on the distributed ledger.

2.4.5 Use of DIDs for IoT Devices Identity

Building a decentralized identity for IoT devices using DIDs has seen great interest in the last year. IoTex and Ockam have both proposed two DID methods did:io and did:ockam, respectively. Both methods are proposing the main identity operations with Ockam providing an SDK for users to register identities and submit and verify claims [62], [63]. The authors in [64] also suggest the use of DIDs and VCs to address how a centralized OAuth server is extended to support issuing DIDs and VCs as a delegated entity on behalf of constrained IoT devices. The authors also addressed an identified path for improving their work by combining DIDs and VCs to offer a granular and flexible method in managing access control policies and user authorization.

Interest from standardization bodies includes the recent work proposal that has been started by the International Telecommunication Union (ITU) to assess the feasibility of Decentralised Identifiers (DIDs) in IoT networks [65]. Also, The Linux Foundation and Hyperledger Blockchain project host two projects to define a decentralized identity management framework for IoT devices and possible use cases. Among these two projects is the IoT subgroup under the Telecom Special Interest Group (TSIG), in which the author of this thesis is participating in defining a decentralized IoT identity and access management specification [66].

2.5 Authentication, Authorization and Accounting

Authentication, authorization, and accounting (AAA) is the framework controlling access to devices resources, enforcing policies, auditing usage, and providing the information necessary for billing, which is centered around identity management service [67].

The first component of the AAA framework is authentication, which is the process by which a user or device identifies itself to another user/device. For example, a device can authenticate to a service or a device can authenticate to another device. The process of authentication is based on each user/device having a unique set of attributes (credentials) grouped as a unique identity. Authorization is the process pertaining to the means by which a device determines what privileges other devices/users have on the device and what actions they can perform. Thus, it defines the process of policy enforcement by identifying the types of activities, resources, or services a device/user is permitted to perform or access.

The third A is for accounting, and it considers the process of measuring resources usage during access time. This includes the amount of time or the amount of data a device has used during a session. The accounting function is mostly performed by logging of session statistics and usage information to be used for control, billing, and analysis.

In existing IT and Telecom networks, major frameworks coexist, including the RADIUS,

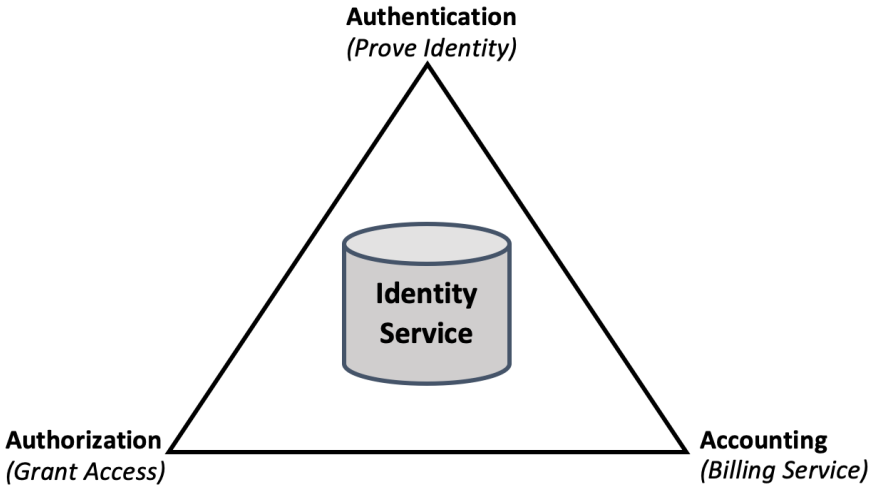

Figure 2.6: AAA Framework Components.

DIAMETER, and KERBEROS frameworks, to provide the AAA functionalities [67]. In IoT networks, and due to the need to support heterogeneous networks with intermittent connectivity and dynamic authentication/authorization relationships, an automated and autonomous framework for authentication/authentication with less centralization and support for scalability need to be devised.

Moreover, most of the existing frameworks are based on a central authority that controls the storage of all identities, performs the matching of authentication requests, and then provides access upon authorization requests. Those centralized systems represent a major single-point-of-failure condition, where an adversary that compromises the central AAA servers can disturb the authorization and policy management of the entire network. This becomes even more challenging when we consider peer-to-peer transactions that are characterized as a mainstream application domain in IoT networks.

Devices and data ownership are among the objectives of many regulatory frameworks such as the General Data Protection Regulation (GDPR) [68]. These regulations require granting users the ownership of the data their devices generate, which shows clearly that a decentralized approach would be a necessity to satisfy technical, business, and regulatory requirements.

Furthermore, analyzing further how current telecom operators and service providers run the AAA systems show that the centralized approach is limiting the granularity level of the AAA functionality required to enable per device or owner policy design and enforcement [69]. As per the current situation, an operator of a AAA service enables authorizing access to services or devices based on coarse-grained attributes such as IP address, port, or QoS level, and this does not offer fine-grained access control on resources and services deployed on IoT devices.

2.5.1 Access Control Models

An access control system is described by the set of rules defining the control functions of the who, what and when. Entities such as users or devices are normally called subjects, and a rule can determine what actions can be performed based on the rights granted on which resources, while resources usually called objects.

The different access control (AC) models include Access Control Lists (ACLs), Role-based Access Control (RBAC), Attribute-based Access Control (ABAC), and Capability-based Access Control (CapBAC). The most common form of access control is based on access control lists (ACLs), which is a centralized approach to support administrative activities, that assigns access rights to specific subjects. However, as the number of subjects and resources increases, confused duty problems are identified in ACL, and access rules become much more complex to manage [70].

To overcome the challenges of simple ACLs systems, the Role-Based Access Control (RBAC) approach [71, 72, 73] was designed. RBAC systems assign access rights to roles, and subjects are granted access only if they have been assigned the right role. RBAC supports access control principles such as least privilege authorization and partition of administrative functions. However, RBAC suffers from the roles explosion problem as the number of resources grows, which makes it unsuitable for implementing security policies that require interpreting complex and indistinct IoT scenarios.

Recently, capability-based access control CapBAC was proposed in few studies [70, 74, 75, 76]. CapBAC is based on issuing a communicable and unforgeable token that the owner can assign to other subjects. The core principle is that those capability tokens are issued by the object owner and granted to the subject with minimal interaction of a central authority, which makes it more aligned for decentralized architectures.

2.5.2 Access Control Models Suitability

Assessing the suitability of those different approaches to the IoT scenarios and specifically to the decentralized architecture proposed in this work is based on several factors. The work in [77] and [78] provide a solid analysis of the different approaches and highlight the major advantages and disadvantages which are compiled here in a radar chart to depict the coverage each approach is providing considering those multivariate factors, Figure 2.7.

The CapBAC model, as shown in the previous figure, demonstrates wider coverage in most aspects except the interoperability factor which defines the portability of any access control rule between different systems. This disadvantage, however, is enhanced in this work by utilizing a standardized token issuance model which provides a standardized interface on how these tokens can be consumed. The context-awareness aspect also ranks low, which can be improved by augmenting more contextual information in the capability token issued in a compromise of lightweight representation.

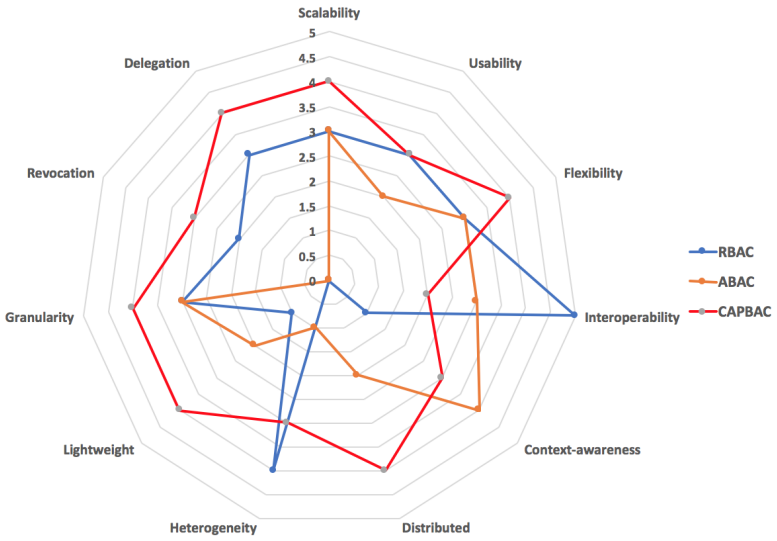

Figure 2.7: Radar View of AC Models fitness to AAA requirements in IoT Networks.

2.6 Summary

This chapter has discussed the fundamentals of IoT networks and the challenges of managing a large number of devices while ensuring security and privacy. The IAM concepts in IoT were highlighted with elaboration on the commonly used identification schemes. The chapter also discussed the leading technologies and frameworks proposed in the domain of decentralization, including Blockchain, smart contracts, and decentralized identifiers.

Based on the study of state-of-the-art, it can be noticed that to secure IoT devices and enable peer-to-peer applications a foundation layer for decentralization is needed. The foundation layer of decentralized global and unique device identity is considered the cornerstone to building other security schemes for authentication and authorization as well as providing more control of users on their IoT devices and the generated data. In the next chapter, an IoT device identity framework will be proposed based on the DIDs specification to provide a framework to manage IoT devices identities in a decentralized approach.

Chapter 3

Decentralized IoT Identity and Access Management Framework

The need for identity management is essential in all IT enterprise systems to handle the authentication and authorization flows. Moreover, in IoT, the demand is driven further by factors related to the high number of devices, complex architecture, change of ownership, and intermittent network association. Those factors require building a global and unique identity of the devices so that they can be transferred across networks and ownership changed among users with less impact on the devices' identification and optimally with no loss of track of the historical records.

In this work, identity is treated as a living asset, such that a device is given an identity at birth (e.g., manufacturing). The device maintains its identity with updates and versions as pre-defined changes happen, which is maintained in a global registry to enable verification, tracking, and auditing. According to the ISO/IEC 24760-1, "an entity is an item inside or outside an information and communication technology system, such as a person, an

organization, a device, a subsystem, or a group of such items that has recognizably distinct existence", while identity is the "set of attributes related to an entity" [79].

The suggested approach for a decentralized IoT devices identity and access framework is based on the concept of Decentralized Identifiers DIDs and the use of smart contracts for rule processing automation. Earlier work by the author was based on proprietary DID definitions [80]. This work builds a DID method for IoT devices that is used for defining devices' global and unique identities, which are used as the base for devices authentication. For access control, decentralized access rules are defined as tokens with engraved capabilities that translate actors given privileges.

This chapter first highlights the framework stack and continues by discussing the decentralized identity model, then the proposed identity management layer architecture listing the different components. Next, the proposed DID method for IoT devices is described and then highlighting how the IoT DIDs are utilized in the authentication process. Authorization and access control are discussed in Chapter 4.

3.1 Framework Layers

A recent trend aims to build decentralized architecture for IoT networks, which is considered very important in the service layer. Achieving that requires that the underlying functionalities represented by the identity, authentication, and authorization management should be designed in support of this scheme. This work suggests a decentralized identity and access management framework that provides global and unique devices identities that facilitates devices to authenticate each other in a peer-to-peer fashion. The framework utilizes the Blockchain tokenization concepts to facilitate the authorization and accounting flows with no middle man.

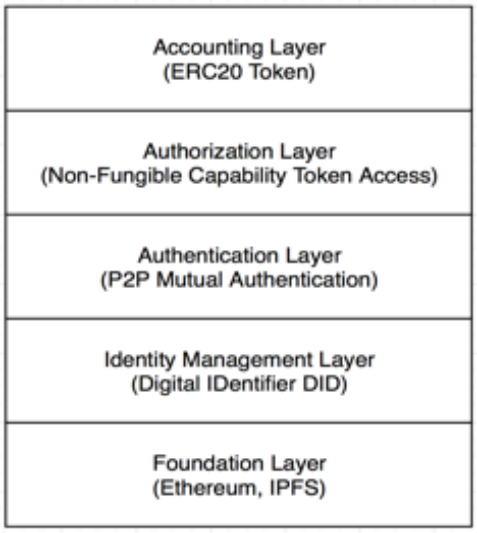

Figure 3.1: Framework Stack.

The proposed decentralized framework is based on the distributed ledger and smart contracts technology, and composed of the following layers:

- the foundation layer is the Ethereum Blockchain network hosting the different smart contracts and providing the built-in features of immutability, provenance, and cryptography. Also, this layer uses the decentralized storage service (InterPlanetary File System - IPFS) to enable global access to identity information [81].
- Decentralized identity management layer is the core element for enabling decentralization in higher layers. This layer facilitates creating decentralized services in other layers by defining a DID IoT method with the necessary smart contract(s) to govern the process flow.
- The authentication layer relies on having devices' identities deployed and accessible in a decentralized way to allow devices to perform mutual authentication with no middle man.

- The authorization layer provides decentralized access control, which is realized by representing access rules as tokens on Blockchain, while these tokens are issued and managed by devices and exchanged in p2p fashion.
- Accounting layer is also another layer utilizing Blockchain tokens (ERC20 tokens) to credit devices for services offered and exchanged.

The next sections provide the details about the first three layers with considerable focus on the identity management layer while the authorization and accounting layers are discussed in the next chapter.

3.2 Foundation Layer

The foundation layer provides implicitly the infrastructure required for cryptography, business rules automation, virtual machine, and storage. The Ethereum Blockchain enables the tracking of all transactions and ensures the existence of the immutable records, which are required for tracking and auditing purposes. The distributed file storage system represented here by InterPlanetary File System (IPFS) [81]. IPFS provides the distributed storage infrastructure necessary to ensure that IoT devices and the generated data have the required portability and that it is offered in a provider-agnostic approach with the capability to be extended to support data protection and a traceability mechanism [82].

The two platforms, Ethereum and IPFS, are used to complement each other. In avoiding storing large data volumes on the Blockchain, where data, including identities attributes and access policies metadata, are stored on IPFS and the hash value of the stored data is linked to the transaction stored in the Blockchain to ensure the integrity of off-chain data.

While it is known that certain security issues may affect the use of IPFS for storing privacy-sensitive applications, this is not the case in this proposed framework. The intended use of IPFS in this thesis is for storing the DID Documents which are to be made accessible to any device to resolve a DID identifier into the corresponding document.

3.3 Decentralized Device Identity Layer

3.3.1 Identity Management Architecture

In general, Identity Management (IdM) systems are based on an architecture represented by three independent parties: the subject (a device or a user), service provider (SP), and Identity Provider (IdP) [83]. The IdP is the party responsible for managing identity information, including identifiers, credentials, and attributes. In a traditional IdM, IdPs have the central role of allowing users/devices to roam across different SPs and obtain access to offered services.

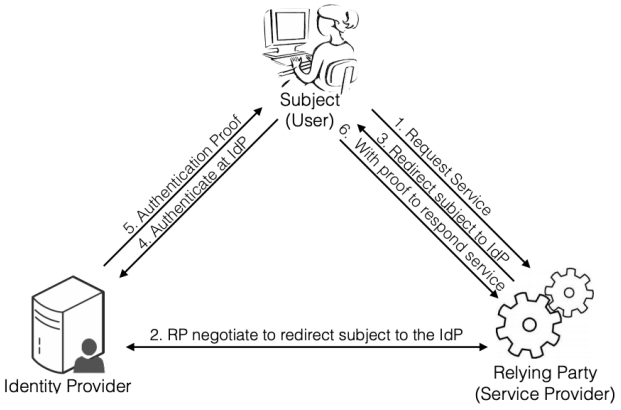

Figure 3.2: Traditional IdM Architecture and Flow of Interaction.

A simple illustration of the main steps involved is depicted in Figure 3.2. The flow is initiated by a subject requesting access to a service, where the SP responds by redirecting the subject to an IdP that handles the authentication process. Upon successfully authenticating the subject, the subject re-issue the access request with a proof of identification. Ultimately, the access is granted based on a trust mechanism that in this case is built based on the centralized third-party IdP.

The proposed decentralized architecture adopts a similar flow of interaction with the elimination of the middle man (IdP) to enable a decentralized global identity register with self-governing over identity management functionalities. In the proposed architecture, Figure 3.3, the IdP is replaced by two components:

- **Identity Manager Smart Contract:** a smart contract handling the identity registration and management interactions acting as an identity anchor and utilizes DIDs

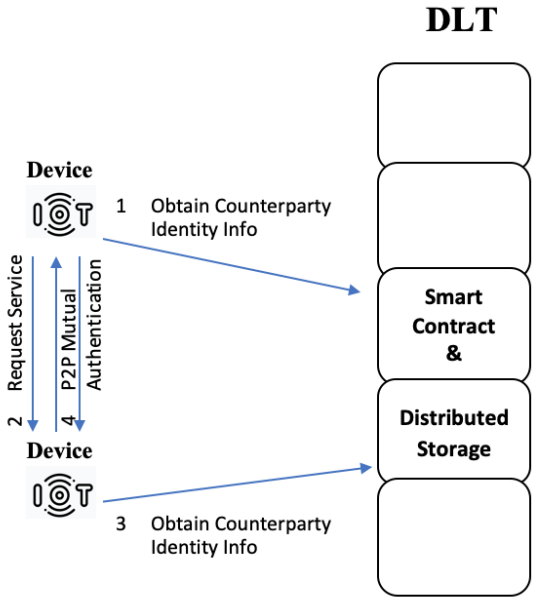

Figure 3.3: Decentralized IdM Architecture and Flow of Interaction.

as the unique primary identifier to ensure a global and unique device identity

- **Decentralized Identity Hub:** A DID documents hub to store identity information and enable querying and verification of identities' attributes.

3.3.2 Identity Relationship Modelling

It was discussed in Chapter 2 that the usage of traditional device management frameworks in IoT lacks a unified and global identification scheme that deals with the issues of heterogeneity and the need for identity interoperability and portability. Thus, by proposing a decentralized identity management framework, this work describes an identity relationship model that defines the different components and maps their inter-relationships. The discussion examines the device identity life cycle and the device-identity relation and how it affirms identity uniqueness.

Device Identity Life Cycle

In the IoT networks, a device will go through a life cycle starting from manufacturing until the end of life. The proposed decentralized IdM architecture depicts a cycle that defines the device status and its ownership. In this life cycle, the device identity is required to be maintained through a global registry that permits authorized registrants (e.g., manufacturers) to create new identities for new devices. It allows the registrant to either perform identity management functions or transfer ownership of the device(s) to other owners, e.g., operators and end-users.

The device identity is maintained and updated through the different phases, where a new version, if required, is issued as the device's owner changes or the device status on

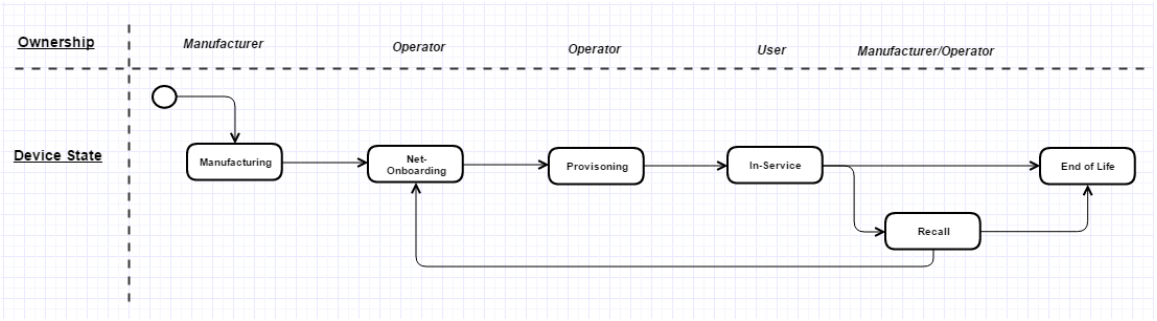

Figure 3.4: IoT Device Lifecycle and Ownership.

the network changes from provisioning to in-service to retire while maintaining a unique global identifier, as illustrated in Figure 3.4.

According to the device life cycle, the ownership and service consumption relationship between an IoT device, the owner, and the different consumers frequently change over a device's lifetime. Thus, the framework asserts that a device, the Subject (S), should have a specified Owner (O) at any point of time (t), and there exist (n) number of Consumers (CS) to which a device is providing services.

$$S \ni O, < CS_1, CS_2, ...CS_n > \tag{3.1}$$

From a service point of view, a subject that is providing a service is considered a Service Provider (SP), and the consumers can be either users, systems, or other devices (subjects). Thus, a subject represented by a device can act as both a service provider or a consumer.

Subject-Identity Relationship

The proposed relationship model between subjects (devices), identities, and attributes is built on a one-to-one relationship between the subject and its identity and one-to-many

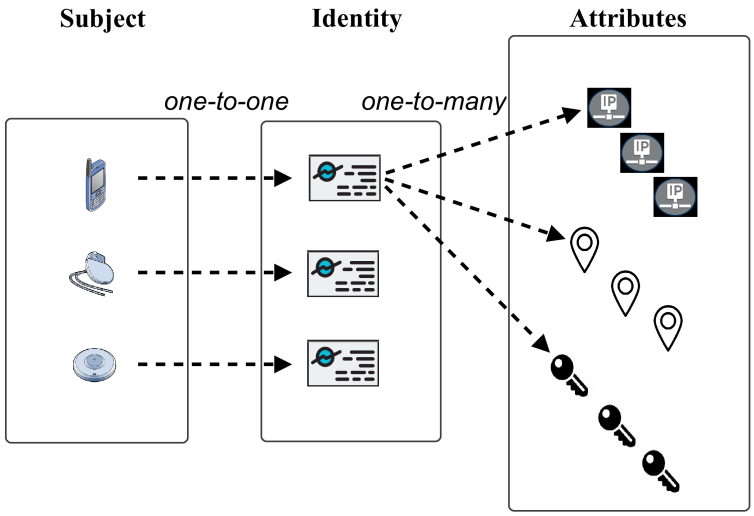

Figure 3.5: Subject-Identity-Attributes Relationship Model.

relationship between an identity and the composing attributes and credentials, as shown in Figure 3.5. In addition, a global unique identifier is essential, and can be anonymous.

Thus, for a network (NET) with (n) subjects (S), there exists (n) unique decentralized identities (DID) each with a unique identifier (ID), where the identity is defined by a set of attributes (A) and credentials (Cr) as defined in Equation. 3.2.

$$S_i \ni DID_i,\ s.t.\ DID_i = (ID_i, < A_{i1}, A_{i2}, \ldots, A_{in} >, < Cr_{i1}, Cr_{i2}, \ldots, Cr_{im} >) \quad (3.2)$$

Uniqueness of Identity

The proposed framework asserts that each device in any network should maintain a global and unique identity. Therefore, for (NET_i) that represents any network among K IoT

networks, each with (n) number of devices/subjects (S).

$$NET_i = \{\ S_{i1}, S_{i2}, \ldots, S_{in}\}\ ; i = 1, K \tag{3.3}$$

Where each subject S_{ij} holds a unique DID_{ij},in different networks of size l and m

$$\forall\, S_{ij} \in NET_i,\ \exists\, DID_{ij},\ s.t. \nexists\, DID_{lm} \equiv DID_{ij},\ i \neq l \vee j \neq m \tag{3.4}$$

A high-level view on different interactions between subjects is depicted in Figure 3.6, which shows how any device (S_{ij}) can have the role of either a service provider or a consumer, and the relationships to owners and other consumers.

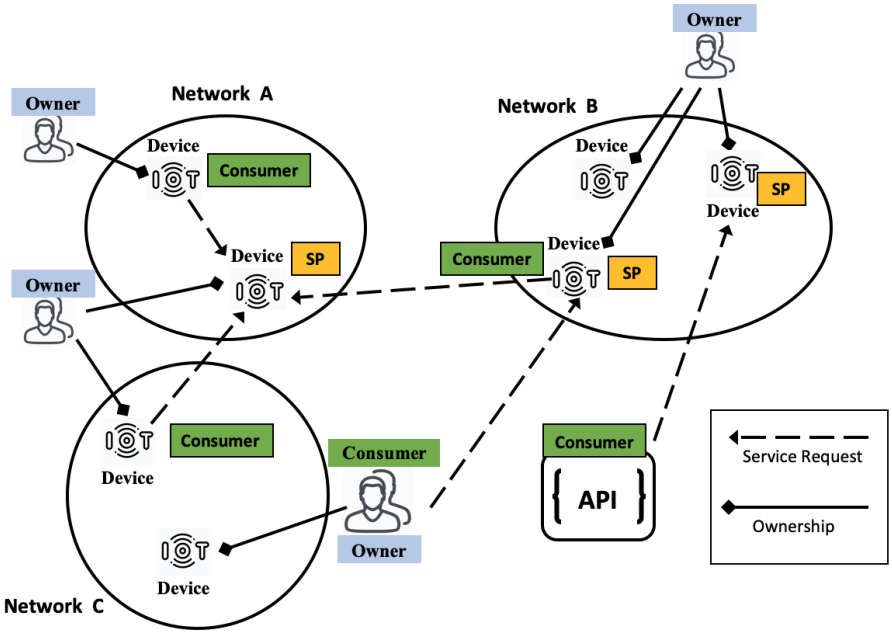

Figure 3.6: High Level View on Subjects Relationships.

3.4 Decentralized IdM Components Description

The essence of the proposed identity management layer architecture is based on the main concepts of Blockchain. These concepts are used to match the objectives of establishing a global registry with a unique and global identifier for each device, a higher degree of anonymity, immutable records that are traceable to the genesis of the identity creation, and with minimal third party authority.

The suggested architecture is based on three components: identity manager smart contract running on Ethereum Blockchain, a DID Hub implemented on IPFS network, and a DID resolver and a newly proposed DID method specific to IoT devices, as in Figure 3.7.

- The IdM smart contract is implemented on Blockchain and is used to build the global registry with specific rules to enable authorized and identifiable entities of the creation of the device identity. The public key cryptography is used as the means to ensure establishing a global and unique identifier for each device if no identification scheme exists or when anonymity is required. The created DID identifier is anchored

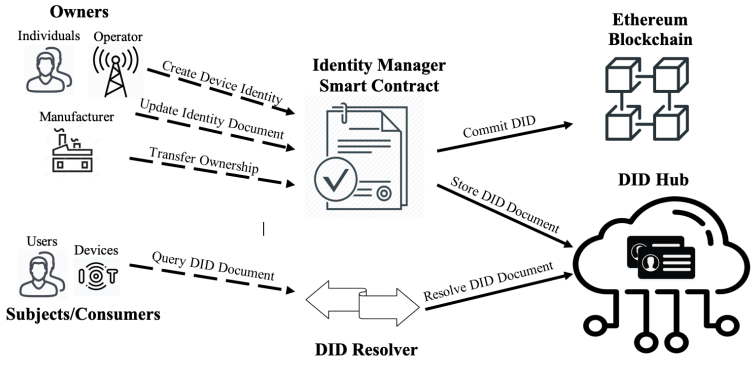

Figure 3.7: Identity Management Layer Components and Actions Flow.

in the smart contract to ensure maintaining an immutable record.

- The DID Hub is the primary storage component for the created DIDs documents. The DID Hub can be hosted on any storage service, and in this proposal, the choice was the IPFS network to guarantee a decentralized storage option.

- A DID Resolver is essential to enable resolving a DID identifier into the corresponding DID document stored in the DID Hub. Both the identity manager smart contract and the DID resolver implement the supported identity operations of the proposed DID method.

- The DID method is not a specific component, yet it is the syntax and definitions of DID identifiers and documents format in addition to the definitions of different methods supports. The different actions in Figure 3.7 are translated to DID method operations.

3.4.1 Actions Flow

As indicated above, each device as a subject can perform different actions, and so are device owners. The actions flow is depicted in Figure 3.7, where actions for creation, updating, or transfer of a DID pass through the smart contract to commit and modify the records in Blockchain and update the corresponding Document in the DID Hub. Querying is performed through the DID resolver that uses the DID identifier as an index to retrieve the corresponding DID Document.

Identity creation

Identity creation is based on public-key cryptography and hashing functions. The identity is composed of the minimal set of attributes, $< A_{i1}, A_{i2}, \ldots, A_{in} >$, necessary to generate a unique and global identity for the device in the global registry. That is performed in two steps: generation of cryptographic keys and Blockchain address and the formation of the Blockchain-based digital identity. The decentralized identity approach is suggested to either build anonymous identifiers using the cryptographic keys or use the device technology unique identifier (e.g., IMEI or UUID) as the DID identifier.

In Blockchain, specifically in Ethereum, each account (e.g., device) first obtains a pair of private and public keys, from which a Blockchain address is derived based on the result of the hashing function of the public key. A reduced version of the details of the generation of the key pair and Ethereum address, as per [45], is given below:

1. An initial seed vector (IV) is used with a key derivation function such as PBKDF2-SHA256 to generate a 256-bit (32 bytes) private key (SK).

2. The 512-bit (64 bytes) public key (PK) is generated from that secret key (SK) using the elliptic curve digital signature algorithm (ECDSA).

3. The Public Key is hashed with SHA-3 (Keccak) to produce a 256-bit output, where the upper 96 bits are discarded, and the lower 160 bits (bits 96 to 255) become the Blockchain Address.

$$Addr(SK) = \beta_{96..255}(KEC(ECDSAPUBKEY(SK))) \tag{3.5}$$

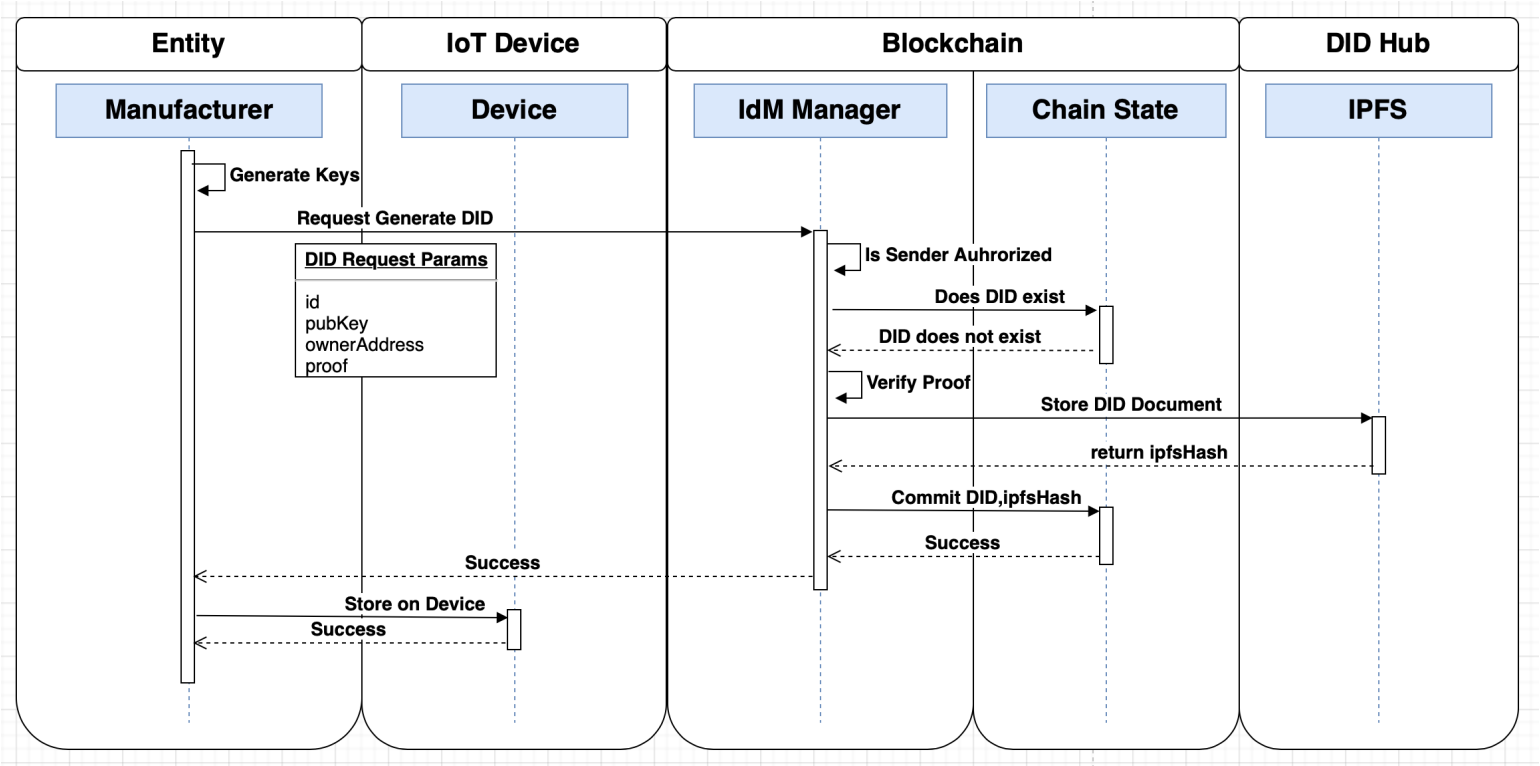

Figure 3.8: Identity Creation and Registration Sequence Diagram.

The identity creation is performed according to the procedure defined hereafter, where the owner issues a request to the IdM smart contract to generate a new DID. Upon success, the contract commits the DID record on-chain and stores the DID Document on the hub.

Algorithm 1: Identity Creation

```
Input: id, pubKey, owner, proof, hash
Result: (DID) Committed on Chain
procedure GenerateDID(id, pubKey, owner, proof);
if IsSenderAuthorized then
    DID ⇐ CreateDIDIdentifier(id);
    if DID not exists then
        valid ⇐ verifyProof(pubKey, proof);
        if valid then
            ipfsHash ⇐ storeIPFS(DIDDocument);
            ListDIDs[DID] ⇐ ipfsHash;
        else
            Ignore;
        end
    else
        Return('IdentityAlreadyExists');
    end
else
    Ignore;
end
```

Transfer of Ownership

According to the IoT Security Foundation (IoTSF) [84], it is inevitable to limit IoT devices from changing the ownership throughout their lifetime, and among the essential tasks in device management is to preserve the auditing trail, the security of the device and its

reputation profile throughout the life cycle. Thus, as devices are transferred from one owner to another, the proposed identity management framework defines the flow highlighted in Figure 3.10 to ensure:

- Sustaining a device's global and unique identity by maintaining the same DID identifier.
- Check the current ownership before performing the transfer of ownership action to disallow unauthorized actions.
- The change of ownership is performed by changing the owner identifier in the DID Document then request the new owner's signature for the updated DID document.
- Commit the update on Blockchain to maintain the auditing trail.

The transfer of ownership process is managed through the identity management smart contract that implements the logic of the IOT DID method. For a dev_i and owner O_j, the owner issues a request of transfer to the smart contract defined ABI. Based on the content of the identity parameters, the smart contract's specific modifiers authenticate the source of the origin (ensure issuer is the owner) and the authority it has.

Upon successful authentication and authorization, the identity attributes get updated; specifically, the owner related fields are now reflecting the new owner public key with a signature proof for the integrity check. The new version of the identity document is stored in the identity hub, and the DID record on-chain is updated with the new hash maintaining the trail of all identity updates.

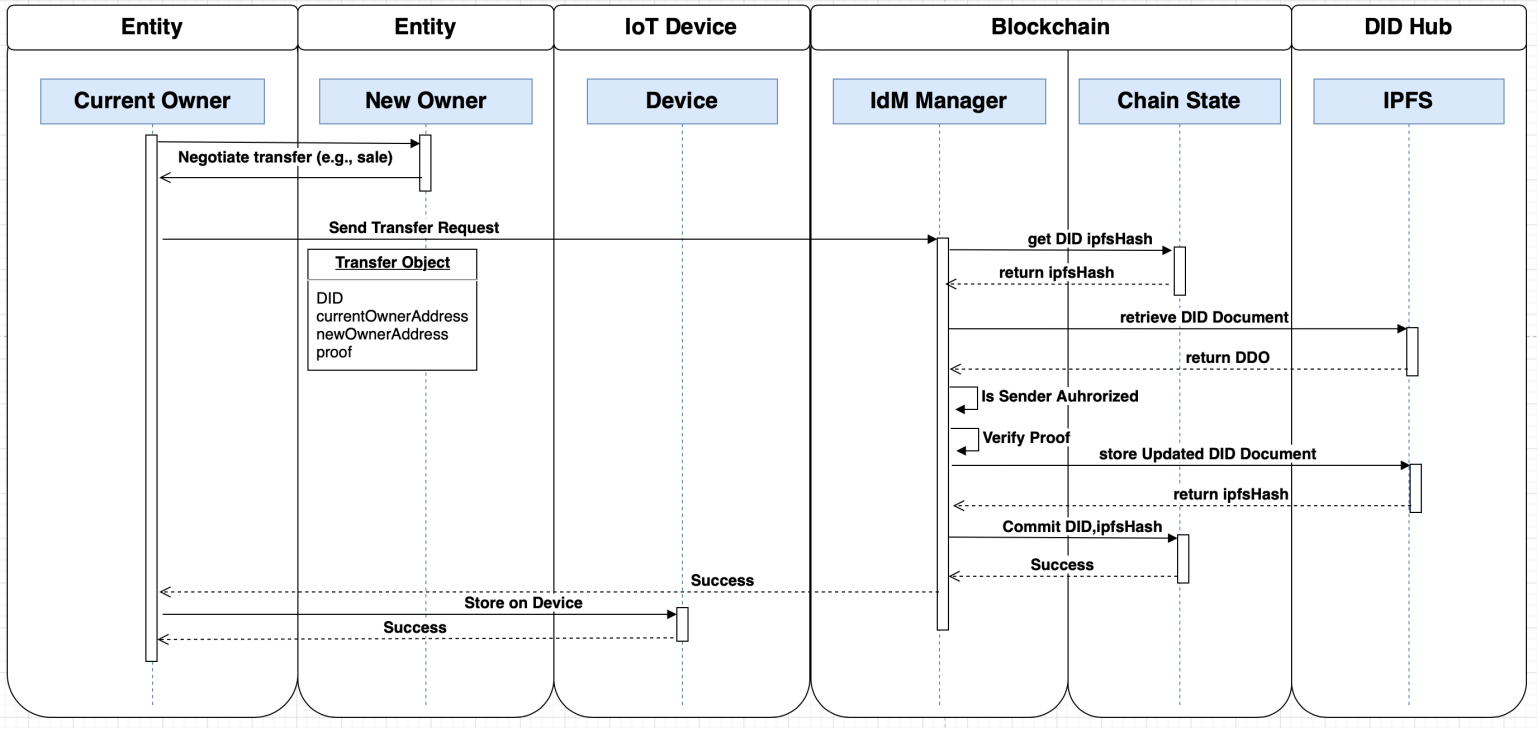

Figure 3.9: Identity Transfer of Ownership Sequence Diagram.

3.4.2 IOT DID Method

Enabling decentralized identity management for IoT devices is based in this proposal on the use of DID specifications and the proposed IoT Devices DID method. This method defines how the framework suggests creating DIDs and DID documents. The introduced method is designed to conform to the DID specification requirements [56].

Method Name and Identifiers Syntax

The proposed method name that shall identify this DID method is *iotdev*, and any created DID that uses the proposed method must start with the following prefix *did:iotdev* in lowercase format. A DID based on *iotdev* method should have the following format:

```
did-iotdev = "did:iotdev:" iotdev-identifier
iotdev-identifier = *(idtype ":") idstring
idtype = "imei" / "epc" / "uuid" / "eui"
idstring = ALPHA / DIGIT
```

The proposed method and identifier syntax supports different IoT devices identification schemes where the identifier has an optional *idtype* path that currently supports IMEI, UUID, EUI, and EPC identifiers. An anonymous DID identifier is also supported as the default *iotdev-identifier*, which is the Ethereum address of the device derived as in Section 3.4.1. The different examples of *iotdev* identifiers are:

```
did:iotdev:0xb217366aDe3f65E8D2A79f0Da1B607Aa0E399f56   // Default
did:iotdev:imei:351756051523999 // IMEI number
did:iotdev:uuid:123e4567e89b12d3a456426655440000  // UUID no hyphens
did:iotdev:eui:00152BFFFEE49B60  // EUI no colons
did:iotdev:epc:06141410000734314159 // EPC no dots
```

DID Document Format

The objective of using DIDs and DID Documents is to enable a global, unique, and immutable identifier and to provide the utility for performing mutual authentication and verification of devices identities with no middle man. The DIDs are also offering a utility to provide the capability of service discovery. The IOT DID method proposed document format is composed of the attributes derived from the W3C specifications as in Table 3.1

Method Operations

The IOT DID method supports the default CRUD (Create, Read, Update, Deactivate) operations. In addition, it provides two extended methods: transfer of ownership and update delegate.

- **Create** The DID *Create* operation is implemented through the identity manager smart contract to offer to register any new device DID. The create process establishes the unique decentralized identifier based either on the technology specified identifier or by creating an Ethereum address for the device. The initial device owner registers a DID and constructs the DID Document providing a cryptographically signed proof as one of the document attributes.

  The DID document is constructed offline and appended by the proof, so it does not require submitting a transaction to the Blockchain except when the DID identifier and DID document hash are recorded on the Blockchain through the smart contract. Storing only the DID identifier and document hash on the Blockchain limits the storage capacity on Blockchain. It instead utilizes the decentralized storage of the IPFS network for storage. The authenticity of the document is ensured through the hash value, which is used to map the DID to the stored document on IPFS.

Table 3.1: IOT DID Method Document Format

Attribute	**Definition**
@context	Contains the default URI schema, 'https://www.w3.org/ns/did/v1'.
id	Defines the subject identifier as per the method definition.
created	A timestamp for the time when the DID document is first created.
updated	A timestamp that gets changed every time the DID document is updated.
revoked	An attribute that exists in a DID document only if the identity has been revoked, and in that case, the timestamp indicates when revocation took place.
publicKey	An array of different public keys of different stakeholders related to the DID. This attribute should include at least two public key one tagged with the fragment (#owner), and the other one is the device public key tagged with the fragment (#device) used by the device for cryptography operations.
authentication	An array of verification methods and again at least one entry should exist linked to the owner public key.
proof	A digital signature used to check the integrity and verify the authenticity of the document stored off-chain. The proof should include four items: the signature method used, a timestamp, the public key of the issuer, and the signature itself.
service	An array listing the different services offered by the device linked to this identity, in this work this attribute is used for advertising the authorization smart contract used for access control and enable other devices to discover it.
recovery	An optional attribute that is introduced to mitigate for key loss where an owner can be encouraged in the business process to generate a recovery key used to manage the DID in case of loss of primary owner key.

- **Read** An IoT DID is resolved to a DID document that is stored on IPFS in the *Read* operation. The DID resolver provides an API endpoint to query a DID identifier and obtain its corresponding hash and then use the hash to retrieve the document object.

- **Update** The *Update* operation is considered in the proposed method to perform an update of any of the fields of publicKey, authentication, or service and to add any new attributes. Updating a DID document is issued as a transaction to identity manager smart contract to check two conditions:

  - If the issuer of the transaction is the owner or an authorized delegate.
  - The signature on the proof attribute is valid and verified against the issuer public key.

  Upon issuing the updating process of a DID document, a new proof is created, and a new DID Document is stored in the IPFS DID Hub. The newly created IPFS document will generate a new hash that is used to update the DID record in the smart contract.

- **Transfer (of Ownership):** The *Transfer* operation is a subset of the update operation where it updates the publicKey attribute defined by the owner fragment. This operation is treated separately to support a different event for the transfer of ownership. Transferring a DID ownership requires that the transaction passes the conditions of ensuring that the owner has issued the transaction and that the new proof signature is valid and verified.

- **Delegate** The *Delegate* operation is also a subset of the update operation where it updates the delegate attribute, either to add, update or delete a delegate. Once

more, the same conditions as in the update operations should apply for the operation to be successful.

- **Revoke/Deactivate** The *Revoke* operation is issued when a device is retired, and in this operation, the DID Document is updated to include the revoked attribute, which contains a timestamp.

3.5 Devices Authentication

The authentication layer in the proposed framework provides a peer-to-peer mutual authentication mechanism based on the decentralized device identity. The fundamental component of this layer is defined by the mutual challenge-response authentication mechanism [85]. The following assumptions are made about any two devices entering into the authentication process:

1. Any device has a secure channel to the Smart Contract ABI and the DID Hub.
2. The devices can generate random numbers with enough entropy.
3. The devices are synchronized.
4. The DID document authenticity is ensured through the identity management layer.

The authentication mechanism starts when device 1 wants to establish a secure connection with device 2, and both devices (Dev_1 and Dev_2) want to confirm each other identity and establish a session key to be used for cryptography operations.

Firstly, Dev_1 sends a query to the DID resolver to obtain the DID Document for device 2 DDO_2 which contains the public key for device 2 PK_2. To mitigate for man-in-the-middle and replay attacks Dev_1 generates a random number r_1 and a timestamp t_{s1} and

submits a challenge composed of an encrypted message of the concatenation of id_1, r_1, and t_{s1} using Dev_2 public key.

Upon receiving the challenge by Dev_2, the challenge is decrypted using Dev_2 private key SK_2, checking the timestamp t_{s1} against the current time to ensure that it contains a recently issued challenge. Dev_2 extracts Dev_1 DID id and retrieves the its DID Document DDO_1 and thus the public key PK_1. Dev_2 generates a random number r_2 and a timestamp t_{s2} and sends back an encrypted response message composed of the concatenation of id_2, r_1, r_2, t_{s2}.

As Dev_1 receives the original challenge r_1 this proves Dev_2 knowledge of the private key and thus that DID_2 belongs to it. At this point, a one-way authentication is achieved. The third message is required, such that Dev_1 returns an encrypted message containing the challenge r_2.The two devices now both have knowledge of the challenges r_1 and r_2 and proved ownership of the private keys of the DIDs retrieved from the DID Hub and thus are both mutually authenticated.

For establishing the session encryption key (K_s), both devices use Hashed Message Authentication Code (HMAC)-based Key Derivation Function (HKDF) to generate the secret session key that is used for further communication. The key generation function is based on SHA-256 hashing function with 256 bits key length (key_l) according to the NIST recommendations on the usage of key derivation functions [86]. The HKDF uses the concatenation of the two challenges as an initial string of bits and a salt of length identical to the hashing function width, as in Equation 3.6.

$$K_s = HKDF(r_1||r_2, salt, key_l) \tag{3.6}$$

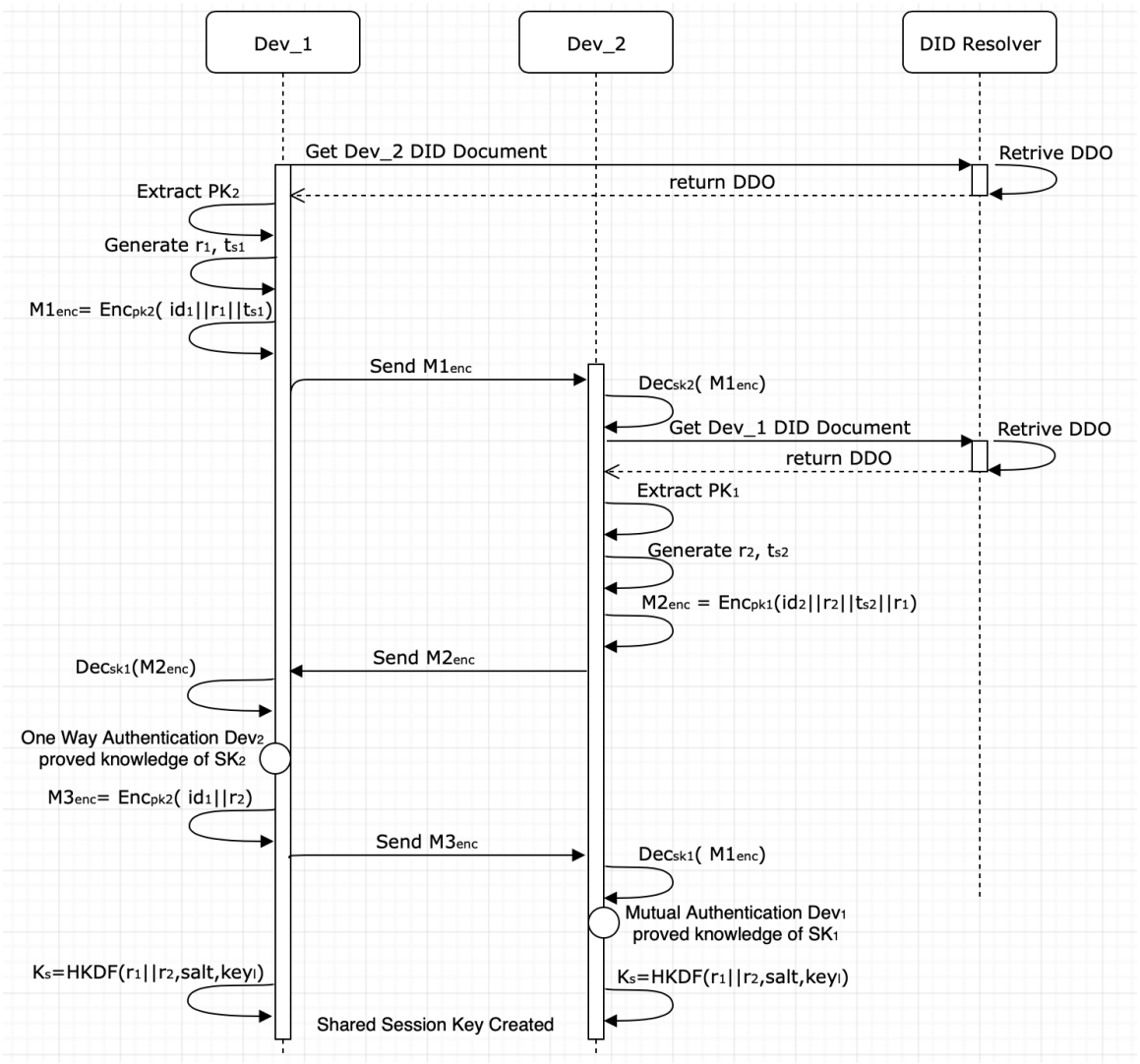

Figure 3.10: P2P IoT Device Mutual Authentication Sequence Diagram.

3.5.1 Device Authentication Attacks Analysis

The strength of the proposed authentication scheme is based on the strength of the used encryption protocol. To provide analysis on the immunity of the proposed authentication and key derivation scheme the next items discuss certain possible attacks.

Impersonation Attack

In this type of attack, an adversary E can impersonate any of the two devices (Dev_1 or Dev_2) by claiming the identity of one of them. For such attack to be successful (E impersonating Dev_1) the adversary must be able to act on behalf of Dev_1. E has the knowledge of both DID_1 and DID_2, so E can query the DID Resolver and obtain DID_2 DID Document to extract PK_2.

For the adversary E to be able to impersonate DID_1 it has to have knowledge of DID_1 private key to be able to send an encrypted message that Dev_2 can verify as it was generated by Dev_1 private key.

One way for E to achieve forging the authentication messages exchanged is by compromising the private key of one of the devices which is not achievable if the encryption protocol is secure. In that case, E is not able to forge the authentication messages even by observing a large number of exchanged messages.

Replay Attack

The other way E can try to forge the authentication messages is by re-using a previously sent authentication message, which is known as a replay attack. However, since the authentication scheme uses a random number and a timestamp this renders a previous exchanges

message invalid by checking the validity of the timestamp. Furthermore, the re-use of a previously exchanged message does not disclose to E the random number used for key derivation and this limits E from establishing the session key.

Reflection Attack

In this attack, the adversary E acts in between both devices acting as Dev_1 for Dev_2 and as Dev_2 for Dev_1. E starts the sequence by sending a valid M1 message using E private key, and Dev_2 sees this as a legitimate message and exchanges back M2 with E. The adversary now generates message M1 directed towards Dev_1 and obtains a response from Dev_1. However, both Dev_1 and Dev_2 have associated this exchange with E DID. If E tries to send M1 with either Dev_1 or Dev_2 DIDs, then the attack fails. Since, as the counter party retrieves the corresponding DID document, the public key extracted will not be a match for the private key used in encryption.

3.6 Summary

In this chapter, the proposed decentralized Identity and Access Management framework has been introduced. The different layers representing the proposal were listed, and further discussion followed to describe the identity relationship model and the details of the identity management layer, including the flows of actions as executed in the main smart contract and the details of the proposed W3C DID IOT method. The authentication process has been discussed later and explained the usage of the decentralized identity to establish a process of mutual authentication in p2p fashion between IoT devices. The next chapter presents the usage of Blockchain tokenization to establish decentralized authorization and accounting processes that form the last two layers in the proposed framework.

Chapter 4

Authorization and Accounting

The identity and authentication layers of a decentralized IoT device management framework were discussed in the previous chapter. This chapter extends the framework presentation by showing how smart contracts are used to implement tokens of both fungible and non-fungible operations to realize a decentralized authorization and accounting mechanisms. The proposed system is described in highlighting core functionalities in the designed smart contracts and the sequence of authorization processes. Performance is demonstrated based on a testbed made of off-the-shelf IoT devices.

4.1 Centralized vs. Decentralized Access Control

The authorization and access control are made of components that focus on how to authorize access, and that is performed by querying, making a decision, and enforcing authorization decisions against defined access policies. Those functionalities are described in AAA frameworks by the terminology of:

- Policy Administration Point (PAP) as the unit responsible for policies creation and modification,
- Policy Enforcement Point (PEP) unit that triggers entitlement policy decisions,
- Policy Decision Point (PDP) providing the actual entitlement decisions on behalf of the PEP, and
- Policy Information/Retrieval Point (PIP/PRP) that supplies the PDP with the information required for a decision.

Figure 4.1 depicts the flow of actions among those different points [67]. Administration creates policies and stores it in the policies repositories. PEP is considered the main interface where all requests are received, and then it consults with the PDP on whether access is granted or rejected. PDP, through the PRP, requests retrieval of the access policy document, and for metadata, it contacts the PIP for information retrieval.

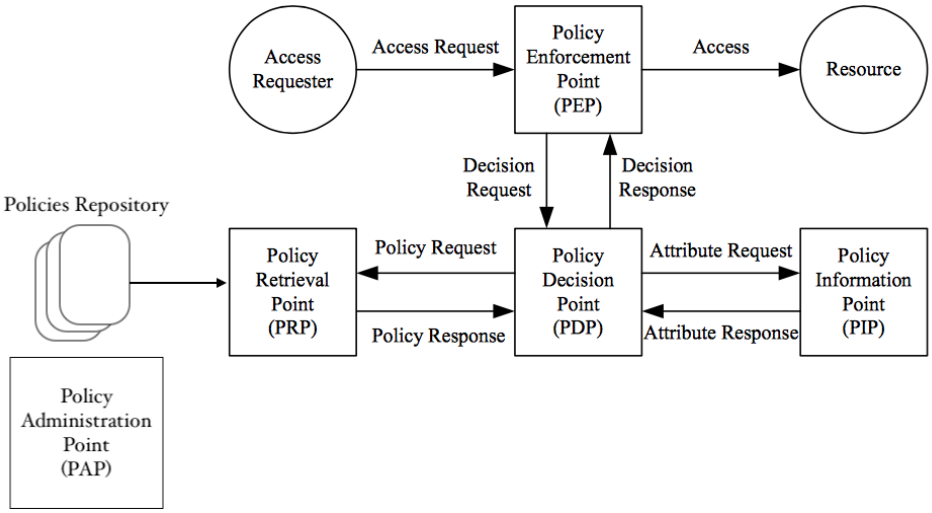

Figure 4.1: The Different Components of Access Control Policy and Flow of Actions.

The difference between a centralized and a decentralized AC model depends on the placement of those functionalities and which entity owns and controls that policy control point. In a decentralized approach, as proposed in this work, the procedure extends beyond a distributed approach, which distributes the policy points among different actors in the network yet under global supervision by the centralized authority.

A decentralized approach places, most importantly, the policy definition and decision making under the control of the device/owner. In such a case, the envisaged architecture of a decentralized access control model grants the PAP and PDP functions to the object owner as they pertain to the points for policy creation and decision making. The object owner defines the policies and makes them available for retrieval to a PEP, which can be implemented inside the device or pushed to a gateway to support device resource limitations.

4.2 Decentralized Authorization and Access Control

In the suggested framework, the central authorization authority is replaced with a set of smart contracts deployed on Ethereum Blockchain, and the authorization and accounting operations are handled according to the logic engraved in the smart contracts' code. The decentralized IoT authorization approach enables devices to grant access to each other in a peer-to-peer fashion that is defined based on the use of the CapBAC model through Blockchain tokenization, based on the author's work on defining decentralized AAA IoT architecture[87].

Devices are required to deploy smart contracts that provide the access control function to permit decentralized service authorization. In traditional networks, access control is performed using either role-based access control RBAC or attribute-based access control

ABAC, and each serves certain applications and offer certain advantages. As it was described in Chapter 2, the CapBAC provides significant advantages over other models, which aligns with the requirement of implementing a decentralized authorization mechanism for IoT.

The suggested decentralized framework relies on several smart contracts for access control that each device deploys on the Blockchain. Each access control smart contract provides a number of functions to enable defining the access control rules for resources and services provided by each IoT device. The architecture also includes a registrar contract that is used to list the different access control contracts information with metadata utilized for resources and service discovery.

4.2.1 A Tokenized CapBAC Model

The main factors in defining a CapBAC model is that a capability token once issued to a subject; the subject should not be able to tamper with it. Also, the token should have a specified data structure that includes at least major fields such as the capability identifier, the delegation rights, the access rights, the issuer, and subject identifiers and timing-related fields. In a CapBAC model, primary functions are also defined, which includes token creation, token revocation, and delegation function. Additionally, to accommodate the constrained environment of IoT networks, the representation has to be light and compact.

The CapBAC model is implemented using the principle of NFT tokens by extending the ERC721 smart contract standard to provide functionalities of CapBAC tokens generation with features of delegation, revocation, and contextual information. The access control smart contract is implemented on a device basis, where for any device, certain access control roles are defined for the resources/services available and issue the required amount

of CapBAC tokens. As a result, one access control smart contract exists for each device with a supply of access tokens as per the rules defined for each resource. In the proposed architecture, to enable dynamic management, the main template for the access control smart contract includes a set of management functions (ABIs):

- issueTokens(): is used to generate new tokens by the device owner when a new resource is available, or a new policy is required.
- revokeTokens(): enables access tokens deletion as an existing policy becomes obsolete or the resource is not available.
- obtainToken(): provides the interface for any requesting device to be granted access by obtaining the CapBAC token issued by the device owner.
- destroyContract(): it provides a tool to disable the access control smart contract on the Blockchain and disables generating any new tokens.

Upon a successful device-to-device mutual authentication, a subject device issues an access request for an advertised resource to the access control smart contract. That is handled by having the subject device issuing a call to *obtainToken* ABI. The function upon success issues a new NFT access token and transfer the token ownership to the subject device. Upon obtaining the token, the device then can access the resource according to the access control rule defined within the token. The PEP functionality, deployed either within the device or a gateway, can query the access control smart contract upon receiving access requests. The smart contract, in this case, is providing the PAP functionality as set by the rules defined in each specific token. In case of failure, the requesting device negotiates with the offering device to amend the authorization rules and generate a token with the desired

access rights. The object device through the management ABI *issueTokens* generates new tokens or deletes obsolete ones and to ensure only the authorized device/owner can access the management ABIs, each of those functions are restricted in access based on a set of modifiers.

4.3 Accounting Layer

The main application of Blockchain is cryptocurrency, and Ethereum is one of the platforms that offers the capability to create a cryptocurrency for entities to use as a transactional mean. The Ethereum ERC20 token standard is a feasible tool to introduce a means of transaction between IoT devices once authorization is granted. The logging and immutability features of Blockchain provide a tool to enable the accounting functionality in a decentralized AAA framework. Thus, this framework introduces a special token that is utilized by the IoT devices to charge for resource usage. The concept of a token has been used for many decentralized applications, including IoT, and a similar approach is followed here with a focus on AAA functionality to support the accounting capability.

The token contract is based on the ERC20 token standard with an addition of a top-up function that will allow a device balance to be topped up to be able to perform transactions with other devices. The device owner usually calls this function through an external interface. The ERC20 contract provides an interface for the ERC721 to charge subjects for granting access tokens based on the defined cost. The accounting model used in this framework is a pre-paid model, where based on the defined set of access rights requested for the access token (either per resource or time) the object device upon access token generation will set the price.

4.4 Smart Contracts Design

In designing of authorization smart contracts implementation, it is required to address the IoT context and list the main requirements to fulfill a decentralized AAA functionality, and those can be described as:

- Full control of devices and owners on policy rules creation/revocation.
- The need to offload AAA management functions out of the constrained IoT devices with the least minimal external authority control.
- The capability to revoke any access rights in a dynamic arrangement and to easily delegate rights with full auditing on resource access.
- The capability to support usage of time-based access tokens that can support spontaneous as well as long-lived connections.

The implementation is based on the DID Documents and two smart contracts handling the mechanisms of discovery, access control and accounting. The advertising and discovery of deployed access control contracts are accomplished through the *service* field in the DID Document, which lists the access control smart contracts associated with a device.

A registrar smart contract is used to enable devices and owners to publish their newly deployed access control smart contracts. The registrar contract represents a repository for the information of available resources and to obtain information on access control smart contract addresses. The registrar contract is used as a replacement of the DID Documents to enable a decentralized authorization layer even in the case of the absence of DID Documents.

The second set is the core of the decentralized access control, where it includes the authorization logic. Specifically, the access control smart contract consists of the PAP, PRP, and PIP functionalities. In contrast, the PEP functionality implementation is left as an application that can reside either in the IoT device or on a gateway that controls few devices. The PEP application has interfaces with the access control smart contract as per the standardized interfaces, including PEP-PDP and PEP-PIP. The third set is the ERC20 token smart contract that provides the mechanism of accounting, which includes the functions to transfer a defined amount of the billing token in return for obtaining an access control token.

4.4.1 Registrar Smart Contract

The primary function of this part is to allow subjects to discover available resources, specifically the access control smart contract and the information of available resources. It is designed with the assumption that an operating entity handles the smart contract management; in terms of adding a registrant or authorizing features. The registrar contract contains a data structure that holds records of registrants and the information about available resources, policies, and addresses of access control contracts deployed. The data structure includes three main identifiers: the registrant is identified by the global identifier based on the Blockchain identity, the unique policy identifier to identify policies, and the access control smart contract by the Blockchain address.

4.4.2 Access Control Smart Contract

This smart contract is designed as a set of templates that a device based on the scenario can select to deploy a specific template to serve a particular access control scenario. The access

control smart contract utilizes the principle of NFT tokens, where capability-based access tokens are defined as an NFT token that is unequivocal and manageable. The skeleton of the access control smart contracts inherits the functions defined by the ERC721 standard including tokenGeneration, tokenRevocation, tokenOwnership and tokensBalance.

CapBAC NFT Access Token

The access control smart contract utilizes the NFT token concept to represent the CapBAC access token. The token structure, as in Figure 4.2, is defined in JSON format to provide a compact representation and allow easy integration with the underlying communication protocol. A number of fields described hereafter defines the structure of the CapBAC token:

- Token ID: this is a unique identifier of the token, which is composed of hashing three other identifiers, subject ID, object ID, and resource URI.
- Token Name: this field is a descriptive field to be human-readable.
- Subject ID: this ID is represented by either the device or user ID, and it represents the token holder.
- Object ID: this is the ID of the object to be accessed.
- Owner ID: this field manages token ownership and also used in delegation confirmation. Different values of Owner ID and Subject ID show that token access rights have been delegated.
- Resource URI: a string representing the resource to be accessed as per the communication protocol.

- Issue Time: a timestamp for when a token is issued to the subject.
- Expiry Time: a timestamp for when a token and access right granted expires.
- ValidFrom: a timestamp indicating when a token is valid. This field should always be greater than Issue Time.
- ValidUntil: a timestamp indicating until when this token is valid. This field should be less than Expiry Time. Both fields are used by the token holder to delegate access rights for a limited time.
- Value: It represents the amount defined by the resource owner to offer the defined access rights.
- Control Flags: A number of flags used to handle token management and represent validation on whether a token can be delegated, transferred, or if it has been revoked.
- Access Rights: a field describing the actions according to the CoAP protocol GET, FETCH, PUT, UPDATE, PATCH, and DELETE.
- Constraints: this field is made of zero or more entries to define granular access rights, and it currently supports merging contextual information either by ANDing or ORing. The entries should have those subfields:

  Type: the contextual condition type such as location, battery level.

  Condition: control condition on access right (e.g. $>, <, =$)

  Value: to define the desired value for condition verification.
- Token Digest: a hashed value of the token JSON document that is used to protect against usage of forged tokens.

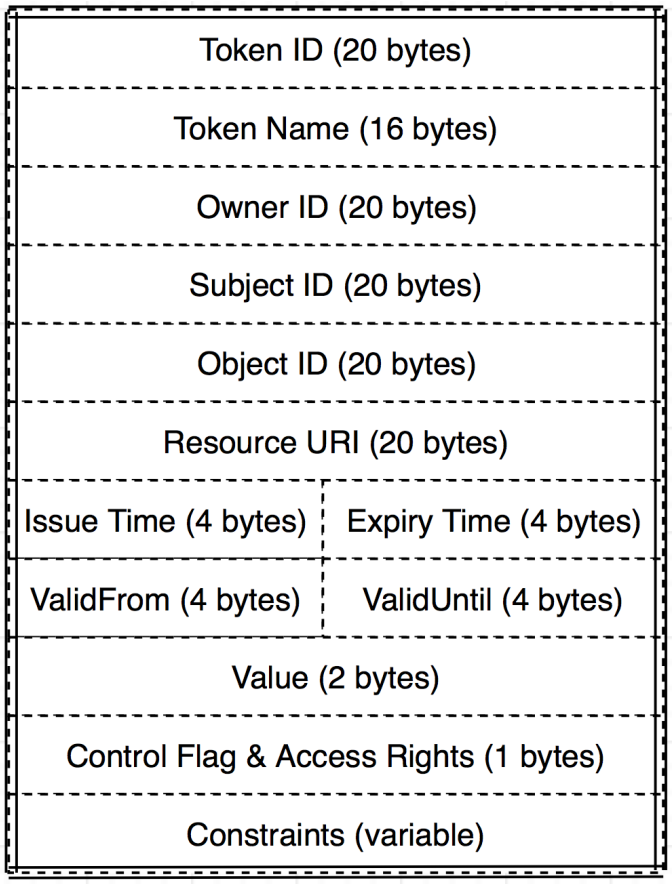

Figure 4.2: Non-Fungible Capability-based Token Structure.

4.4.3 Accounting Smart Contract

The functionality of accounting is performed using a simplified pre-paid usage model implemented using an ERC20 token. The token is used to represent the value of transactions involving obtaining a CapBAC token or to charge for a delegation right. Such an approach provides the entities participating in the authorization step the autonomy needed to price the services offered and flexibility in providing services to the different actors interested.

The implemented accounting tokens is a straightforward translation of the ERC20 standard with uncapped total supply and one additional function, which is the top-up function to allow any device/owner to top-up its account balance that is tied to the total supply. The accounting smart contract design defines an assumption that minting new accounting token is linked to an external crypto or fiat currency. The ERC20 main functions are mapped to specific usages in the accounting mechanism as per the following:

- balanceOf(): The function is called by the access control smart contract when a device requests to obtain a CapBAC Token to check sufficient balance.
- transferFrom(): the main function used whenever a CapBAC token issued to transfer the corresponding value of the token from the requester to the issuer.
- Transfer(): It is also used for crediting against CapBAC tokens issuance as well in funding device accounts by the owner.
- Approve(): offers the device owner the ability to grant a device a certain balance to be used for transacting. Yet, the accounting token is still under the authority of the owner, and this balance can be updated at any time.
- Allownace(): provides the same functionality of balanceOf() but for tokens available and approved by the owner to the device.

The accounting smart contract implements the events to record the transactions involving a token transfer and approve as well as the minting transactions; therefore, all accounting information is registered and accessible for auditing purposes.

4.5 Authorization Process

The authorization sequence diagram, Figure 4.3, shows the steps in a device-to-device authorization process. The requesting device sends an access request to the device where the resource/object exists; with no access token the object device redirects the requester to the address of the deployed access control smart contract. The requester issues a transaction to the smart contract address to obtain a token, which triggers the smart contract to generate a token with the specified arguments, define the value associated, and charge the requester by calling the ERC20 accounting contract.

Upon a successful transaction, the issued token is forwarded to the requesting device. Henceforth, and based on the expiry period of the access token, the requester starts issuing access commands (GET, PUT, ... etc) that pass through the PEP point, which in this diagram is represented by the gateway.

The gateway performs the policy enforcement role, as most decision factors are already embedded in the access token. The only calls need to be made to the access control smart contract is to retrieve the token hash value and check the ownership and delegated authority on the token.

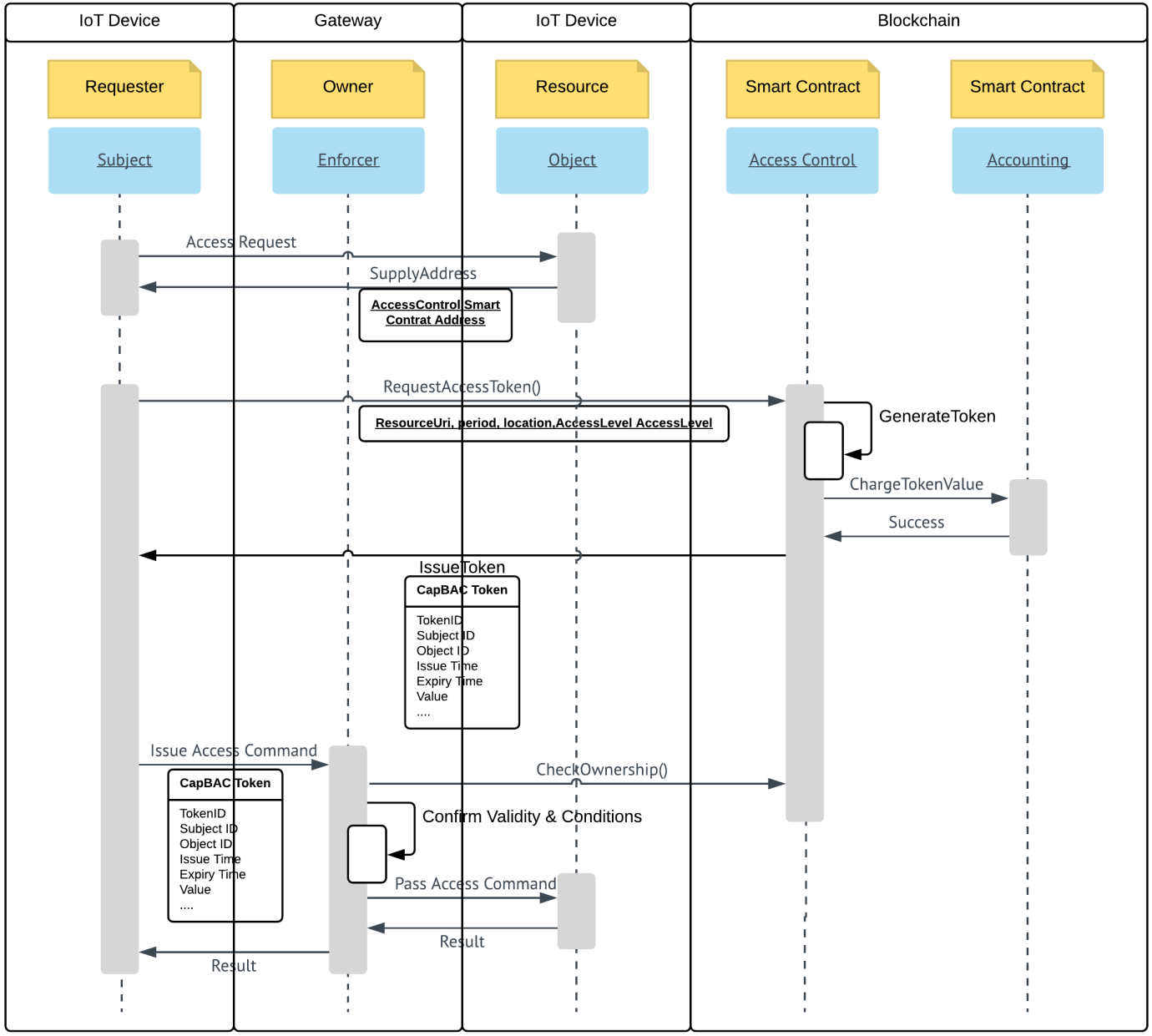

Figure 4.3: Decentralized Authorization Sequence Diagram.

4.6 Delegation Process

The delegation process, as per Figure 4.4, describes how the CapBAC token design provides the mean for a token holder to delegate access rights to other devices. The token field isDelegable defined by the token issuer dictates if the holder can delegate rights or not.

Once a delegate submits a delegation request the token holder (delegator) confirms the delegation capability and then issues a transaction to the access control smart contract. The transaction updates the fields of subject ID to match the delegate ID, define the delegation period by setting the validFrom and ValidUntil fields and then, if required, update the access rights and define other constraints on the delegation. The constraints refinement can only be made to further fine the details of the permitted zone and not to exceeded. For example, if the holder is granted UPDATE right, the delegate cannot obtain PUT right, yet it can still be restricted only to GET operations. The revocation of delegation is not shown in the figure above; however, it is performed by the holder issuing a transaction to the access control smart contract to reset the Subject ID to match the Owner ID and the validity fields to match the issue and expiry time.

4.7 Design Factors for Policy Management

4.7.1 Placement of Policy Management Functions

In the authorization steps, the different policy management functions handle the flow to grant or deny access to a specific resource. The access control smart control hosts the functions of PAP, PIP, and PRP, where it describes the flow of actions and map them to the different components. This approach provides a high level of dynamicity, where CapBAC

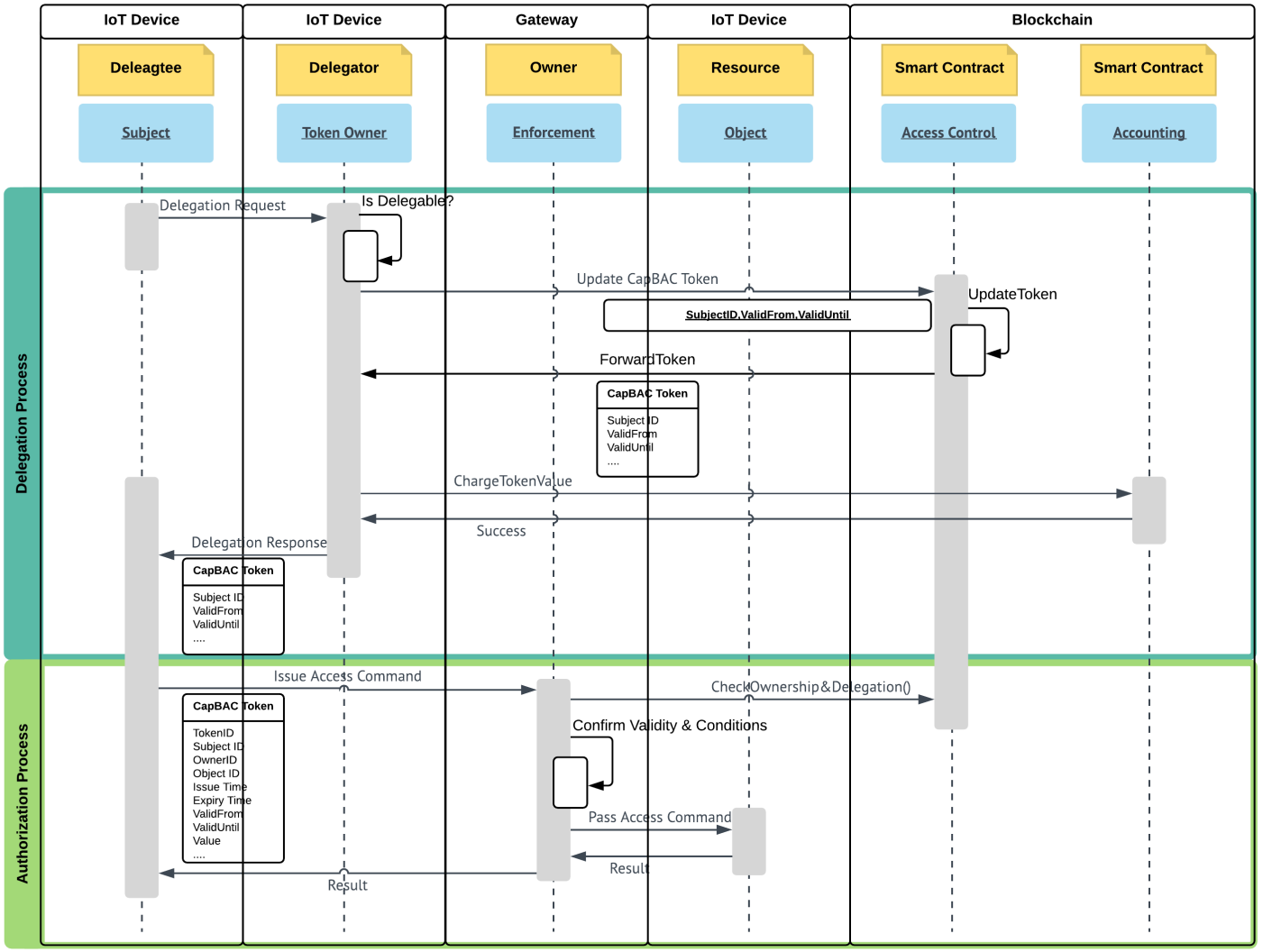

Figure 4.4: Delegation Process Sequence Diagram.

tokens can be generated on-demand according to a negotiated set of attributes. PAP functionality is provided through the tokenIssuance function, which creates the CapBAC token with the set of attributes defined by the issuer. The PRP function is offered by the capability to query the available CapBAC tokens in a smart contract and retrieve all details.

The PDP function is embedded in the CapBAC token itself since all parameters required to make a decision are already compiled, and once a device presents the token and proves the ownership access should be granted. This design allows for resource-constrained devices to handle decentralized authorization with less burden on computing, storage, and battery requirements as the smart contracts are hosted and executed on the Blockchain. The only functionality that is left to be either implemented at the device or a gateway is the PEP functionality, since any request should pass through it before access is granted. The PEP function also has access to the smart contract to perform actions that confirm the validity and ownership of the presented CapBAC token in any access request.

4.7.2 PEP Process Flow

The application residing on the device or the gateway handles the enforcement process by applying a checklist to confirm the token validity and match access rules. These steps are carried out once receiving the CoAP access command with the issued CapBAC token. The process design opted for embedding the CapBAC token, so the PEP function is more self-contained. The PEP process as depicted in the process chart, Figure 4.5, is a series of checklist with a final outcome of the PEP process to be either deny access or grant access response, those checkpoints are:

- Check against Revocation: firstly, the PEP application checks the isRevoked flag to

handle the case of revoked tokens, and if revoked, then it replies with denied access.

- Check for Forgery: the PEP queries the token hash value from the smart contract and compares it with the hash value of the presented token. On an unmatched event, the PEP replies with denied access.

- Confirm Token Ownership: to ensure that the authorized token holder has issued the request, the receiving party check both the subject ID and Owner ID. If both fields are equal, then it is only required to check token ownership by querying the specified interface on the smart contract. If fields have different values that show the case of delegated access, thus again, the receiving party will confirm the ownership and delegation rights through the smart contract.

- Check Token Validity: once more, if the token rights have been delegated, then the gateway will check the validity of token delegation by ensuring that the current timestamp lies between ValidFrom and ValidUntil timestamps. If the original owner still holds the token, then the validity is checked against both Issue and Expiry timestamps.

- Check Access Rights: by comparing the access method used in the CoAP request received and the access right defined in the token metadata against the specified object URI, the PEP application will either deny or grant access to the specified resource.

- Check Constraints: Upon a successful check of access rights, the PEP application validates the condition set in the constraints array. The application pulls the different conditions and values defined and fuses them according to the defined operator (OR/AND).

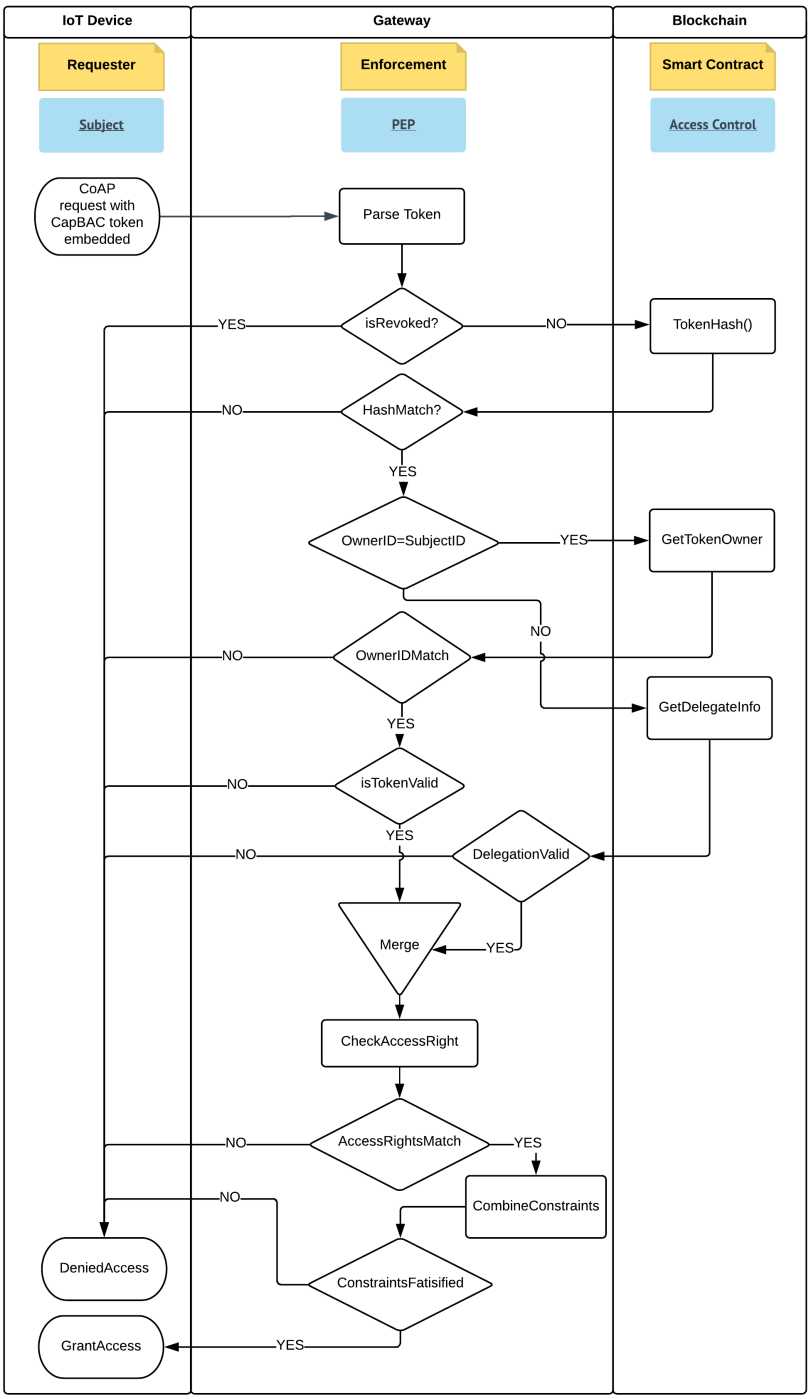

Figure 4.5: Policy Enforcement Process Flow chart.

4.8 Testbed and Performance Evaluation

To evaluate the performance of the suggested approach and highlight main issues a testbed was built hosting an implementation of a use case of few IoT nodes providing sensor data consumed in a p2p approach, Figure 4.6. The testbed architecture was based on 2 IoT nodes hosted on raspberry pi boards and a sense-hat shield. The testbed emulates a group of environmental nodes providing measurements of temperature, humidity, and barometric pressure. Each node work as an end node and as a gateway and hosts a client and server interfaces of the CoAP protocol based on the node-coap library [88], and interfaces to the Blockchain using the Web3.js library [89]. To emulate number of concurrent connections, an application is installed on a laptop that uses the node-coap library to create multiple instances of the client and issue access requests. The smart contracts for identity management, access control, and accounting were deployed on a private Ethereum node, and hereafter the main performance factors considered in the analysis are discussed.

4.8.1 Timeout Ratio

Transactions on Blockchain require confirmation, and in the case of Ethereum public network confirmation time is estimated at 5-20 secs. Thus, it is expected that offloading the AAA functionalities to the smart contract will introduce latency. The CoAP confirmable message exchange specifies default transmission parameters for message timeout and number of retries, which is used in this analysis as per section 4.8 of the CoAP standard [20].

The timeout ratio factor is measured in a scenario of one IoT device serving a resource with several concurrent connections initiated within 30 seconds, equivalent to half the value of the default MAX_TRANSMIT_SPAN value in CoAP [20]. The number of concurrent

connections were increased from 1 to 1000 concurrent connections in steps of 1, 5, 10, 20, 50, 100, 200, 300, 400, 500, 600, 700, 800, 900, 1000.

The results, in Figure 4.7, show that for as low as 200 concurrent connections, no timeouts were recorded. The number of timed-out requests rose slightly above 5% as the total requests sent approaches 600 concurrent connections with a spike appearing at 800 concurrent connections represented by a timeout ratio of 16.25%, and at 1000 connections the timeout ratio is measured at 22.1%. The author associates this sudden increase with the fact that the CapBAC token generation step introduces significant processing delay waiting for block confirmation, and the congestion in block production is the main attributing factor.

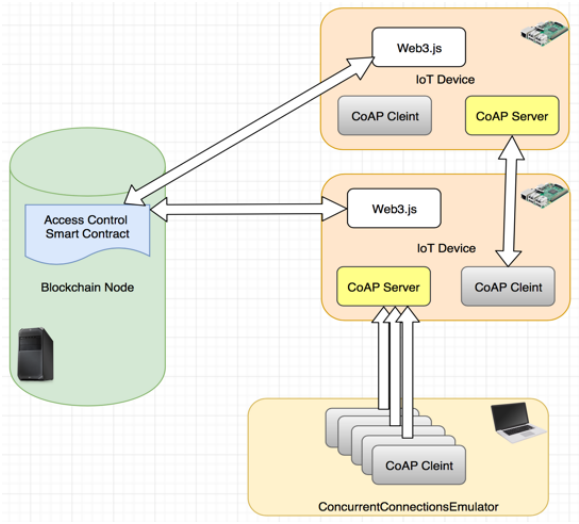

Figure 4.6: Testbed Configuration.

4.8.2 End-to-End Access Time

This parameter is the time measured from the time a device issues an access request for a resource until it receives the response for the resource requested, encompassing all steps in the authorization sequence diagram. Measuring the average time over 1000 runs between two devices in a scenario of newly established connections, which require new CapBAC token generation, shows an average value of 729ms and measuring a maximum value of 1411ms. While in the case of established connections and already issued CapBAC tokens, the average time is at 246ms and a maximum value of 316ms.

The analysis of the authorization sequence show that the bottleneck is present the first time the CapBAC token is generated as it requires changing the state of the smart contract. The subsequent access requests, when the token is still valid, involve only querying the

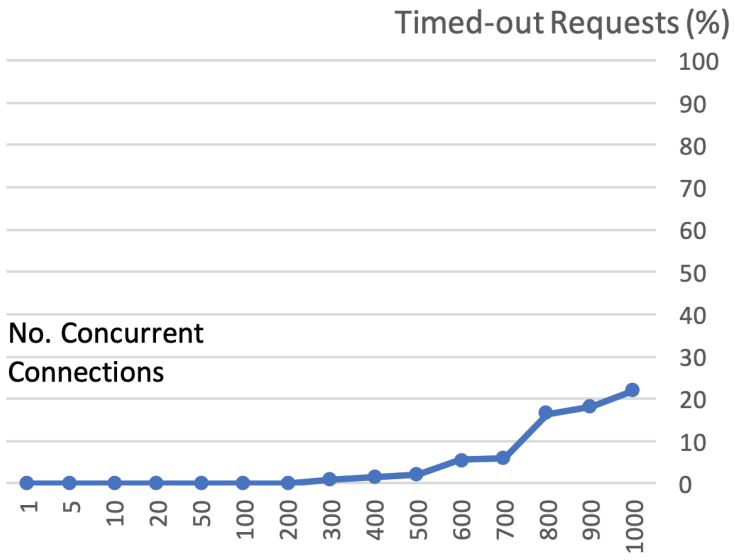

Figure 4.7: Percentage of Timed-out Requests vs. Number of Concurrent Connections.

Table 4.1: Average Gas Cost Per Operation

Operation	**Gas (GWei)**
issueToken	8139
revokeToken	74
updateToken (Delegate)	276
updateToken (Undelegate)	83
transfer	1218
approve	738

smart contract for ownership and delegation information. These results show the benefit of utilizing the NFT capability token approach to issue a self-contained policy decision point.

4.8.3 Overhead

The proposed approach defines a capability token that the requester embeds in any request such that it offloads the policy management functions into the smart contract and make the capability token self-contained for policy decisions. However, that results in an overhead in the data to be sent in each CoAP packet. According to the proposed token data structure, the maximum length is 120 bytes, which in case of having IPv6 as the underlying protocol (1152 packet size), [90], that represents 10.4% overhead in the data link budget.

4.8.4 Gas Cost

While the proposed framework is intended to be deployed in a permissioned Blockchain setup where real gas is not used, Table 4.1 lists the average Gas cost for a number of operations to work as a guideline for the design complexity. The different operations are highlighted with the consumed Gas. Since a query function does not consume Gas, the functions listed include only the functions that changes the chain state.

4.9 Discussion

This section introduces a brief discussion on other factors including generality, scalability, and governance. In terms of scalability, one concern would be in the case of a failure of a single request for token access or delegation and how that affect the rest of interactions. However, with the design of the flows of smart contracts it presents no dependency between different transactions and thus one failed transaction affects only the intended operation.

The proposed token structure contains a metadata that is related to a specific subject and resource and the only variable is the constraints section which varies based on the policy to be imposed. This offers the advantage of having a token of fixed size that does not increase as the number of devices increases.

In order to present the reader with the suggested IoT device hardware requirement, the proposed framework suggested to offload many of the computation intensive operations to the access control smart contract and the self-contained token design, however other operations are still required to be performed by the device or gateway. Based on the IETF RFC 7228 [91] terminologies for constrained-node networks, the proposed framework suggests that a class 2 device capability is required for implementing the different flows described. This device class ensures that it can accommodate for the requirements of network connectivity by offering a memory footprint of 50 KBytes and storage capacity of 250 KBytes.

In the case of framework generality, the framework utilizes standardized identification schemes and standardized tokens specification making it platform agnostic. Furthermore, since the logic is automated through smart contracts, the system can be exported to other platforms running different consensus mechanisms. The main effort required in such case is to rewrite the smart contract in the supported languages by the targeted platform.

Governance is also among the topics addressed when considering deploying a Blockchain network. The thesis work assumes that an enterprise deployment is considered and thus a permissioned setup is needed. The setup requires that Blockchain nodes to be owned and operated by different entities joining the network as a consortium while users and devices can still have access to smart contract through the ABIs. Furthermore, the process of enrolling enterprise entities in the network varies from one case to another. However, in case of an existing managing authority, this entity will assume the role of deploying and owning the registrar smart contract which is the approach proposed in the use cases.

4.10 Summary

Building distributed and decentralized applications on heterogeneous IoT networks require that lower-level layers to support these operations. The IoT device management layer is among the layers responsible for enabling a decentralized service consumption model. The decentralized AAA functionality presented is a working solution through the utilization of the standardized Ethereum ERC20 and ERC721 tokens.

The work defines a framework that has a foundation of decentralized identity management and built on top of it a mutual authentication scheme. Moreover, by mapping the CapBAC access model to the ERC721 standard, the work demonstrates how access tokens are defined as collectibles, which can be extended to an economy of tradable and collectible access tokens.

The testbed results show the feasibility of the approach with further investigation required to compare the performance when the same approach is implemented on top of PoS and DPoS consensus algorithms. The data overhead issue observed in the testbed results also requires an assessment of a trade-off between embedding the token as a whole or only

add the token ID and perform the token retrieval at the PEP side. The latter suggestion is also expected to suffer communication overhead as the token needs to be retrieved on each request.

Chapter 5

DEIR - A Decentralized Smartphone Identity Service

This chapter introduces a use case that implements the concept of decentralized device identity in combating smartphone device counterfeiting and theft. The use case implements a decentralized version of the existing cellular networks equipment registry function, which is part of the 3GPP specification. This function handles the reporting of stolen and lost devices and ensuring that those devices are blacklisted and are not permitted for usage in any other network. The chapter first introduces the problem at hand and describes the current approach. Then, the suggested decentralized approach is described and compared to the existing solution.

5.1 Introduction

The number of smartphone devices shipped in Q3 2019 reached almost 380 million units with the expectation of reaching around 1.4 billion devices by year-end [92]. Most of those devices are used and connected to different mobile networks operating around the globe, and the challenges arising are on how those devices' identities are maintained and verified. The current approach is a centralized database maintaining the phone identifiers and providing a mechanism for operators to host a local database registry that replicates and pushes updates to the central registry (IMEIDB). The IMEIDB is a centralized registry hosted and operated by the Global System for Mobile Communication Association (GSMA), which is acting as the Global Decimal Administrator (GDA) since 2004 [27].

The IMEIDB utilizes the unique IMEI identifier (International Mobile Equipment Identity) that distinguishes each device among the billions of smartphone devices manufactured. IMEI is a 15-digit number where all digits have the range of 0 to 9 coded as a binary coded decimal and used to identify each device on mobile networks [28]. The IMEIDB uses these identifiers to enforce operations related to combating devices' theft and counterfeiting as devices can be labeled as either whitelisted, graylisted, or blacklisted. The access to the GSMA IMEIDB is offered to the 3GPP (3rd Generation Partnership Project) members, including network operators, device manufacturers, and qualified industry parties. The 3GPP TS 22.016 specification defines the IMEI identifier [28], which has the structure depicted in Figure 5.1.

IMEI numbers are composed of the three main elements divided into subsets to identify manufacturers codes and serial numbers:

- Type Allocation Code (TAC): It has a length of 8 digits, and it represents the manufacturer code with the first two digits representing the issuing authority;

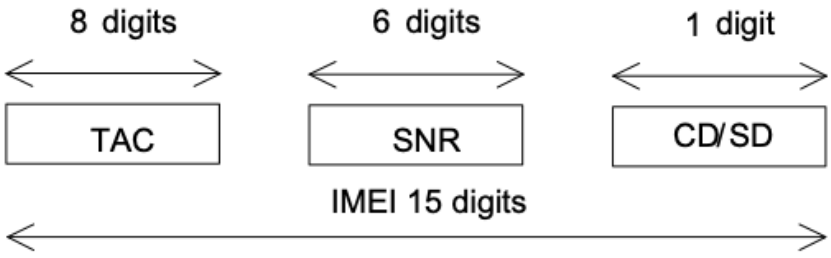

Figure 5.1: IMEI Number Format.

- Serial Number (SNR): This is an individual serial number uniquely identifying each equipment within the TAC with a length of 6 digits;
- Check Digit (CD): The Check Digit is used as a checksum to ensure IMEI correctness, and it is computed in a specified way as defined by the 3GPP TS 22.016 specification [28].

5.2 Publicly Administered Smartphones Database

The smartphone industry is one among many other industries lucrative for counterfeiting and grey markets, due to the high demand for the product and the high margin that can be made from counterfeited products of smartphone brands such as Samsung, Huawei, and Apple. The unique feature of smartphones is that each device has a unique identifier called IMEI. The IMEI is a 15-digit number that is used to identify a device on a mobile network. The mobile industry uses these identifiers to improve operations and deter theft and counterfeiting. The IMEI's 15 digits have the range of 0 to 9 coded as a binary coded decimal, and it is divided into subsets to identify manufacturers' codes and serial numbers [27].

The main IMEI maintained database is the IMEIDB, maintained by the Global System for Mobile Association (GSMA). The IMEIDB is a central database under the full control of GSMA, and it contains basic information on serial numbers, (IMEI) ranges of millions of 3GPP devices (GSM,3G, LTE) e.g., mobile phones, laptop data cards etc. that are in use across all 3GPP networks worldwide. GSMA handles the operation of the database and the allocation of IMEIs to handset manufacturers to ensure that no two devices are made with the same IMEI. The GSMA performs this role by recording a list of all IMEIs that it has allocated in the IMEIDB. Upon assigning IMEIs to a device manufacturer, GSMA stores some basic information associated with the IMEI. The information includes the manufacturer name and the model identifier of the associated device and some of its technical capabilities (e.g., frequency bands, Operating System, WLAN & Bluetooth capability, etc.) [27].

Access to the IMEIDB is provided through GSMA, and it is limited to its members, including network operators and manufacturers. The IMEIDB is utilized to offer blacklists for devices that should be denied service on the network because they have been reported as lost, stolen, faulty or otherwise unqualified for use. Part of the IMEIDB is the Central Equipment Identify Register (CEIR), which provides a tool for network operators to share their individual blacklists. Thus, devices denied service (blacklisted) on one network will not work on other networks even if the SIM card in the device is changed [93].

The main purpose is to allow operators to deploy local Equipment Identity Register (EIR) in their networks and synchronize with the CEIR to offer a global view to all participants of blacklisted devices. According to many reports, including the Federal Communications Commission's Mobile Device Theft Prevention Working Group, IMEIDB has been in use since 1996. Yet, a minimal effect is achieved in preventing counterfeited and stolen devices from reaching consumers [94].

Another type of IMEI management database is local and regional databases introduced by regulators that did not join the GSMA due to many issues related to membership conditions. One of the local IMEI DB is managed by the Malaysian Communications and Multimedia Commission (MCMC) to implement a Public Cellular Blocking Service (PCBS). A system called MCEIR is implemented with a purpose to handle the lost/stolen devices by maintaining a central repository for the reported devices [95].

Similarly, in 2016 both the Omani and Pakistani telecommunication regulatory authorities launched a service that helps consumers to find whether the mobile device a user plans to purchase is authentic or fake within minutes through text messages. A smartphone device must be marked with the authority name stamp, and customers can report about counterfeit or stolen devices to regulatory authority through an email. Consequently, the necessary action is initiated by the authority to update the local IMEI database [95].

Regional attempts to build an IMEIDB also exist. The East African Communications Organization (EACO) is a regional organization formed from national ICT regulators, operators, services providers in certain sub-Saharan countries that built a regional IMEIDB. EACO effort started in 2012 to build a blacklisting IMEI database to handle counterfeited and smuggled smartphones in the region [96].

5.3 Drawbacks of the Centralized IMEIDB

In all mentioned centralized database systems, access to the database is maintained and controlled by an authority that grants access to the information based on a subscription model. The subscription model is made only available to certain parties based on certain qualifications. For example, the global GSMA IMEIDB is accessible mainly to network operators, device manufacturers, and law enforcement agencies.

A major drawback is the limited adoption of such a system due to the fact that access to the information regarding device identity is limited. In the case of local or regional databases, in addition to access restriction, the information gathered is made only available in a limited region, though the effect of counterfeiting and theft is more global, and hence this makes such systems ineffective.

In the case of the GSMA database, few drawbacks hinder the adoption by operators and involved parties. The main drawback is the prohibitive membership fees, which is, according to GSMA starts at an annual fee of $13,000 for entities with revenue between 0-50 million USD and reaches $124,000 for entities with $700 million or more in annual revenue [97]. The high fees prohibit many operators, especially in developing countries, from joining and connecting their EIR with the CEIR. Additionally, the supply chain of smartphones includes many entities, e.g., recycling agents and retailers that currently cannot join the system. The reluctance of many entities to join is evident; the map illustrated in Figure 5.2 only operators in 35 countries worldwide are connected to the IMEIDB [98].

Figure 5.2: Usage of GSMA IMEIDB Worldwide. Source: GSMA

Another drawback is the slow and manual process, where the IMEIDB is accessible only through service providers and law agencies for reporting lost or stolen devices, and the main reason is because of the lack of trust and proving the ownership of the device by the reporter. Thus, reporting a stolen device requires days and weeks to be populated and reflected in the system and then replicated to other operators' EIRs. This manual process forms a gap that allows thieves and fraudsters with the capability of using those supposedly blacklisted devices in different networks and territories. This issue also has been reported by the FCC Technological Advisory Council (TAC) Mobile Device Theft Prevention (MDTP) Working Group as one of the tools used in organized crimes and trafficking [94].

By the end of 2018, among the 1250 mobile operators worldwide, only 119 operators have joined the GSMA IMEIDB [98]. This number represents approximately 21% of the total number of operators worldwide participating in providing information and utilizing the GSMA IMEIDB to combat a problem amounting to losses estimated at 45.3 billion EUR and among the main reasons for lack of adoption is the centralized approach used [99]. The author, therefore, proposes that a decentralized scheme built on trust granted to the end-user and all parties can help in increasing the adoption rate, enhancing the speed

Figure 5.3: Current Effect of the Centralized IMEIDB

of information propagation among all parties, and providing ease of access.

The effects of devices' theft and counterfeiting include financial loss to the manufacturer, deterioration of service level and extends to other security and privacy concerns linked for example to the SIM swapping problem and the possible consequences of thieves to obtain access to users login information in financial services, social platforms, and many others. Fraudulent activities based on counterfeiting and theft rely mostly on the hardship for information dissemination among all parties involved in the supply chain and the lack of visibility on the device identity throughout its life cycle.

In this use case, the framework of decentralized identity and DID identifiers is applied as a solution for smartphone counterfeiting and stolen devices. The use case defines a solution for establishing a unique and global digital identity that can be maintained throughout the smartphone life cycle.

The use case proposes utilizing the IMEI as a unique and global digital identity to support device ownership management and identity update that enable the consumer to have more control over the device and all the attached services. This approach provides a tool for consumers, distributors, network operators, IP protection agencies and device manufacturers to combat the issues of mobile phone theft and counterfeiting and enhance the mechanism for reporting stolen phones and make it more accessible to the end-user.

5.4 3GPP Related Specification

The IMEIDB solution is based on a Central Equipment Identity Register CEIR, and local Equipment Identity Registers EIRs, where the CEIR is hosted and maintained by the GSMA and any operator connected need to maintain a local EIR. The Equipment Identity

Register (EIR) is a cellular network entity that stores lists of the IMEI numbers, with a one-to-one mapping to physical devices. The IMEI is used to identify the actual handset and is not dependent upon the International Mobile Subscriber Identity (IMSI), which is used to uniquely identifies every user of a cellular network and linked to the SIM card [28].

The EIRs function of combating theft and counterfeiting of devices is done by comparing the queried IMEI during handset registration to the cellular network against three lists provided by the network operator:

- Blacklist - Devices of the Blacklist are denied access to the network
- Graylist - Devices on the Graylist are allowed on the network, but may be tracked.
- Whitelist - Devices on the Whitelist are allowed access to the network.

The use of IMEIDB is based on the unique IMEI identifier that is checked against black and white lists. This process is governed through a number of 3GPP related standards that define the components used in the operator network to maintain those lists and provide the required interfaces to interact with other network elements. EIR is connected to other components over the S13 and S13' interfaces. These two interfaces connect the EIR towards both the Mobility Management Entity (MME) and the Serving GPRS Support Node (SGSN), respectively, which handles devices registration and allocation of services, both voice and data [100].

Figure 5.4 below shows the different components in the core network of a mobile network and how the EIR is connected to other components. According to the 3GPP 22.016 specification [28], the EIR performs the procedure as defined by the two commands ME-Identity-Check-Request/Answer (ECR/ECA). The ME Identity check procedure is executed between the MME and the EIR, and between the SSGN and the EIR.

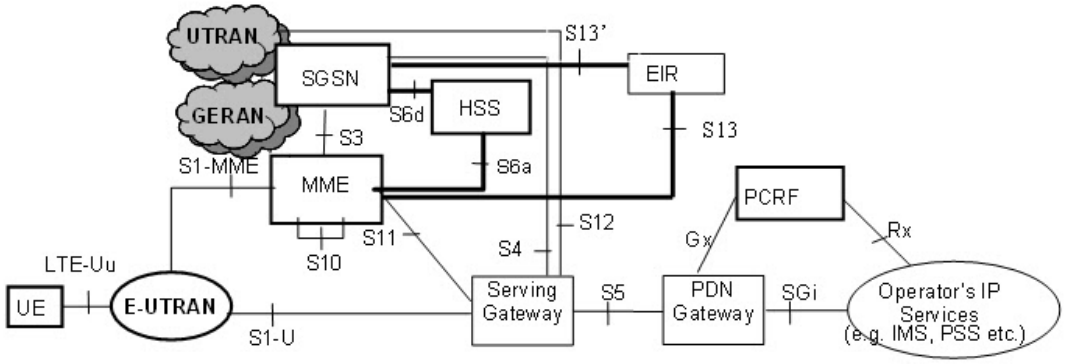

Figure 5.4: EIR in LTE Networks

Once the MME and SSGN are configured to use the EIR, they start performing the registration procedure by sending first the ME-Identity-Check-Request with certain information elements that include IMEI as a mandatory field and IMSI as an optional field. The EIR responds with the ME-Identity-Check-Answer that includes two information elements. The result code indicates if successful or if there is an error. The other information element is the equipment-status, which is only included if the result of the ME Identity Check is flagged as successful. For the EIR to be in synchronization with IMEIDB, the northbound interface for EIR is linked to the CEIR, which is an SFTP-based interface according to the GSMA SG.18 IMEI database file format specification [27].

5.4.1 EIR in 5G

EIR as component continues to exist in 5G networks (known as 5G-EIR) with interfaces to the 5G network components replacing the functionality of both MME and SGSN. As diameter interfaces are replaced by RESTful APIs (Application Programming Interface) in 5G, the S13 and S13' are both defined in 5G using RESTful API endpoints. The interface

to the 5G-EIR has a reference point N17 for interaction between the 5G-EIR and the AMF (Access and Mobility Management Function) that replaces MME function in LTE (Long Term Evolution) networks [101].

The 5G-EIR API endpoint named *N5g-eir_EquipmentIdentityCheck* is consumed by AMF to check the Permanent Equipment Identifier - PEI (a general term for IMEI and other possible identifiers in 5G) and check if the PEI is in the blacklist. The procedure in 5G is based on HTTP RESTful API calls [101].

1. The AMF issues an API GET request to the resource representing the PEI equipment Status. The call parameters include the PEI as a mandatory parameter and the SUPI as optional (SUPI can be the IMSI)

2. If successful, a response code of 200 is received along with the message body containing the equipment status of the PEI.

3. "404 Not Found" error code is received with the "details" attribute set to "ERROR_EQUIPMENT_UNKNOWN".

4. The AMF based on the PEI status will send either a Registration Reject if the 5G-EIR indicates that the PEI is unknown or blacklisted or accept if it is in the whitelist.

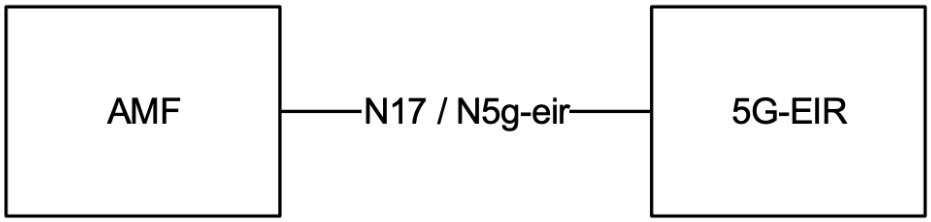

Figure 5.5: EIR Interface in 5G Networks

5.5 Decentralized IMEIDB

The proposed solution is based on developing a blockchain-based Decentralized EIR (DEIR). The DEIR is a 3GPP standard-compliant as it is assumed to be substituting the current EIR implementation and maintaining all standard interfaces to other network elements (e.g., AMF, MME, and SGSN).

The solution, as per the proposed decentralized identity management framework in Chapter 3, utilizes the decentralized identifiers and verifiable credentials specifications to provide the identity layer required to enable trust among the different parties involved [102, 103].

The solution is implemented using Ethereum as the Blockchain technology and the ETHR-DID method specification developed by the uPort team, [57], with extensions developed by the author to accommodate the specific details of the use case.

5.5.1 System Architecture and Operations Flow

The proposed system architecture is composed of a DID Hub for DID Documents storage; the hub provides means for storing generated DID Documents and making it accessible. The DEIR which replaces the EIR functionality and implements the smart contracts for identity management. The other component of the architecture is the IMEI DID method, which is an extension of the ETHR-DID method. As for any DID method, a DID resolver also is an integral part of enabling resolving the DID to the corresponding DID Document.

The system architecture in Figure 5.6 depicts the different components and stakeholders and describes different action flows. For identity creation, marked by hard lines, the flow shows the steps required to create and register a DID and store the corresponding DID

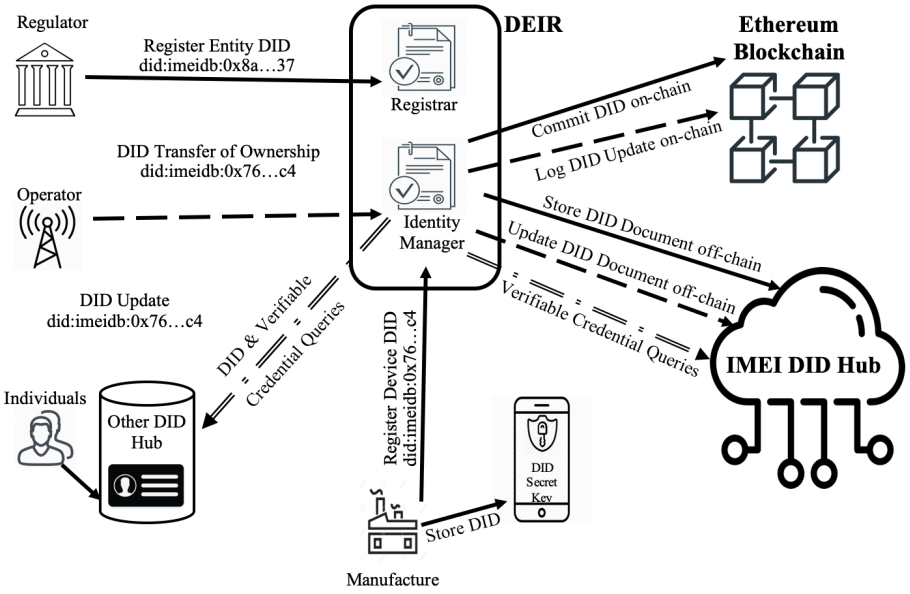

Figure 5.6: System Architecture and Operations Flow

Document using on-chain and off-chain storage capability.

1. The first step creates the DID, which does not require interaction with the Blockchain as the manufacturer creates a pair of Ethereum keys and use the Ethereum address to construct the DID.

2. Upon device assignment through sale, the DID document gets created according to Listing 5.1. The owner DID can be based on other DID Hub for individuals identities such as uPort [57]. The DID Document is submitted to the DID manager.

3. The DID manager performs the necessary verification's through pulling required verifiable credentials. The smart contract registers the DID Document and stores it in the Blockchain.

The flow also highlights the transfer of ownership flow, which is initiated by the owner, verified by the DID manager through querying the DID Hub for the corresponding DID Document and any associated verifiable credential, including current proof of ownership. The process final step is represented by updating the DID document to reflect the transfer of ownership event using the owner field and generate a new proof of ownership for the new owner. The structure of the used verifiable credentials is presented later.

5.5.2 Smart Contract and IMEI DID Methods Details

The implementation details are based on the two smart contracts: the Registrar and the DID Manager. Additionally, the use case defines a specific DID method to support the features required by the IMEIDB, which are described hereafter.

IMEIDB DID Method

In this use case, the main DID methods for CRUD operations (Create, Read, Update, Delete/Revoke) are extended to include other methods that fulfill the need for IMEIDB DID documents verification and ownership management. The four main methods are a straight forward implementation of the DID methods to create a DID and associated document, to retrieve and verify a DID document, to update a DID document attributes, and to revoke a DID. The added methods include change of ownership, verify ownership, report a stolen/lost device, and blacklist/whitelist a device.

Based on the ETHR DID method specification, each DID is based on an Ethereum address as a fully self-managed DID, which provides the feature of better integration and scalability. The proposed format of DID identifier is using the Ethereum address of an entity in addition to the prefix 'imeidb' to construct a DID, e.g., "did:imeidb:0x2dE5...A8Be".

$$imeidb - did = "did : imeidb : "EtherAddress \tag{5.1}$$

5.5.3 IMEIDB Verifiable Credentials

Similar to a DID Document, the verifiable credential (VC) is a JSON-LD Document that contains claims about an identity [61]. In the smartphone identity system, a VC is a proof that an individual has ownership of a device. In ensuring that a VC authentic, a digital signature is employed. The digital signature is constructed for the contained metadata for the claim represented, and this is contained in the field 'proof'. In the case of a proof of ownership claim, it includes the DID of the device, a timestamp of when it was issued (can also include an expiry date for the claim). Most importantly, the claim object includes the DID of the claim's subject represented by the field "credentialSubject".

```
{
    "@context": "https://w3id.org/credentials/v1",
    "id": "did:imeidb:0xd0...2b",
    "type": ["VerifiableCredential", "ProofOwnership"],
    "credentialSubject": {
        "id": "did:ethr:0xeb..21"
    },
    "proof": {
        "type": "Ed25519Signature2018",
        "created": "2019-05-10T17:43:35Z",
        "issuer": "did:ethr:0x25..4f#key-1",
        "domain": "imeidb",
```

```
        "nonce": "1b..h6",
        "signatureValue": "4v3cT...Du+Bg=="
    }
}
```

Other verifiable credential types used include device whitelisting/blacklisting to mitigate for stolen devices incidents. The structure of the claim includes the fields of issued time, incident info, and the proof field; the proof field must include a signature generated by the private key of the owner. Any entity can confirm the legitimacy of the claim by verifying the signature using the owner's public key in addition to pulling the proof of ownership claim and verifying it against the issuer signature. In the last case, the credential subject field is not required since the claim is about the DID of the device, which is represented by the "id" field.

The entire credential object is hashed and signed by the issuer, which constructs a two-part VC: the claim metadata and the proof of authenticity represented by the signature.

```
{
    "@context": "https://w3id.org/credentials/v1",
    "id": "did:imeidb:0xd0...2b",
    "type": ["VerifiableCredential", "ClaimofLostDevice"],
    "incident": {
        "title": "text description",
        "loc": {
            "@type": "GeoCoordinates",
            "latitude": "29.33",
            "longitude": "-41.18"
```

```
        },
        "carrierid": "did:ethr:0xb8...f6"
    }
    "proof": {
        "type": "Ed25519Signature2018",
        "created": "2019-05-10T19:05:17Z",
        "issuer": "did:ethr:0x25..4f#owner",
        "domain": "imeidb",
        "nonce": "2h..d5",
        "signatureValue": "4kgk6...rQMY=="
    }
}
```

5.5.4 DEIR Smart Contract Implementation

On top of the DID layer, the use case implementation includes two smart contracts to handle two streams of operations. The first stream includes operations related to manufacturers' registration, query, and assignment of unique Type Allocation Codes TAC and the second is related to the operations for device identity issuance, transfer of ownership, and verification of claims.

Registrar Smart Contract

The Registrar smart contract is responsible for maintaining the entities registration including functions related to issuing entities DIDs, providing and authorizing access to the

registry for creating and updating devices DIDs, and implementing a mechanism to generate a unique IMEI through establishing a method for assigning vendor code and TACs and ensuring their uniqueness.

Since the DID creation step is based on the Ethereum address, it requires no interaction with the smart contract. However, this step requires first to generate the DID document. The DID Document for any registered entity contains in its minimal format the fields of context, id, publicKey, and authentication.

However, the suggested extra fields are required to fulfill the proposed functionality, which are: *type* - to differentiate between entities and devices DID documents, *vendorCode* - which is assigned by the registrar smart contract. Other optional fields include *created* and *updated* fields that are timestamps for the time of creation and latest update and used for tracing and auditing capability.

```
{
  "@context": "https://w3id.org/did/v1",
  "id": "did:imeidb:0x2dE5...A8Be",
  "publicKey": [{
       id: 'did:imeidb:0x2dE5...A8Be#owner',
       type: 'Secp256k1VerificationKey2018',
       owner: 'did:ethr:0x2dE5...A8Be',
       ethereumAddress: '0x2dE5...A8Be'}],
  "authentication": [{
       type: 'Secp256k1SignatureAuthentication2018',
       publicKey: 'did:imeidb:0x2dE5...A8Be#owner'}],
  "type": "entity",
  "vendorCode": "23",
```

```
  "created": "2019-05-10T17:22:36Z",
  "updated": "2019-05-10T17:22:36Z"
}
```

In the current governance model, it is assumed that the GSMA is handling the entities registration, the Registrar smart contract include the functions that allow a managing authority to perform adding entities and generate entity DID. Other functionalities, also allow the managing authority to assign a delegate to manage the smart contract functions. These functions are among of the requirements by the GSMA specifications, where regional and other delegated entities exist.

The smart contract includes few modifiers that control access to certain functions, including ensuring that entities' generation is permitted for only the managing authority and delegated entities. Another modifier exists to ensure that only authorized entities are allowed to generate new device DIDs.

```
contract Registrar {
    address main_registrar;

    struct delegate {
        address ea;
        uint8 privileges;
    }

    mapping (address => delegate) delegates;

    struct entity {
        address ea;
```

```
        bytes32 name;
        bytes16 country;
        bytes16 vendorCode;
        uint8 tac;
    }

    mapping (address => entity) entities;

    Constructor {
        main_registrar = msg.sender;
    }

    modifier onlyMainRegisrar(){
        require(msg.sender == main_registrar);
        _;
    }

    modifier onlyDelegates(){
        require(msg.sender == delegates[msg.sender].ea);
        _;
    }

    modifier onlyRegisteredEntities(){
        require(msg.sender == entities[msg.sender].ea);
        _;
    }
    ...
}
```

Listing 5.1: Manufacturer Registrar Smart Contract Modifiers

To eliminate the managing authority, the registrar contract can utilize other DID Hubs to verify the identity of any entity. For example, the VON (Verifiable Organizations Network) DID hub deployed recently by the government of British Columbia can be used to verify claims presented by an identity as either a device manufacturer or an operator [104].

Device Registry Smart Contract

The device registry smart contract provides an open global registry used to store the unique cryptographically generated DID, where each DID is pointing to a DID document generated by the manufacturer for each device using IMEI in the *id* attributes. The transfer of ownership operation of the devices is maintained in the global registry so that an auditing trail exist for all transfer events since device manufacturing date.

In this proposal, it is suggested to use an owner attribute to provide the device owner the capability to perform operations on the device DID document and access to DID methods. The owner attribute is an array of at least two entries, the first entry is the device publicKey and the other entries list the device actors publicKeys, where at least one public representing the device owner need to exist.

Since loss of key pairs can render the DID useless, then a backup key or a delegated account key should exist to ensure that the DID methods are accessible. This is a design decision that can be enforced through the ob-boarding business process by instructing the owner to generate a backup key pair.

The device registry smart contract contains the functions for identity creation, transfer of ownership and also update device status in case of reporting a lost or stolen phone. These functions have access control mechanism based on the different modifiers implemented in both this contract and the Registrar contract. Listings show that the creation function is

limited in access to only authorized identity through the *onlyRegisteredEntities* modifier, while the transfer of ownership and update device status is controlled by the *onlyOwner* modifier.

Algorithm 2: DID Creation

Input: imei, publicKey, [attributes ...]

Result: DID Created

if *onlyRegisteredEntity(senderAddress) & imei exists* **then**

DDO $\Leftarrow$ $generateDIDDocument\,(imei, publicKey, [attributes...])$;

dids[imei] $\Leftarrow$ $senderAddress$;

logEvent(senderAddress, imei, DID, timestamp);

else

Ignore;

end

With such a level of control on identity generation, the root of trust is established, and ultimately the device owner has full control over the device identity. Table 5.1 lists the different identity operations and how that builds a root of trust that offers the end-user authenticated access to the operation of reporting device loss and theft

5.5.5 DEIR Interaction in 3GPP Network

The proposed solution is not considered to be a complete replacement for the IMEIDB but rather to decentralize the EIR functionality while maintaining all standardized 3GPP specification [100].

The DEIR is offering interfaces to interact with the two system components (MME and

SGSN), also providing an API endpoint to interface with external actors (manufacturers, operators, user owners). The message flow described here has three flows, MME-to-DEIR, SGSN-to-DEIR, User-to-DEIR. Since the S13 and S13' interfaces carry messages in Diameter protocol format once issued and anticipated by the MME and SGSN, thus a Diameter to DEIR adapter is used. The adapter encodes and decodes the ECR and ECA message formats and extracts the main parameters (mainly IMEI) to query the ledger state.

- **IMEI Check Request**

  The sequence below, Figure 5.7, describes the message exchange flow as a device UE submits a registration request to the network. Once a registration request is received, and depending on whether this is a voice or a data connection, the MME or the SGSN creates a ME-Identity-Check-Request formatted as an ECR message and submits the request to the DEIR.

  The DEIR smart contract for identity management extracts the IMEI and queries the blacklist state. If a verifiable credential for the specific IMEI is found, then

Table 5.1: Identity Operations Root of Trust Establishment.

Actor	**Role**	**Accessible Operation**	**Trust Source**
Managing Authority (GSMA)	Main Registrar	Generate Delegates and Entities Identities	Default Authority & Contract Creator
Regional Office or Appointed Parties	Delegate	Generate Entities Identities	Main Registrar
Manufacturer & Operators	Entity	Generate Devices Identities	Main Registrar or Delegates
User	Device Owner	Manage Devices Identities	Manufacturers & Operators

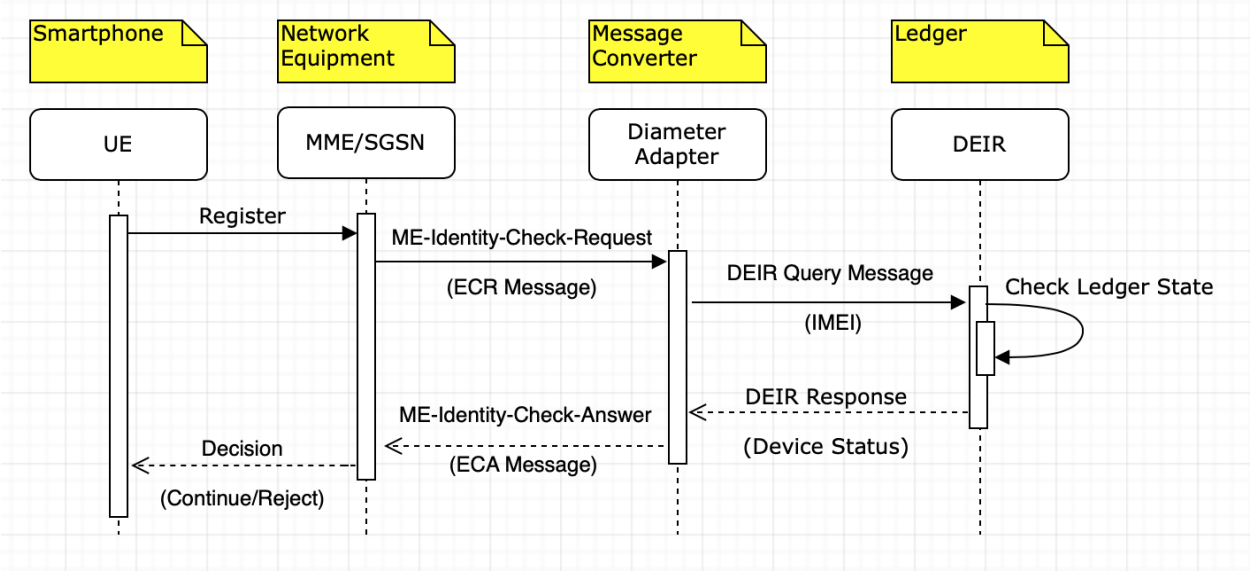

Figure 5.7: IMEI Check Request Process Flow

the DEIR returns the ME-Identity-Check-Answer (ECA message). Consequently, the MME/SGSN, as it receives a reply with confirmation that the IMEI is in the blacklist, it rejects the connection. If the response shows that the IMEI is either whitelisted or graylisted, the registration process continues as in the normal flow.

- **Device Status Update Request**

  The device status update request is issued through a DEIR API endpoint as a device has to be reported as stolen or lost and thus updating the blacklist. The process starts by the user submitting a device status update request (e.g., reporting it as stolen) with an attached VC.

  The API server receives the call, formats a request, and submits it to the Identity Manager smart contract. The smart contract confirms that the issuer is authorized and updates the associated record. If successful, the ledger state is updated, and the

event is logged for auditing purposes. The same flow is used once the owner or an authorized provider wants to revert the blacklisting.

Algorithm 3: Device Status Update

Input: imei, statusUpdate

Result: IMEI Record Updated

if *onlyOwner(senderAddress) & imei exists* **then**

imeiRecord[imei] $\Leftarrow$ *statusUpdate*

logEvent($senderAddress, imei, statusUpdate, timestamp$)

else

Ignore;

end

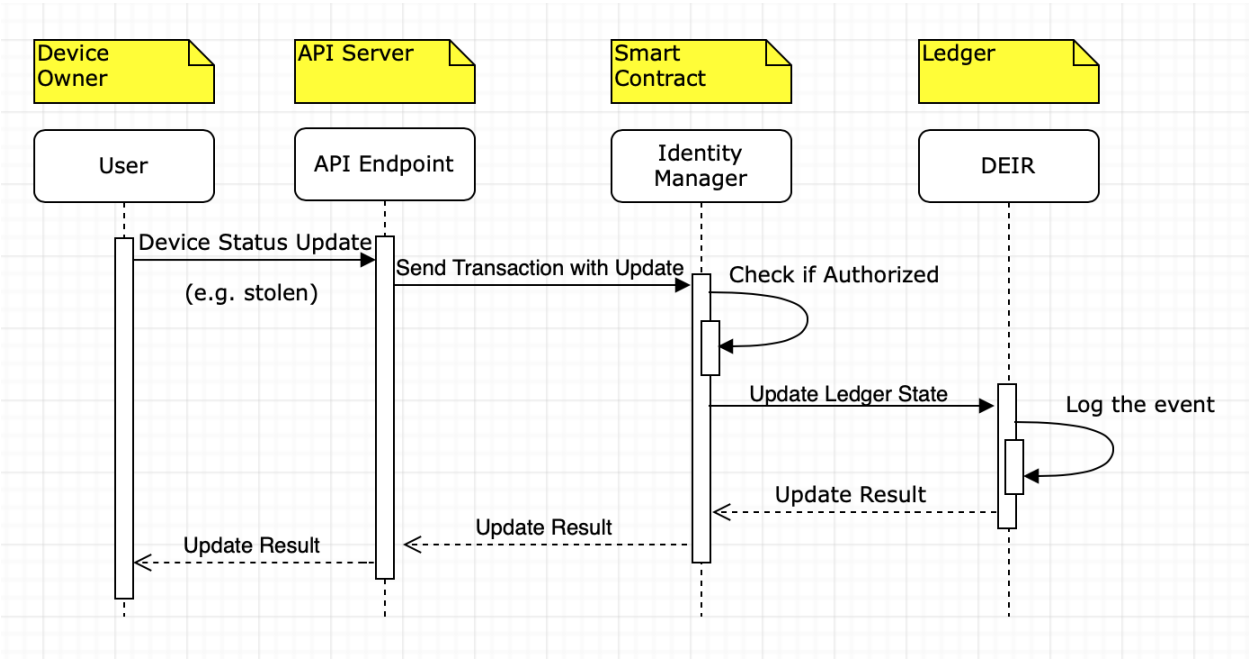

Figure 5.8: IMEI Status Update Request Process Flow

5.6 Summary

The presented use case shows how a decentralized identity management framework can facilitate network operations to solve issues such as counterfeiting and lost and stolen devices. The question addressed has a very high economic, technical, and legal effect and the currently used approach while it has proven to help to combat the problem it still lacks in certain aspects.

The proposed decentralized approach for managing smartphone identities in the IMEIDB provides advantages in terms of cutting costs, minimize the propagation time of the information and ensure easy and global access, Figure 5.9 shows a comparison for these indicators. The use case is a demonstration of the concept of decentralized identity deployed in a private network, and the author's work has been recognized and adopted by

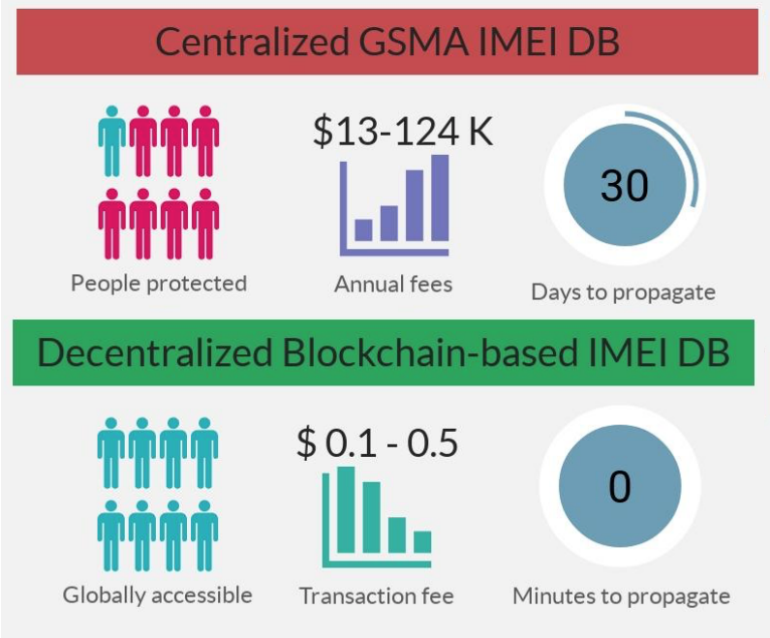

Figure 5.9: Decentralized vs. Centralized IMEIDB

the Linux Foundation and the Hyperledger Project. The use case is a work in progress to be extended by the IoT subgroup in the Linux Foundation Telecom Special Interest Group to be implemented on top of the Hyperledger Fabric and Hyperledger Indy DLTs to test the system in a permissioned DLT setup [105].

Chapter 6

Secure Anti-Counterfeiting Pharmaceuticals Supply Chain System Using Composable Non-Fungible Tokens

This chapter addresses another use case where Blockchain tokenization is applied in one of the IoT applications, which is supply chain management. Management of the supply chain systems is addressed by the use of IoT devices, such as the use of electronic tags to streamline the processes of monitoring, tracking, and authenticating. The use case addresses a specific challenge in the pharmaceutical supply chain system imposed by recently introduced regulatory frameworks related to tracing and tracking. This chapter first introduces the targeted domain, then highlights Blockchain usage, and later presents the proposed solution.

6.1 Introduction

Pharmaceutical companies and other parties involved in the drug supply chain face difficulties meeting the traceability requirements from the regulatory bodies in different jurisdictions and provide the capability to authenticate goods and drugs, which results in failure to combat major issues such as drugs counterfeiting [106].

Moreover, the drug supply chain is highly fragmented, and that prohibits establishing a global view on handling drug packages from manufacturing till reaching the customer with many loopholes that permit the introduction of counterfeit products that patients cannot authenticate. The recently introduced regulations in the US and EU aim at building electronic systems to enable tracing of drugs. Those regulations focus mainly on attaching a 2D barcode for identification. However, applying these regulations and ensuring having a global view and perseverance of records and actions is not yet addressed [107].

The pharmaceutical industry has one of the most complex supply chain systems; it includes different stakeholders such as manufacturers, distributors and pharmacies and with many overlooking bodies. The security of the supply chain in the pharmaceutical industry is considered very important where a breach in the chain can cause failure in delivering the required health service that has a direct impact on patients' health and life.

A prominent issue in the pharmaceutical industry is drug counterfeiting. The impact of counterfeit, substandard, and grey market medicines is estimated to account for $75 to $200 billion yearly, and in particular developing countries, this represents 50% of all drugs sold [108]. In sub-Saharan Africa, the World Health Organization (WHO) estimates that around 116,000 people die yearly due to ineffective malaria medication [109]. Counterfeit and substandard medicines are contributing to hundreds of millions of extra costs in healthcare systems, due to the failure to treat patients right at the first time.

The two main reasons for drug counterfeiting are the high margin that can be gained and also the loose supply chain system in the pharmaceutical industry. The drug bottle or container changes ownership from manufacturer to wholesaler, distributor, and then pharmacist before it reaches the patient.

The issue of tracking and verifying genuine products is even more laborious once it reaches the retailer and pharmacy and enters into what is known as the post supply chain. For example, in post supply chain, manufacturers have limited or zero knowledge about their products, and a drug recall notification usually does not reach the end patient directly.

6.2 Supply Chain in Pharmaceuticals

Supply chain management systems in the pharmaceutical industry are one of the most sophisticated and sensitive verticals. It contains different streams and also many participants. The chain starts with drug manufacturers that can be brand manufacturers or generic manufacturer, where the later focus on manufacturing generic compounds that are used by other manufacturers. In Pharmaceuticals, the distribution of drugs is managed mostly by manufacturers starting from the production site to the drug wholesalers and, in some cases, even to dispensers, especially for hospitals [110].

Another stream in the drug supply chain includes the direct distribution between manufacturers and governments' agencies, especially in vaccines and drugs falling under specific assistance programs. The last segment in the drug supply chain is called the post supply chain, and it is the segment between the retailer (e.g., pharmacy) and the end customer (patient). This segment lacks better tracing and tracking capabilities and suffers the most from the risk of an increase in counterfeit and altered products to reach end customers [110].

6.2.1 Regulatory Environment

The pattern of how technology is used in the drug supply chain has been heavily driven by the tight regulations and policies imposed. The most recent developed regulations in the US and EU are bringing potential implications and introduce the need for smart and intelligent packaging and require an increasing usage of distributed, connected, and trusted technologies to meet the requirements of tracing, visibility and compliance.

Technological efforts focused more on introducing sensing technologies and the use of the RFID and NFC tags to provide the feature of electronically reading the unique identifiers of each package. The tracing and tracking requirements are streamlined using RFID and NFC tags, however cloning of these tags can still take place, and counterfeit and substandard products still reach end customers. However, ensuring one global view and boosting the level of trust among the different stakeholders is still an issue in addition to the lack of interoperability and capability of disseminating events between the adopted systems utilizing different standards.

Many industry bodies identify that healthcare supply chain systems to have five pain points, including:

- **Visibility:** which pertains to the capability of having global view on the products from manufacturing till the end customer.
- **Traceability:** and it relates to the capability to monitor products, their associated events, through the supply chain, and metadata associated with any product
- **Flexibility:** considering the example of a recall this pain point addresses how efficient and adaptable a supply chain system to abrupt events and issues, with the least operational costs.

- **Compliance:** to ensure adherence to the standards and policies imposed by regulatory authorities, specifically in relation to traceability and anti-counterfeiting.
- **Governance:** in the case of healthcare supply chain system with many different stakeholders managing the stakeholder relationship and building the required trust among the involved parties to reduce risks and streamline the process is a significant hurdle.

In combating these issues, the regulatory bodies introduced compliance policies such as the Falsified Medicines Directive (FMD) in the European Union [111] and Drug Supply Chain Security Act (DSCSA) in the US [112]. The aim is to build an interoperable electronic system to define a unique identifier for each sealable drugs lot, package or unit and enable tracing and tracking of prescription drugs as they are distributed in the supply chain in the US and the EU. These regulations are developed to enhance the capability in fighting drug counterfeiting, stolen products, and drugs manufactured in a substandard formula.

6.2.2 DSCSA and FMD Policies

Both DSCSA and FMD mandate that the originators of drugs in the supply chain represented by manufacturers and packagers need to provide a unique verifiable identifier for each package, and those identifiers are printed as 2D barcode labels and attached to each package and lot [111, 112]. Both share many similarities in the type of data needed to build the unique identifier. The DSCSA requires the use of the National Drug Code (NDC) that is interchangeable with GTIN in the EU FMD, and both also require that each label contains the lot number, the expiry date, and a serial number for each package. The data mentioned above is considered as the metadata for every single package.

The FMD regulation takes a step further by introducing the requirement for the dispenser (e.g., a pharmacy) to ensure the verification of the use of each sealable package. The FMD and DSCSA also define the set of the business processes that a product could experience throughout the product life cycle based on the GS1 supply chain rules. That includes all events starting from the time of origin till the drug being dispensed, which requires that 3rd Party Logistics Providers (3PLs) also need to perform verification and decommissioning activities and store that in the system with a high level of trust in performing those operations.

By highlighting those issues and regulatory requirements, the author suggests a Blockchain-based system that offers a viable solution to simplify the logistical process and enable quality control throughout the supply chain. The next section describes the technology and the fit to the drug supply chain regulations.

6.3 Blockchain and Supply Chain

6.3.1 Blockchain Usage in Supply Chain

The usage of Blockchain to enable supply chain applications has received significant attention from many small and big companies to employ Blockchain in enhancing supply chain processes of tracing and tracking and also in combating the distribution of fake and counterfeit products.

A few of the worth mentioning projects include IBM and Walmart's partnership in the Food Trust project, which builds permissioned, immutable, and shared record of food system data among all participants in the food supply [113]. In logistics, IBM and Maersk also partnered to launch a freight network that digitizes and track shipment transactions

worldwide [114]. Other companies are building Blockchain-based supply chain solutions for luxury items, such as the latest announcement by the fashion giant LVMH to establish a provenance platform for its brands such as Luis Vuitton and Christian Dior [115].

Few other startups are also building supply chain platforms for tracing, tracking, and anti-counterfeiting solutions. The list includes Verisart, Provenance, and VeChain [116]. Many of these solutions focus on art and luxury items and limited evidence of the existence of solutions targeting pharmaceuticals. To the author's best knowledge, the only project in the industry based on the use of Blockchain for the drug supply chain is the MediLedger project led by drug giant Pfizer and implemented by a California startup, Chronicled Inc., [55]. The project uses JP Morgan's Ethereum-variant platform (Quorum) to launch a supply chain system for drugs and medicines [117].

6.3.2 NFT Match to Drug Supply Chain Policies

Blockchain's features of immutability, provenance, and tokenization provide a solution to meet the requirements of authenticating drugs' source and secure the supply chain. This use case presents how non-fungible tokens are used to build a drug supply chain tool that enables identifiers creation, authentication of origin, transfer of ownership, and accessibility of drugs information to the different stakeholders, including end customers.

One of the main features of Blockchain is immutability where the committed transactions are irreversible and permanent. For example, a transaction that updates an account balance is not updating directly a balance field. Rather all recent transactions are grouped together and used to form a block that once validated is distributed and become linked to the previous block.

Immutability is an important feature of Blockchain as it ensures that written transac-

tions and data are unchangeable, and this boosts the trust level among network participants even with no prior trusted relationship. Another Blockchain feature is provenance, where an item can be traced to its very first origin. That provides a very powerful tool for auditing and the capability of tracing drugs even to identify the original compounds used in manufacturing.

The author suggests that the features NFTs offer can establish the verifiable immutability and authenticity required in addition to other features such as building unique and related identifiers on both the package and lot level. NFTs also provide simple means for transfer of ownership, revocation, and metadata representation. Those features are well mapped to the previously listed drug supply chain processes.

The proposed work, based on the author's best knowledge, is pioneering the employment

Table 6.1: Blockchain and NFT Match to Drug Supply Chain Policies.

Feature	**DSCSA**	**FMD**	**Blockchain**
Tracking	item & lot-level	item & lot-level	Composable NFTs
Unique Identifier	NDC, lot number, expiry date, serial number	GTIN, lot number, expiry date, serial number	Token Metadata
Events	drug manufacturing, purchase, dispensing	custom	NFT change of ownership and other events records
Data Perseverance	Required	Required	Immutability
Origin Tracing	Required	Required	Provenance
Trust Level	High	High	in-built features of immutability and provenance

of the ERC721 and NFTs for tokenizing drug sealable packages and providing supply chain processes through extending the ERC721 defined interface [118]. The author suggests a composable structure for the NFT tokens that can enable tracing on package level, lot level, and even unit level.

6.4 Composable NFTs for Drugs Supply Chain

In this section, an overview of the proposed solution is presented. Particularly, a description of the proposed composable NFTs is explained. Through the detailed description, snippets and pseudo-codes of the implemented smart contracts are presented with a description of the procedures used among the different stakeholders in the supply chain. Fig. 6.1 describes the overall system components.

6.4.1 Composable NFTs

ERC-721 is established as the de-facto standard for creating non-fungible tokens (NFTs) in Ethereum Blockchain and other Blockchain platforms. ERC-721 standard is used to represent unique, digital assets that range from collectible digital cards to real-estate and art and luxury items. It defines the token-related functions: such as token name or identifier, symbol, the total supply of tokens, the balance of a specific token, query the ownership of a token, approval on taking of ownership, transfer of ownership, and retrieval of token metadata. It also defines major events related to ownership management, which are Transfer and Approval events.

In the process of evaluating the use of ERC-721 to represent sealable packages and identify each package as NFT, a major gap is identified. This gap is related to the fact

that regulatory policies require that tracing and identification should be achieved at both package level and also lot level. However, an ERC-721 token is mapped to a single item with no relationship among different tokens (items). Therefore, The author investigates the use of a composition of tokens under a parent token to represent this relationship.

To address this gap, a promising proposal has been evaluated by the author, which is the Ethereum Improvement Proposal EIP-998 for Composable NFTs [119]. The EIP-998 is an extension to the ERC721 to enable building composable tokens such that it is possible to compose lists or trees of simple NFT tokens that have a joint ownership relation. The entire composition can be transferred with one transaction by changing the root owner of the composite NFT token.

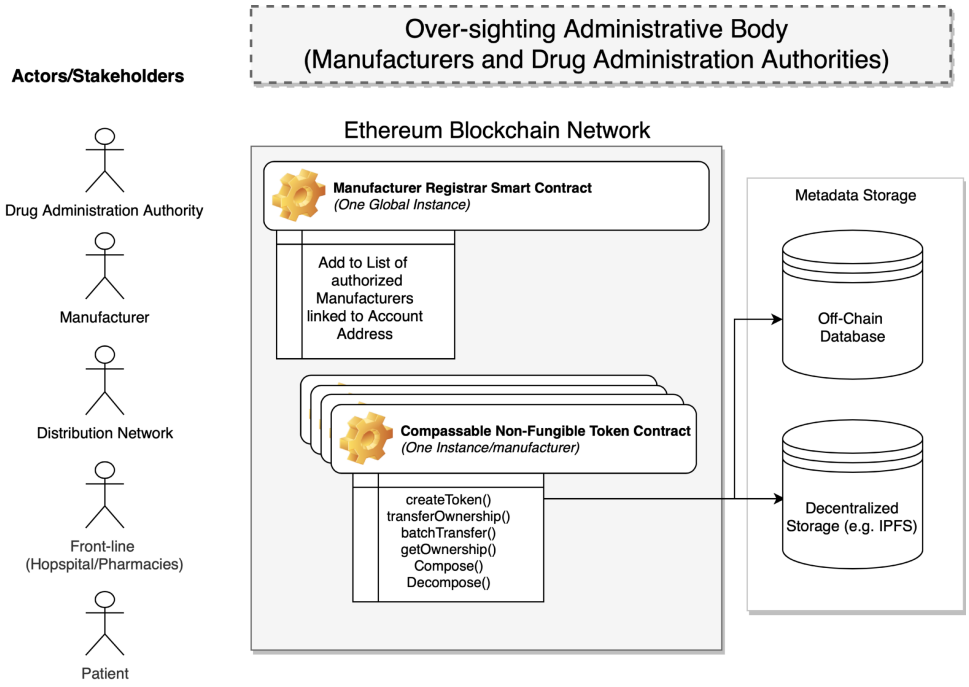

Figure 6.1: Composable NFT Drugs Supply Chain System Components.

NFT Tokens Ownership

ERC-721 ownership is determined by an array of token indexes (TokenIds) that is mapped to the owner Ethereum address. The token ownership can be queried against any tokenId and returns the owner's Ethereum address. Transfer of ownership is performed by changing the owner address value from the old owner to the new owner, given that certain assertions are validated to ensure that an authorized owner performed the transfer of ownership action. The ERC-721 also enables adding delegated actions, such as the owner can approve specific addresses to be able to perform the transfer of ownership on its behalf. This delegation feature is necessary to enable multiple actors to have access to the tokens and perform the supply chain process assigned. Listing 6.1. shows the details of the ownership function and how it is implemented.

```
mapping (uint256 => address) private owners;
mapping (uint256 => bool) private doesTokenExist;
// Query Token Ownership
function tokenOwner(uint256 tokenId) constant returns (address owner) {
    require(doesTokenExist[tokenId]);
    return owners[tokenId];
  }
// Transfer of token ownership
function transfer(address toAddress, uint256 tokenId){
    address tokenCurrentOwner = msg.sender;
    address tokenNewOwner = toAddress;
    // Perform assertion ensure token exists and confirm ownership
    require(doesTokenExist[_tokenId]);
    require(tokenCurrentOwner == tokenOwner(tokenId));
    require(tokenCurrentOwner != tokenNewOwner);
```

```
    require(tokenNewOwner != address(0));
    // Perform transfer actions
    removeTokenFromOwnerList(tokenId);
    tokenBalance[tokenCurrentOwner] -= 1;
    owners[tokenId] = tokenNewOwner;
    tokenBalance[tokenNewOwner] += 1;
    // Record transfer event in Blockchain for auditing
    Transfer(tokenCurrentOwner, tokenNewOwner, tokenId);
  }
// Delegate authority of invoking transfer actions
function approve(address toAddress, uint256 tokenId){
    // Perfrom assertion before delegating
    require(msg.sender == tokenOwner(tokenId));
    require(msg.sender != toAddress);
    // Add delegation
    allowed[msg.sender][toAddress] = tokenId;
    // Record approave event for auditing and tracing
    Approval(msg.sender, toAddress, tokenId);
  }
```

Listing 6.1: NFT Token Ownership Functions

Operations of Composable NFTs

Composable NFTs provide a simple way to form composite assets on the Ethereum blockchain, those assets are still utilizing the same concept of ERC-721 but with the capability to link between parent tokens and child tokens. The targeted use case in this work utilizes composable NFTs in tracing and tracking drugs in the supply chain that are either composed or decomposed along the supply chain, e.g., a sealable package in a lot or a generic compound

origin used in a specific drug.

Composable NFTs, according to the EIP-998, can be formed in a top-down or bottom-up approach. For this scenario, the author selects a top-down approach where the parent token is the lot, and the child token is the package, with a traceability feature from top-down and bottom-up directions. The need for implementing tracing in both directions is introduced to satisfy different supply chain processes.

The need for tracing in both directions is highlighted in two examples. A recall event usually starts by the manufacturer and with the main parameter being the lot number. In this case, the lot number is an identifier for the parent token; thus, the recall action should be able to result in notifications to owners (distributors/retailers/consumers) of affected packages belonging to that lot. On the other side of the chain, a customer querying about a specific package issues the query based on the child token identifier. For transparency, the customer should be able to see the transfer of ownership happening on the lot level to confirm adherence to certain policies.

Using composable NFTs to perform drug supply chain procedures according to the FMD and CSDSA regulations is simplified in a series of actions that are described and defined hereafter as well as the key requirements for the proper operation of the proposed approach.

1. Access to token minting and creation is granted only to the legitimate manufacturers that are assumed to be the original owner of a drug.

2. The manufacturer can create tokens in batches.

3. The capability of storing metadata in a readable and accessible format.

4. The smart contracts can generate the required randomness for the identifiers.

5. The capability to perform operations of composition and decomposition on tokens.

6. The required events by both regulations to enable tracing, tracking and query of change of ownership and origin shall be recorded on-chain.

7. Visibility on information to all stakeholders and the capability of invoking queries.

For the first item, bootstrapping the network is assumed to be performed to ensure only legitimate manufacturers can create tokens and thus ensure the provenance feature and capability to trace drug item change of ownership since the origin. This requirement is necessary to eliminate non-authorized access and to combat counterfeiting. The author assumes the role of a system administrator to handle manufacturers' registration. The possibility of decentralized verification of the legitimacy of manufacturers is not within the scope of this work. However, the author envisions the use of a Blockchain solution based on the Decentralized Identifiers (DIDs), [56], in a similar approach to the Verifiable Organization Network (VON) deployed by the Province of British Columbia in Canada to register business entities [104].

The second requirement is to ensure effectiveness in operation and thus lowering the cost of creating tokens. For ensuring portability, each token has metadata that and can be retrieved using the tokenId. The token metadata representing package or lot attributes are stored off-chain using off-chain database with the possibility to choose decentralized storage such as IPFS. The FMD regulation mandates the fourth requirement, and it is ensured through the use of a hashing function that digests the metadata provided in addition to a timestamp to generate a random serial number for each token.

To ensure the capability to provide tracing and tracking on package and lot level, the proposed composable NFTs require that the smart contract should provide two functions

that perform the function of compositions and decomposition. The first function constructs a composite asset from a number of child tokens or amends child tokens to an existing composable NFT. That also includes the step of updating the parent field of the child tokens with the tokenId of main composable NFT. The decomposition function dismantles a composable NFT and voids the parent token relationship to the child tokens.

Preserving the events initiated by the different stakeholders in the supply chain and ensure its visibility among them is achieved by utilizing the Events feature in Ethereum smart contracts. A specific event in the smart contract is used for each operation to register the actions performed by all actors in the supply chain. All these events are stored on-chain for visibility and indexed using tokenId, manufacturer, owner, and with a timestamp to allow easy retrieval and querying capability. The overall process of establishing and building the composable NFTs representing the drug packages is as follows.

- An authority is assigned the role of an administrator to approve all registration requests by manufacturers and thus boost the trust level of the network.
- Manufacturers utilize a token generation smart contract for batches of drugs with main parameters required, such as lot number, CDN/GTIN, expiry date, manufacturer details (name, country of origin, manufacturing facility .. etc). The token generation smart contract generates the serial number required by the regulation, which is used as the NFT tokenId.
- The tokenId is generated using a Keccak hashing function that consumes as input all the supplied in addition to the current timestamp to ensure uniqueness and randomness.
- A composition function is called to form a composable token representing a lot.

- Upon an event of selling, or in general, transfer of ownership of drugs from one entity to another, a batch transfer function is called.
- In the event of dispensing, a transfer of ownership function is called to change the owner parameter of the tokens to the end-user and locking the transfer feature.

6.4.2 Details of Smart Contract Implementation

This section details the implemented smart contracts that are developed to fulfill the requirements. The implementation requires two smart contracts; the first is the manufacturer registrar (Manufacturer Registrar - MR), and the second is the Composable NFT Manager (CNFT).

Manufacturer Registrar Smart Contract

The MR smart contract implements the functions for managing the information of manufacturers, including the functions of manufacturers' enrollment and manufacturers' verification. The current implementation assumes a permissioned ledger, where the existence of an oversight body to act as an administrator for the MR smart contract is necessary. The drug administration authority is the default actor that can assume the administrator role.

The administrator role requires the validation of the submitted manufacturer information to ensure that the root of the process is kept intact and boosts the trust relationship among all other parties. By ensuring a verifiable and validated identity of the manufacturer, then the origin of all drug tokens creation is ensured to be intact and no counterfeiting. The author does not propose a fully decentralized approach since the nature of the application require authority oversight. However, the author suggests that in case of the existence

of a DID and verifiable credential network similar to the VON network, [104], for manufacturers, then the MR smart contract can leverage that for validating manufacturers' registration requests.

Algorithm 4: Manufacturer Smart Contract

Input: Manufacturer Information and Ethereum Address, Admin EA
Result: Manufacturer Ethereum Address (EA) Added to Authorized List L
procedure MRContractCreation(AdminEA)

$Owner \Leftarrow AdminEA$;

procedure EnrollManufacturer(Manufacturer Info) Manufacturer submit enroll request;
if *information provided complete* **then**
  if *Admin Approval* **then**
    $L \Leftarrow [L, EA]$;
    Event $(EA, Name, Timestamp)$;
  else
    Ignore;
  end
else
  Ignore;
end

Algorithm 4. shows the pseudo-code of the MR smart contract, the smart contract includes the owner variable which is assigned to the administrator, upon invoking the smart contract creation the owner variable is assigned to the administrator Ethereum address. A manufacturer submits an enrollment request, including the parameters of the manufacturer name, country, license number, and Ethereum address. The manufacturer enrollment process requires a validation step by the administrator to approve the request and commit the manufacturer Ethereum address in the list of authorized manufacturers. Listing 6.2. is a snippet of the MR smart contract that shows the main variables and data

structures used.

```
contract MR {
    address admin;
    struct asignee {
        address assignee;
        uint8 privileges;
    }
    struct manufacturer {
        address ea;
        bytes32 name;
        bytes16 country;
        bytes32 license;
        uint8 reputation;
    }
    Constructor {
        admin = msg.sender; }
    ...
}
```

Listing 6.2: Manufacturer Registrar Smart Contract Variables

Composable NFT Smart Contract

The composable tokens CNFT smart contract is created by each manufacturer. The CNFT smart contract is an extended version of the ERC-721 standard with two main functions added that perform the functions of (de)composition of composite assets. The CNFT smart contract also adds other main variables such as rootOwner and childTokens, to allow traversing the ownership of composable tokens in top-down and bottom-up directions.

The CNFT smart contract in addition to the (de)compostion functions four other main functions:

- createToken(): For a new batch of drugs, the manufacturer invokes this function to generate the NFT tokens for each package and call the composition function to generate a composite asset for the lot.
- transferOwnership(): To support tracing and tracking, the CNFT smart contract extends the transfer function in Listing 1. to include the capability to reflect a change of ownership of all child tokens as well as the rootOwner. This function is called for each shipping, selling, or dispensing event in the supply chain management.
- batchTransfer(): This function supports transferring batches of packages that belong to the same lot. The function iterates through the list of tokenIds for all packages need to be transferred and perform the (de)composition steps required to update ownership details.
- getOwnership(): this is a query that will traverse the composable NFT to return the single token owner as well as the root owner token.

Listings 6.3 and 6.4 show the details of the create token function and batch transfer function.

Algorithm 5: NFT Batch Transfer Function

Input: [tokenIds ...], fromAddr, toAddr

Result: Updated list of childTokens in original and destination CNFTs

if $fromAddr == requestSender \, || isApproved(requestSender)$ **then**

  for $i \leftarrow 0$ **to** $tokenId.length \; 1$ **do**

    $tokens[fromAddr] \Leftarrow [tokens[fromAddr], pop(tokenId)];$

    $tokens[toAddr] \Leftarrow [tokens[toAddr], tokenId];$

    $tokens[tokenID][toAddr].owner = fromAddr;$

    $tokens[tokenID][toAddr].rootOwner = fromAddr;$

  end

else

  Ignore;

end

```
function createToken(bytes32 cdn, bytes16 lotNumber, uint32 expiryDate)
    public returns (uint256) {
        require (msg.sender == adminAddress);
        tokenCount++;
        tokenId = keccak256(cdn, lotNumber, expiryDate, block.timestamp);
        tokenIdToTokenOwner[tokenCount] = msg.sender;
        tokenOwnerToTokenCount[msg.sender]++;
        emit (tokenId, cdn, lotNumber, msg.sender);
        return tokenId;
}
```

Listing 6.3: Create Token Function

```
function batchTransfer(address fromAddr, address toAddr, uint256[]
   tokenIds) external {
    // Perform necessary assertions
    require(_to != address(0x0), "ensure that destination address is non-
       zero.");
    require(fromAddr == msg.sender || operatorApproval[fronAddr][msg.
       sender] == true);

    for (uint256 i = 0; i < tokenIds.length; ++i) {
        uint256 id = tokenIds[i];
        tokens[fromAddr] = tokens[id][fromAddr].sub(id);
        tokens[toAddr] = tokens.add(tokensId[id]);

    }

    // MUST emit event to register that on-chain
    emit TransferBatch(msg.sender, fromAddr, toAddrs, tokenIds);

}
```

Listing 6.4: Batch Transfer Function

Proof of Concept Environment

The proposal implements a proof of a concept for a tool to be used in the drug supply chain traceability system that adheres to the DSCSA and FMD requirements. The implementation is performed in a testbed where the smart contracts were deployed on a private Ethereum node with the following specifications: Dell OptiPlex 7010, with Intel Core i5 Quad 3.20 GHz processor, 4 GB main memory and 64-bit Windows 7 Professional OS. The

proposed tool requires performing a thorough evaluation to discuss questions regarding the total system cost of ownership, scalability, and privacy.

The cost of the system is considered to be affected by the Gas paid for performing each transaction. However, that is the case for a public Blockchain. The expected deployment scenario is an enterprise system and a permissioned Blockchain configuration where Gas cost is not relevant.

6.4.3 Summary

The proposed use case addresses a critical requirement in the pharmaceutical supply chain system imposed by the regulatory bodies. The use case utilizes Blockchain tokenization and composable assets in building a tool that enables tracking and tracing of drugs' lots and packages. The use case provides an enhancement in records accessibility and interoperability and in establishing trust.

The use case, however, includes specific limitations. The main issue is system scalability, for example, the capability of handling the creation of batches of new items. Our proposal tested the creation of 100 tokens in a single transaction; however, further testing is required to assess the trade-off between the number of tokens created in a single transaction and the cost associated. The administration of the network and governance is also a factor that needs to be further investigated. That includes the need to devise of a mechanism to boost the root of trust of the network in the initial phase, and also in establishing a reputation model for the different parties involved.

Finally, privacy is a concern, especially when the ownership of the product reaches the end customer. Suggested future work is to look at investigating the use of Zero-Knowledge Proof algorithms, to provide full privacy over the records of ownership [120].

Chapter 7

Data Provenance and Verifiability in IoT Networks

IoT devices generate data that is processed by different machine learning methods, which are controlled by a diversified list of actors. Moreover, open data models are becoming mainstream for distributing IoT datasets, which requires a mechanism that can prove data derivatives authenticity, and ownership. This chapter presents how the proposed decentralized IoT identity management framework can be extended to support application layer processes that are decentralized and distributed. In this chapter, the author shows how the proposed framework is exploited in building the concept of provenant and verifiable data paths. The approach explains how to build a trusted data chain where datasets generated and processed are stamped using the actor DID signature and presented as verifiable credentials traced to its origin. An example is presented to demonstrate how this concept can be utilized in different IoT verticals.

7.1 Data Provenance in IoT

IoT networks bring value by integrating data from different sources, which can be problematic due to network heterogeneity and dynamic nature. Questioning how to authenticate the sources of the data raises the concern of IoT network vulnerability and the possibility of manipulating and inserting inauthentic records along the data path. This problem is addressed using the techniques of data provenance and verifiable claims.

Data provenance allows a verifying party to check datasets' authenticity to the origin. That can enable tracking the dataset throughout the data path from the source where it is gathered through all the processing pipeline and thus offer the capability to identify any inauthentic data handling. Data provenance has been addressed by both centralized and decentralized approaches, including:

- Centralized Logging: Using centralized and standalone systems that keep records of data generated and log it with necessary metadata, e.g., timestamp, location, and others. One example is the S2Logger system used to track the transmission of data among hosts in the cloud. The main issue with this mechanism is that it is isolated, either in local hosts or within separated networks [121].

- Cryptography Functions: Hashing functions and digital signatures present a method to confirm data integrity and ownership. However, this approach has the drawback of the difficulty of keys management in heterogeneous networks and also the lack of capability to track data changes along the processing pipeline [122].

- On-Chain Recording: The use of smart contracts and Blockchain to track data provenance is achieved by recording data generation and processing events on-chain and

thus maintain a trail of data origin and changes through the record lifetime. The primary issue for this approach is the interoperability across different chains [123].

This chapter introduces how to utilize the concept of DIDs and VCs along with smart contracts and DLT to build verifiable data chain for IoT networks and addresses this through IoT examples with an initial focus on the example of collecting IoT data in a mobility network. The mobility network example is used to describe the overall concept which is described in the next sections how it can be applied in other verticals

7.2 System Overview

In describing the proposed concept, a smart vehicular network is used. It is assumed that the vehicles are equipped with a set of sensors that collect information for use by internal subsystems or external entities in vehicle-to-vehicle (V2V) or vehicle-to-infrastructure (V2X) data exchanges.

7.2.1 Network Components

Building a verifiable vehicular network data path combines a set of components that include first the vehicular network elements and the components responsible for maintaining data provenance. Figure 7.1 depicts the different network components that consist of:

- Vehicles: Vehicles and sensors are the main components producing and originating mobility data. Each vehicle owns a DID based on the vehicle identification number (VIN), while sensors might or might not own a DID depending on their capability.

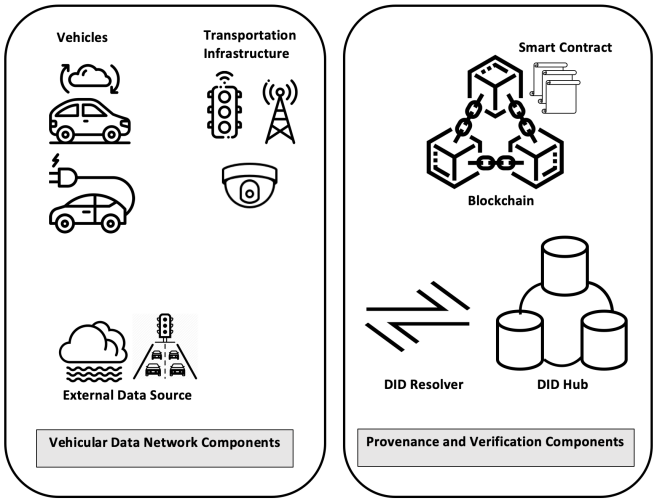

Figure 7.1: Verifiable Data Paths Network Components

- Intelligent Transportation Infrastructure: This component includes different systems in the infrastructure such as traffic lights, roadside units (RSUs), parking meters, tolling stations, and others. These components also own their DIDs and can originate data, aggregate it, or forward it.

- External Data Sources: This component pertains to data sources used to augment the measured and collected vehicular data, e.g., weather information and traffic patterns.

- Blockchain: Blockchain usage through smart contracts enables building a data chain and provide the provenance feature. Data created are hashed and stored on-chain, and all subsequent transformations are linked and stored to provide an auditing trail.

- DID/VC infrastructure: DIDs are the foundational component to offer unique global identifiers for all actors, while VCs are the component used to prove the authenticity of data records. Each data record or set has a verifiable credential issued by, and link to the actor DID. This component includes DIDs hubs and resolvers.

7.2.2 Verifiable Data Path

DIDs and VCs are the cornerstones for establishing secure and provenant data chains. To achieve global provenance, each sensor with processing capability is assigned a unique DID at manufacturing, and each processing unit, including the vehicle itself, is also assigned a DID. DIDs are also assigned to infrastructure elements in case of V2X data exchange, and to complete the path also entities and processing engines (a data pre-processing or ML algorithm) should be assigned their respective DIDs, as indicated in Figure 7.2.

It is also assumed that entities can encompass other entities, e.g., a vehicle containing a set of sensors and subsystems. Thus, DIDs are issued with support for sub-identities, where a DID might contain a parent DID, and child DID fields or both. The DIDs, as shown in Figure 7.2, can be based on different methods to ensure working cross-organizationally. The main requirement is the accessibility of each DID method resolver to the participating verifiers.

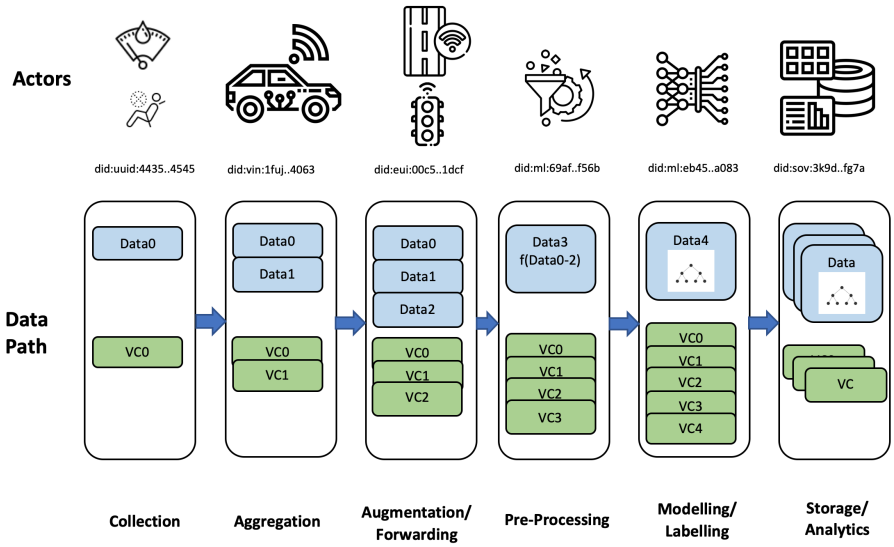

Figure 7.2: A Verifiable Mobility Data Path.

The issued DID for each actor in the data path contains the typical fields for any DID in addition to the parent DID and child DID if they exist. An example of a vehicle DID is given below. The DID also contains the proof field that is digitally signed by the issuer key (e.g., manufacturer) to ensure the integrity of the DID document.

```
{
  "@context": "https://w3id.org/did/v1",
  "id": "did:vin:1fuj..4063",
  "publicKey": [{
       "type": "Secp256k1VerificationKey2018",
       "owner": "did:vin:1fuj..4063",
       "ethereumAddress": "0x2dE5...A8Be"}],
  "authentication": [{
       "type": "Secp256k1SignatureAuthentication2018",
       "publicKey": "did:vin:0x2dE5...A8Be#owner"}],
  "childDID": [{"id": "did:uuid:4435..4545"}],
  "proof":{
        "type": "Secp256k1VerificationKey2018",
        proofPurpose: "assertionMethod",
        "created": "2019-12-23T14:53:03Z",
        "verificationMethod": "did:ethr:0x2fa3..4a62#issuer",
        "jws": "eyJhbGciOi...tx0dPfDi0"}
  "created": "2019-12-23T14:53:03Z",
  "updated": "2019-12-23T14:53:03Z"
}
```

7.2.3 Verifiable Data Claims

Generating a verifiable data path is achieved through a series of steps that create a verifiable claim (VC) for each data record or set of records. Deciding on the number of records that share a VC depends on the data record size and frequency of data generation. Each VC has a size of a few hundred bytes, thus adding a VC for each data record might constitute a significant overhead. The generation of a VC for a set of data records utilizes DIDs, Blockchain, and cryptography functions represented by hashing and digital signature; the complete flow is described hereafter and illustrated in Figure 7.3.

Generating Claims

Producing a variable dataset starts by combining a set of records and generate a hash value to ensure the integrity of the data records. Proof of issuance is produced and represented by the digital signature of the RecordHash using the actor DID private key. Therefore, for a set of collected data records $\{D_0, D_1, ...D_n\}$ the above-mentioned steps can be summarized by the following equations.

$$\left.\begin{aligned} RecordHash &= SHA_{256}(\ D_0, D_1, \ldots, D_n) \\ Proof &= Sign_{actorPrivateKey}(RecordHash) \end{aligned}\right\} \tag{7.1}$$

The third required field in a VC is the last transaction hash, which is used to build the auditing trail. The three fields are used to generate a VC in addition to a metadata field listing the main properties of the data set. Upon generating the VC, the hash of the VC is calculated, which is then committed on-chain to ensure the integrity of the chain of proofs, thus providing the feature of provenance.

Verifying Claims

On the consuming side, once a dataset is obtained by an entity, the origin and authenticity of the data are verified by first hashing the dataset and verifying it against the latest VC. For a transformed dataset, multiple VCs exist, and the process requires verifying the signature of the proof of each VC against the DID of the corresponding actor.

To ensure that the data throughout the path has not been manipulated, a VC shall contain a link to the previously issued VC represented by the transaction hash. Thus, by verifying all the transactions, it ensures that the trail of transformation is authentic. So, each VC ensures that the action performed is actually performed by the same actor according to the DID, while the chain of transactions hashes confirm that the path is intact.

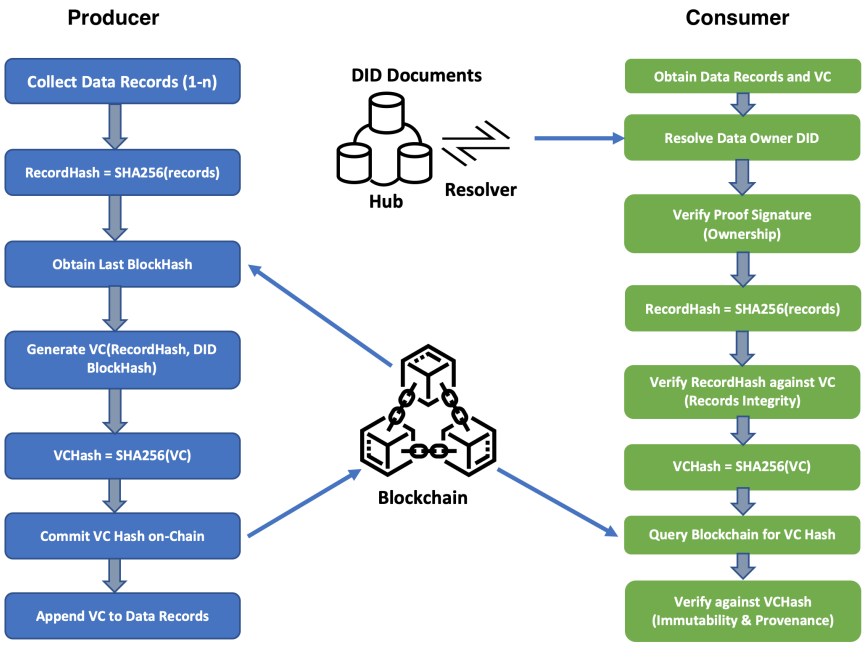

Figure 7.3: Verifiable Data Claims Processing.

A generated VC contains mandatory fields used to prove the integrity and ownership and to enable extracting the provenance trace. An example of a VC created for an accelerometer measurement record is shown below.

- Issuer: linking the VC to the issuer DID.
- Proof: containing the digital signature of the data records hash using the actor's private key, represented in JSON Web Signature (JWS) format.
- Transaction Hash: the last transaction hash is used to build an immutable data chain.

```
{
  "@context": "https://www.w3.org/2018/credentials/v1",
  "id": "http://server/credentials/1872",
  "type": ["VerifiableCredential", "DataCredential"],
  "credentialSubject": { "id": "http://server/records/5832",
    "issuer": "did:imei:3520..0826",
    "issuanceDate": "2019-12-24T11:33:24Z",
    "RecordHash" : "62c3...fbde" },
  "proof": { "type": "Secp256k1VerificationKey2018",
    "created": "2019-12-24T11:33:24Z",
    "proofPurpose": "assertionMethod",
    "verificationMethod":"did:imei:3520..0826",
    "transHash" : "0x2c34...8df2",
    "jws": "eyJh...Kb78" }
}
```

7.3 Linked Data Proofs

Building a provenance trace and a data chain requires linking the verifiable claims issued for the data set and its derivatives. The suggested approach allows the extraction of the provenance trace by linking the data proofs using the hash of the previous VC. Upon querying the blockchain for a specific VC hash, the returned object contains the previous VC id, which is a URL to access the VC. Repeatedly, the verifier pulls the dataset using the id record in the *credentialsSubject* field hash it and verifies it against the VC proof. The process continues until reaching the original dataset for which the corresponding VC record on-chain has a null value for the URL, indicating it is the genesis VC.

Since certain datasets may propagate through different actors while remaining the same, building a trace of data provenance requires the capability to authenticate the integrity and ownership of the dataset with a compact representation. A VC representation, in this case, can include instead of the proof field a proof chain field. A proof chain is useful to allow validating a data set that has been signed by multiple entities while the order of proofs generation is important.

The flow for performing the procedure to verify the provenance of a data set is illustrated in Figure 7.4. The illustration describes the steps for an example of an open data set generated by several sensors owned by entity A, e.g., a vehicle. The collected data is pre-processed (e.g., handling missing values) by entity B. The dataset is later used by entity C to run a specific machine learning algorithm (e.g., k-mean clustering) and generate a labeled dataset. A user that is accessing the labeled dataset runs the steps in the procedure to verify the data authenticity along the whole data path to its provenance.

The next section describes an example to demonstrate the applicability of the proposed concept. This example is not intended to include thorough performance analysis of the

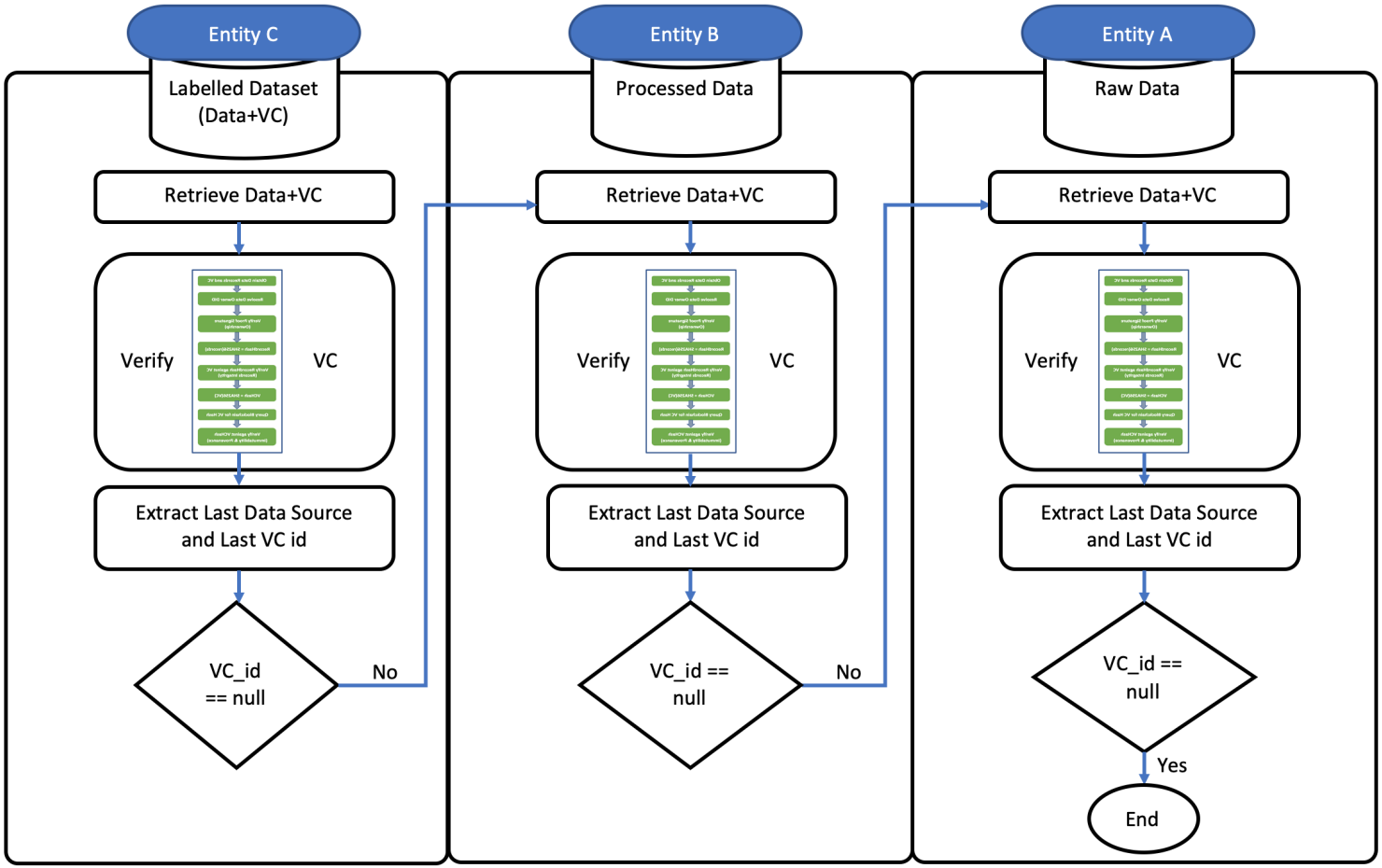

Figure 7.4: Procedure of Verifying Data Provenance through Linked Data Proofs.

concept. Nevertheless, it aspires to present evidence on the extendability of the thesis suggested framework in building decentralized IoT applications.

7.4 Smart Mobility Example

This section describes a simple setup using a crowdsensing scenario where users' smartphones, equipped with accelerometers, are used to collect the acceleration reading on 3-axis while driving a vehicle around the city. To emulate the suggested scenarios, four different smartphones were mounted in separate trips on a vehicle to collect the measurements, which are assumed to be used to build a model for road surface conditions and pothole detection. In the experiment, each smartphone is issued a DID using its IMEI identifier and the raw data gathered by the accelerometer is stamped with the DID issued to each smartphone.

Three separate trips were made by riding the city bus and assuming that the city performed a pre-processing task to clean any missing values before publishing the data. The pre-processed accelerometer readings on bus trips are signed by the city DID.

To show how a provenance test can run, a separate set of accelerometer readings collected in a scenario where the smartphone device is placed in a user's pocket while running. This set of records where uploaded as part of the crowdsensed data with an invalid proof field and digital signature in the verifiable claim.

For the purpose of demonstrating the entire process, a simple algorithm was used to generate a labeled data set for collected data. A z-threshold algorithm is used that classifies any measured value on the z-axis reading above 1g (9.8m/s^2) as a pothole. The algorithm uses the data collected from all trips, including the injected data collected from

a smartphone while the user is running.

The experiment was conducted over a route of a length of 3.2 km starting from the location given by the coordinates (43.45259816,-80.55250328) to the point (43.47029615,-80.53797224), as indicated in Figure 7.5. The process of generating data claims and verifying them includes:

- Collect data records over a single trip by a smartphone.
- Upon finishing a trip, the data records are saved in a CSV file an d then hashed, and a VC is created using the smartphone DID attributes.
- Both the data records and the generated VC are uploaded, and the VC hash is committed on-chain.

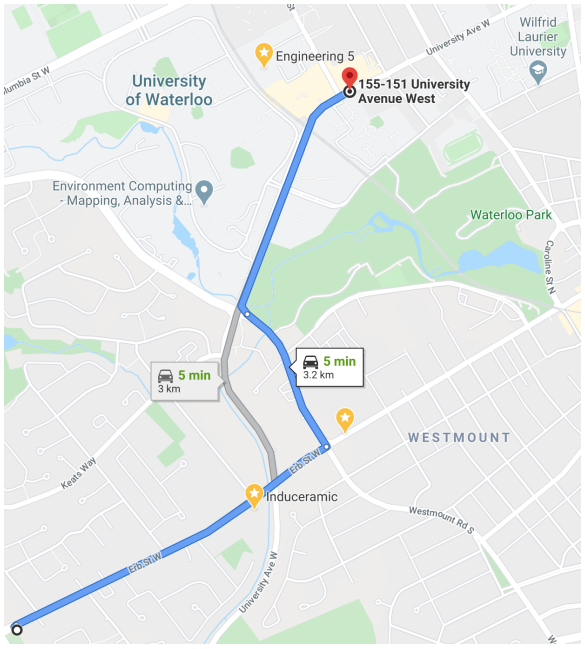

Figure 7.5: Map Showing The Path Used for Car and Bus Trips to Collect Accelerometer Readings.

- Similarly, the corresponding VCs for the bus trips data and the data collected for a running user are generated and uploaded.
- Before running the z-threshold algorithm, all data records and corresponding VCs are verified.
- Given that the list of participants is known and the corresponding DIDs are known, the verification step identifies the misleading data records collected by a different smartphone.
- The z-threshold algorithm, which is owned by a certain research entity owning a specific DID, processes the data and generates a labeled dataset identifying each coordinate in the path as if it contains a pothole or not.
- The generated labeled set is saved, and a VC is generated using the same steps.

Based on this experiment, it is noted that collecting small records from several sensors and aggregating the data to build a dataset can grow in size. Since each smartphone has its unique DID, thus each trip data is issued a VC proof signed by the specified DID. Therefore, collecting measurements in (n) number of trips means creating a new dataset that has (n) different VCs.

Based on the results, for the trip specified above, each data record has an overhead, due to VC attachment, of approximately 1.75 KBytes representing around 7-8% of the original data record size. Thus, building a dataset based on (n) measurements that include all VCs of each record will increase the size to grow to n*1.76 KBytes, yet it still contributes to an overhead of 7-8%. That is attributed to the fact that using the data record hash to build the VC makes the VC size almost fixed with a small margin, which depends on the URL length.

7.5 COVID-19 Contact Tracing Example

Late 2019, a new strand of the Corona virus family, called COVID-19, spread in specific Chinese cities to spread later across the globe. The WHO has announced the COVID-19 outbreak a pandemic [124]. Among the main measures, imposed internationally, to limit the spread and flatten the curve of number of cases is social distancing. The social distancing measure requires to maintain a distance of 2 metres (6 feet) between people in public places. Though, adherence by the public due for instance to the impossible complete shutdown limited the effectiveness of this measure. Furthermore, the pattern of virus spread showed that it can long persist on surfaces and it can be transferred through asymptomatic patients [125].

With the limitations and difficulties of imposing social distancing, other ways to trace infected people and who they came close to become important. Countries like China and Singapore used widely contact tracing to be able to identify people at risk of capturing the virus. Though, concerns on privacy mounted as people felt a very high level of surveillance [126].

As the outbreak spread to more countries, the tech-community worldwide stepped to propose solutions for virus spread prediction and contact tracing. A noteworthy project is the PrivateKit SafePath mobile app by MIT researchers [127]. The PrivateKit application records all user movements by continuously recording the GPS coordinate. In an interview with IEEE spectrum, it has been criticised that such an invasive approach can risk the users' privacy, and the MIT researchers responded that data is encrypted and not readable even by the cloud host [128].

7.5.1 Application Design

In addressing these issues, the author, in this example, demonstrates how DIDs and VCs applied through the concept of verifiable data path can solve these issues. The example is demonstrated using a mobile app that the users opt-in to record places visited including trips made on public transportation. The user is issued a DID that is linked to no other physical identities and all data collected is only associated to the user DID thus ensuring that all data is anonymous. The complete flow is described in the following steps:

- The user adds points that include geographic coordinates and a timestamp and all data is saved internally on the user device.
- Once the user tested positive or suspected being exposed to the virus, the user can upload the data to the cloud.
- Once the user issues the upload the command, the process described in 7.3 is followed. The data records for traces of user whereabouts is first hashed and VC is generated using the user DID.
- The VC hash is committed on-chain and appended to the data records to be submitted and stored in the cloud.
- The flow assumes an authority responsible on further verifying uploaded data specifically positive cases. This entity can be a public health department.
- The public health department issues a VC using the department DID that is appended to the data records to a linked data proof.

7.5.2 Establishing The Root of Trust

Among the major issue of contact tracing applications is the trustworthiness of data uploaded by the users. A false positive case that is claimed by a user results in false notifications for quarantine and self-isolation for people who came in close contact with the falsely claimed case. Therefore, the suggested approach proposed the use of DIDs and VCs to eliminate this issue, and to further enhance the trustworthiness of the data the following system setup is suggested.

The suggested setup looks at the main stakeholders in the process; the user and the public health departments. Once a DID is generated for the user inside the application, the health department issues a verifiable credential for the user DID to prove the validity of the DID.

In addition, as mentioned above, the uploaded data records for positive cases are amended with a VC issued by the health authority. Thus, the trust is established through

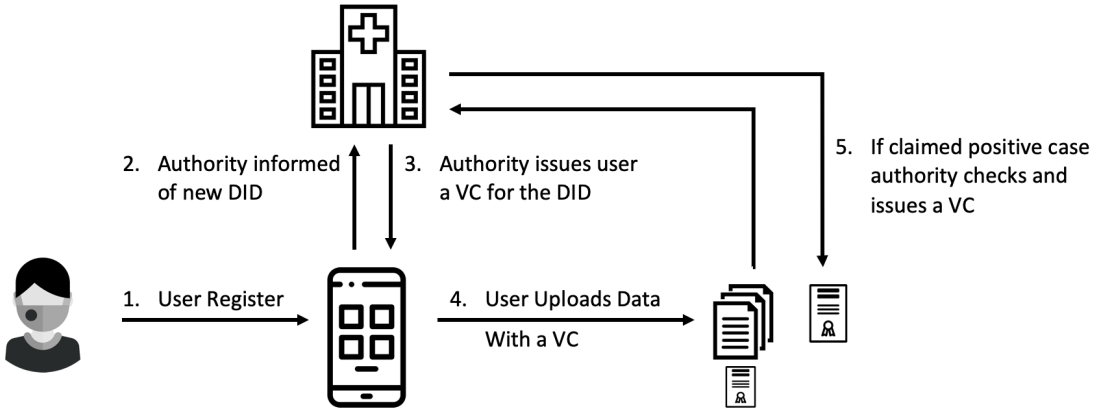

Figure 7.6: The Flow of Verifiable Credentials Generation for COVID-19 Application of Contact Tracing.

verifying that the data is issued by a trusted DID and by verifying that claimed positive cases include a chain proof that includes two VCs, one issued by the user and one amended by the health authority. Figure 7.6 depicts the trust links and the listing below shows an example of a data record with a chain of proofs.

7.6 Summary

In this thesis, it has been shown that decentralized IoT devices' identities are crucial for building decentralized authentication and access management. However, this chapter describes that the exploitation of this foundation layer is not limited to the core device management functionalities, and it is extended to enable decentralized services in the application layer and specifically to enable building verifiable data paths. The proposal describes an example of a mobility network scenario though this applies to any data path, including medical and health IoT networks and supply chain and industrial IoT networks.

The concept builds on the usage of DIDs, verifiable claims, and the immutability of Blockchain records to build a data chain that is traced to the origin of the data derivative. Moreover, a provenance and verifiable trace enable securing the data path from illegitimate manipulation of data records and possible injection of inauthentic records.

Chapter 8

Conclusion and Future Work

This chapter provides a brief summary of the thesis and future directions. Section 8.1 concludes the thesis, and Section 8.2 highlights the main points for research opportunities to extend this work further.

8.1 Conclusion

Recent developments in IoT networks promise an ecosystem where devices have the ability to interact and communicate with each other in a peer-to-peer model without a centralized authority that plays the role of identifying and authenticating the devices and authorizing access to offered services. Furthermore, the challenges imposed by the complexity of heterogeneous IoT networks stimulated research interest in building decentralized architectures focusing on enabling decentralized device management functionalities as a foundation that can be extended for offering decentralized services and applications.

The work presented in this thesis describes the development of a decentralized IoT

devices identity and access management framework that can be used in enabling peer-to-peer and decentralized IoT applications. The work in this thesis addresses three main challenges:

- IoT networks are characterized by being heterogeneous, and most of these networks are built with fragmented identity management structures. These factors impose a challenge on the capability of enabling user/device-centric services.
- The mobility of IoT devices requires a global and unique identification scheme supporting identity portability and self-ownership with the capability of maintaining the history of the device life cycle events.
- The growth in adopting IoT applications based on peer-to-peer models highlighted the need for authentication and authorization schemes with no middleman involved.

The presented work defines a framework that has a foundation of decentralized identity management. The identity framework is built using W3C specifications of Decentralized Identifiers (DID) and Verifiable Claims (VCs). An IoT DID method has been devised describing the CRUD operations of IoT devices identities. The work of the thesis utilizes the DID method in drafting a mutual authentication scheme that involves no central authority with perfect forward secrecy.

In addressing the challenge of enabling a decentralized access control mechanism, the thesis work presented an authorization approach by mapping the capability-based access control model (CapBAC) to the non-fungible (ERC721) blockchain tokenization standard. The approach demonstrates how access tokens are designed to be a self-contained policy control function and can be defined as collectibles that support an economy of exchange

and trading for access tokens. The fungible (ERC20) token is utilized as a facility for services exchange that can be performed with accounting and charging support.

Blockchain main challenges are related to its distributed nature, which is mainly introducing delay and overhead. The work evaluates the proposed access model using a testbed, where the results show the feasibility of the suggested approach with further investigation required to compare the performance when the same approach is implemented on top of PoS and DPoS consensus algorithms. The data overhead issue observed in the testbed results also requires an assessment of a trade-off between embedding the token as a whole or just the token ID and offloading the token retrieval task to a gateway. Such an approach is expected to introduce communication overhead since the token need to be retrieved in each request.

The suggested framework has been applied in two use cases that include a suggestion for an improvement of the IMEIDB system used to combat counterfeiting and loss and theft of smartphones. Another supply chain use case addressed building a tracing and tracking system to adhere to the recent FMD and DSCSA drug regulations by utilizing composable non-fungible tokens to track pharmaceutical products on both unit and lot level.

The capability of extending the framework for the application layer and addressing data ownership and authenticity was addressed through the data provenance concept in which a verifiable chain of data for IoT data is built utilizing devices decentralized identities and capability of generating linked data proofs.

8.2 Future Work

Decentralized IoT device identity and access management field continues to receive attention in both academia and industry and represents an active area of research and development. The tendency for decentralized approaches and offering device owners more control over their data brings many new challenges. Though, as the Blockchain technology matures and more IoT business use cases emerge, new research venues for extending this work are sought.

8.2.1 Permissioned and Proof-of-Stake DLTs

This thesis work is based on utilizing a permissionless DLT platform that uses the PoW consensus mechanism. The main challenges for such a platform are the speed, and the privacy concern, which permissioned and Proof-of-Stake (PoS) promise to resolve. PoS is considered to be a primary consensus algorithm for most public Blockchain to replace PoW, including Etherum, which will help in speeding up the number of transactions per second and provide the capability to host applications for larger IoT networks. On the other hand, permissioned DLTs such as Hyperledger Fabric provide a setup that offers enterprises higher throughput and more privacy.

Migrating the framework to PoS once it is released is a straightforward extension and enhancement of this work. Furthermore, particular use cases would require migrating this framework to permissioned DLT platforms. The thesis author has joined the Hyperledger Telecom Special Interest Working group to extend this work to these platforms [105].

8.2.2 Privacy and Anonymity

Blockchain technology was built to enhance trust and transparency, though some applications require the protection of participants' privacy and having anonymous identities. The field of zero-knowledge proof (ZKP) is a research topic that is sought to be a venue for extending this work.

While using DIDs and Smart contracts in this work presented the capability to build a unique global identity, though, as devices start to transact, identity traces can be a source of a breach for devices' privacy. ZKP algorithms enable two entities to successfully exchange information and transact without the need to expose sensitive data. Obscuring IoT devices sensitive identification information is a very active research domain. This work provided a decentralized IoT devices identity approach using DIDs and VCs, which the later possess many attributes to extend it into a ZKP authentication framework. Verifiable claims can be extended to a ZKP proof format using features such as selective disclosure to present the minimal set of data for verification purposes [129]. Notable research challenges in this regard include designing ZKP algorithms that are lightweight, and among them non-interactive zero-knowledge proofs NIZKP [120, 52].

8.2.3 Reputation Management

In decentralized systems, the absence of a middleman or an authority requires a mechanism for establishing the root of trust. It has been highlighted in this work that the usage of Blockchain and DIDs and VCs helps in establishing a root of trust. However, an extension to this work would include defining a reputation management framework that offers a scoring mechanism to permit devices to evaluate reliability of other devices and establish a score for each device that is used as a factor in granting or revoking access requests.

References

[1] " Cisco Annual Internet Report (2018 - 2023) ," tech. rep., Cisco, 2019.

[2] " Ericsson Mobility Report 2018 ," tech. rep., Ericsson, Nov. 2018.

[3] "Ericsson Mobility Report 2016," tech. rep., Ericsson, June 2016.

[4] B. Safaei, A. M. H. Monazzah, M. B. Bafroei, and A. Ejlali, "Reliability side-effects in Internet of Things application layer protocols," in *2017 2nd International Conference on System Reliability and Safety (ICSRS)*, pp. 207–212, Dec 2017.

[5] S. Popov, "The Tangle, IOTA white paper," tech. rep., IOTA Foundation, Oct 2017.

[6] "Grid+, Welcome to the future of energy, white paper," tech. rep., ConsenSys, 2018.

[7] N. N. Henri Pihkala, Risto Karjalainen and M. Malka, "Unstoppable data for unstoppable apps: Datacoin by streamr," tech. rep., Streamr, July 2017.

[8] "Energy savings from the nest learning thermostat: Energy bill analysis results," tech. rep., Nest Labs, Feb 2015.

[9] J. V. Wagenen, "Smart street lights lay the groundwork for future iot deployments," Oct 2017. [Online]. Available: `https://statetechmagazine.com/article/2017/10/smart-street-lights-lay-groundwork-future-iot-deployments`.

[10] C. Lawrence, "How data monetization is creating a new data economy for iot," Mar 2019. [Online]. Available: https://dzone.com/articles/how-data-monetization-is-creating-a-new-data-econo.

[11] "Internet of things (iot) platform market," tech. rep., Kenneth Research, July 2019.

[12] "IoT Security Market Report 2017-2022," tech. rep., IoT Analytics, Sep 2017.

[13] "Inside the infamous Mirai IoT Botnet: A retrospective analysis," Dec 2017. [Online]. Available: https://blog.cloudflare.com/inside-mirai-the-infamous-iot-botnet-a-retrospective-analysis/.

[14] U. Gerber, "Authentication and Authorization for Constrained Environments," Master's thesis, University of Zurich, 2018.

[15] L. Cruz-Piris, D. Rivera, I. Marsa-Maestre, E. D. la Hoz, and J. R. Velasco, "Access control mechanism for iot environments based on modelling communication procedures as resources," *Sensors*, vol. 18, p. 917, Mar 2018.

[16] J. L. Hernández-Ramos, M. P. Pawlowski, A. J. Jara, A. F. Skarmeta, and L. Ladid, "Toward a lightweight authentication and authorization framework for smart objects," *IEEE Journal on Selected Areas in Communications*, vol. 33, pp. 690–702, April 2015.

[17] R. Venkatesan, M. V. Raghavan, and K. S. S. Prakash, "Architectural considerations for a centralized global iot platform," in *2015 IEEE Region 10 Symposium*, pp. 5–8, May 2015.

[18] R. Wattenhofer, *Distributed Ledger Technology: The Science of the Blockchain*. North Charleston, SC, USA: CreateSpace Independent Publishing Platform, 2nd ed., 2017.

[19] N. Gershenfeld, R. Krikorian, and D. Cohen, "The Internet of Things.," *Scientific American*, vol. 291 4, pp. 76–81, Oct 2004.

[20] Z. Shelby, K. Hartke, and C. Bormann, "The Constrained Application Protocol (CoAP)." RFC 7252, June 2014. [Online]. Available: `https://rfc-editor.org/rfc/rfc7252.txt`.

[21] U. Hunkeler, H. L. Truong, and A. J. Stanford-Clark, "Mqtt-s - a publish/subscribe protocol for wireless sensor networks.," in *COMSWARE* (S. Choi, J. Kurose, and K. Ramamritham, eds.), pp. 791–798, IEEE, 2008.

[22] "Lightweight machine to machine technical specification: Core v1.1," tech. rep., Open Mobile Alliance OMA, Aug 2018.

[23] D. Recordon and D. Reed, "Openid 2.0: A platform for user-centric identity management," in *Proceedings of the Second ACM Workshop on Digital Identity Management*, DIM '06, (New York, NY, USA), p. 11–16, Association for Computing Machinery, 2006.

[24] "NGN identity management framework," tech. rep., ITU, Jan 2009.

[25] H. Aftab, K. Gilani, J. Lee, L. Nkenyereye, S. Jeong, and J. Song, "Analysis of identifiers in IoT platforms," *Digital Communications and Networks*, 05 2019.

[26] M. Abu-Elkheir, M. Hayajneh, and N. Ali, "Data management for the internet of things: Design primitives and solution," *Sensors*, vol. 13, p. 15582–15612, Nov 2013.

[27] " TS.06 - IMEI Allocation and Approval Process, Version 17.0," tech. rep., GSM Association, Jun 2019.

[28] " 3GPP TS 22.016: International Mobile station Equipment Identities (IMEI), Release 15," tech. rep., 3rd Generation Partnership Project, Jun 2018.

[29] C. A. Repec and M. Harrison, " EPC Tag Data Standard, Version 1.13 ," tech. rep., GS1, Nov 2019.

[30] P. J. Leach, R. Salz, and M. H. Mealling, "A Universally Unique IDentifier (UUID) URN Namespace." RFC 4122, Jul 2005. [Online]. Available: `http://www.ietf.org/rfc/rfc4122.txt`.

[31] "Bluetooth SIG Specifications." [Online]. Available: `https://www.bluetooth.com/specifications/`.

[32] B. Hinden, "IP Version 6 Addressing Architecture." RFC 2373, July 1998. [Online]. Available: `https://rfc-editor.org/rfc/rfc2373.txt`.

[33] "Lorawan^{TM} 1.1 specification," tech. rep., LoRa Alliance, Oct 2017.

[34] "The design principles of identity relationship management. v1.0," tech. rep., Kantara Initiative, Inc, Feb 2015.

[35] D. Hardt, "The OAuth 2.0 Authorization Framework." RFC 6749, Oct. 2012. [Online]. Available: `https://rfc-editor.org/rfc/rfc6749.txt`.

[36] V. Mladenov, C. Mainka, J. Krautwald, F. Feldmann, and J. Schwenk, "On the security of modern single sign-on protocols: Openid connect 1.0," *CoRR abs/1508.04324*, 2015.

[37] L. Seitz, S. Gerdes, G. Selander, M. Mani, and S. Kumar, "Use Cases for Authentication and Authorization in Constrained Environments." RFC 7744, Jan. 2016. [Online]. Available: `https://rfc-editor.org/rfc/rfc7744.txt`.

[38] I. Indu, P. R. Anand, and V. Bhaskar, "Identity and access management in cloud environment: Mechanisms and challenges," *Engineering Science and Technology, an International Journal*, vol. 21, no. 4, pp. 574 – 588, 2018.

[39] S. Rahimi Moosavi, T. Nguyen Gia, A.-M. Rahmani, E. Nigussie, S. Virtanen, J. Isoaho, and H. Tenhunen, "Sea : A secure and efficient authentication and authorization architecture for iot-based healthcare using smart gateways," *Procedia Computer Science*, vol. 52, pp. 452–459, 2015. QC 20150618.

[40] C.-Y. Chang, C.-H. Kuo, J.-C. Chen, and T.-C. Wang, "Design and implementation of an iot access point for smart home," *Applied Sciences*, vol. 5, p. 1882–1903, Dec 2015.

[41] "P2P Weakness Exposes Millions of IoT Devices." [Online]. Available: `https://krebsonsecurity.com/2019/04/p2p-weakness-exposes-millions-of-iot-devices/`.

[42] "IBM Watson Role-base Access Control." [Online]. Available: `https://www.ibm.com/support/knowledgecenter/SSQP8H/iot/platform/reference/rbac/index.html`.

[43] B. K. Mohanta, S. S. Panda, and D. Jena, "An overview of smart contract and use cases in blockchain technology," in *2018 9th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*, pp. 1–4, July 2018.

[44] V. Buterin, "Ethereum: A next-generation smart contract and decentralized application platform," tech. rep., Ethereum Foundation, Jan 2014.

[45] G. Wood, "Ethereum: A secure decentralised generalised transactionledger," tech. rep., Ethereum Foundation, 2014.

[46] F. Vogelsteller and V. Buterin, "Eip 20: Erc-20 token standard," tech. rep., Ethereum Foundation, Nov 2015.

[47] W. Entriken, D. Shirley, J. Evans, and N. Sachs, "Eip 721: Erc-721 non-fungible token standard," tech. rep., Ethereum Foundation, Jan 2018.

[48] X. Zhu and Y. Badr, "A Survey on Blockchain-Based Identity Management Systems for the Internet of Things," 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)*, pp. 1568–1573, 2018.

[49] MustafaAl-Bassam, "SCPKI: A Smart Contract-Based PKI and Identity System," in *Proceedings of the ACM Workshop on Blockchain, Cryptocurrencies and Contracts*, BCC '17, (New York, NY, USA), p. 35–40, Association for Computing Machinery, 2017.

[50] Y. Liu, Z. Zhao, G. Guo, X. Wang, Z. Tan, and S. Wang, "An Identity Management System Based on Blockchain," in *2017 15th Annual Conference on Privacy, Security and Trust (PST)*, pp. 44–4409, Aug 2017.

[51] L. Axon and M. Goldsmith, "PB-PKI: A Privacy-aware Blockchain-based PKI," in *Proceedings of the 14th International Joint Conference on e-Business and Telecommunications - Volume 4: SECRYPT, (ICETE 2017)*, pp. 311–318, INSTICC, SciTePress, 2017.

[52] T. Hardjono and A. Pentland, "Verifiable anonymous identities and access control in permissioned blockchains," *CoRR*, vol. abs/1903.04584, 2019.

[53] J. Chik, "Decentralized identity and the path to digital privacy." [Online]. Available: https://www.microsoft.com/security/blog/2019/05/15/decentralized-identity-digital-privacy/.

[54] "Evernym — The Self-Sovereign Identity Company." [Online]. Available: https://www.evernym.com/.

[55] "Chronicled: Linking the physical world to the blockchain." [Online]. Available: http://www.chronicled.com.

[56] D. Reed, M. Sporny, D. Longley, C. Allen, R. Grant, and M. Sabadello, "Decentralized identifiers (dids) v1.0: Core architecture, data model, and representations," tech. rep., World Wide Web Consortium (W3C), Mar 2019. [Online]. Available: https://w3c.github.io/did-core/.

[57] "Uport - tools for decentralized identity." [Online]. Available: https://www.uport.me/.

[58] "Sovirn foundation." [Online]. Available: = https://sovrin.org/.

[59] P. Braendgaard and J. Torstensson, "Erc1056: Ethereum lightweight identity," tech. rep., Ethereum Foundation, May 2018.

[60] T. Berners-Lee, R. T. Fielding, and L. M. Masinter, "Uniform Resource Identifier (URI): Generic Syntax." RFC 3986, Jan. 2005. [Online]. Available: https://rfc-editor.org/rfc/rfc3986.txt.

[61] M. Sporny, D. Longley, and D. Chadwick, "Verifiable credentials data model 1.0: Expressing verifiable information on the web," tech. rep., World Wide Web Consortium (W3C), Mar 2019. [Online]. Available: https://www.w3.org/TR/vc-data-model/.

[62] "IoTeX DID Method Specification," Sept. 2019. [Online]. Available: https://github.com/iotexproject/iotex-did/blob/master/README.md.

[63] "Ockam DID Method Specification," Nov. 2018. [Online]. Available: https://github.com/ockam-network/did-method-spec.

[64] V. A. Siris, D. Dimopoulos, N. Fotiou, S. Voulgaris, and G. C. Polyzos, "OAuth 2.0 meets Blockchain for Authorization in Constrained IoT Environments," in *2019 IEEE 5th World Forum on Internet of Things (WF-IoT)*, pp. 364–367, April 2019.

[65] "A proposal for a new work item on Feasibility of Decentralised Identifiers (DIDs) in IoT," Nov. 2019. [Online]. Available: https://www.itu.int/md/T17-SG20-C-0615/en.

[66] "Hyperledger Telecom Special Interest Group - IoT Subgroup," Oct. 2019. [Online]. Available: https://wiki.hyperledger.org/display/TCSIG/IOT+Sub+Group.

[67] M. Nakhjiri and M. Nakhjiri, *AAA and Network Security for Mobile Access: Radius, Diameter, EAP, PKI and IP Mobility*. John Wiley Sons, Ltd, Oct 1993.

[68] "General Data Protection Regulation," tech. rep., European Commission, May 2018.

[69] O. Liberg, M. Sundberg, E. Wang, J. Bergman, and J. Sachs, *Cellular Internet of things: technologies, standards, and performance*. Academic Press, Sep 2017.

[70] R. Xu, Y. Chen, E. Blasch, and G. Chen, "Blendcac: A blockchain-enabled decentralized capability-based access control for iots," 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)*, pp. 1027–1034, July 2018.

[71] R. S. Sandhu, "Role-based Access Control," vol. 46 of *Advances in Computers*, pp. 237 – 286, Elsevier, 1998.

[72] S. I. Gavrila and J. F. Barkley, "Formal Specification for Role Based Access Control User/Role and Role/Role Relationship Management," in *Proceedings of the Third ACM Workshop on Role-Based Access Control*, RBAC '98, (New York, NY, USA), p. 81–90, Association for Computing Machinery, 1998.

[73] Guoping Zhang and Jiazheng Tian, "An extended role based access control model for the internet of things," in *2010 International Conference on Information, Networking and Automation (ICINA)*, vol. 1, pp. V1–319–V1–323, Oct 2010.

[74] J. L. H. Ramos, A. J. Jara, L. Marín, and A. F. Gómez-Skarmeta, "Distributed capability-based access control for the internet of things," *Journal of Internet Services and Information Security (JISIS)*, vol. 3, pp. 1–16, 2013.

[75] A. Ouaddah, A. A. Elkalam, and A. A. Ouahman, "FairAccess: a new Blockchain-based access control framework for the Internet of Things," *Security and Communication Networks*, vol. 9, pp. 5943–5964, dec 2016.

[76] R. Xu, Y. Chen, E. Blasch, and G. Chen, "A federated capability-based access control mechanism for internet of things (iots)," *CoRR*, vol. abs/1805.00825, 2018.

[77] S. Gusmeroli, S. Piccione, and D. Rotondi, "A capability-based security approach to manage access control in the Internet of Things," *Mathematical and Computer Modelling*, vol. 58, no. 5, pp. 1189 – 1205, 2013.

[78] A. Ouaddah, H. Mousannif, A. A. Elkalam, and A. A. Ouahman, "Access control in the internet of things: Big challenges and new opportunities," *Computer Networks*, vol. 112, pp. 237 – 262, 2017.

[79] "ISO/IEC 24760-1 Information technology — Security techniques — A framework for identity management — Part 1: Terminology and concepts," tech. rep., International Organization for Standardization, Dec 2011.

[80] A. S. Omar and O. Basir, "Identity Management in IoT Networks Using Blockchain and Smart Contracts," in *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), iThings/GreenCom/CPSCom/SmartData 2018, Halifax, NS, Canada, July 30 - August 3, 2018*, pp. 994–1000, IEEE, 2018.

[81] "InterPlanetary File System - IPFS." [Online]. Available: `https://ipfs.io/`.

[82] E. Nyaletey, R. M. Parizi, Q. Zhang, and K.-K. R. Choo, "Blockipfs - blockchain-enabled interplanetary file system for forensic and trusted data traceability," in *2019 IEEE International Conference on Blockchain (Blockchain)*, pp. 18–25, 2019.

[83] Y. Cao and L. Yang, "A survey of Identity Management technology," in *2010 IEEE International Conference on Information Theory and Information Security*, pp. 287–293, Dec 2010.

[84] "IoT Security Compliance Framework: Release 2," tech. rep., Internet of Things Security Foundation, Dec 2018.

[85] L. Chen and G. Gong, *Communication System Security*. Chapman Hall/CRC, 1st ed., 2012.

[86] E. Barker, L. Chen, and R. Davis, "Recommendation for Key-Derivation Methods in Key-Establishment Schemes," tech. rep., National Institute of Standards Technology, Gaithersburg, MD, USA, Apr 2018.

[87] A. S. Omar and O. Basir, "Capability-Based Non-fungible Tokens Approach for a Decentralized AAA Framework in IoT," in *Blockchain Cybersecurity, Trust and Privacy* (K.-K. R. Choo, A. Dehghantanha, and R. M. Parizi, eds.), pp. 7–31, Cham: Springer International Publishing, 2020.

[88] M. Collina, "CoAP - Nodejs Library." [Online]. Available: `https://github.com/mcollina/node-coap`, 2016.

[89] E. Foudation, "Web3.js - Ethereum JavaScript API." [Online]. Available: `https://github.com/ethereum/web3.js/`, 2016.

[90] D. S. E. Deering and B. Hinden, "Internet Protocol, Version 6 (IPv6) Specification." RFC 8200, July 2017. [Online]. Available: `https://rfc-editor.org/rfc/rfc8200.txt`.

[91] C. Bormann, M. Ersue, and A. Keränen, "Terminology for Constrained-Node Networks." RFC 7228, May 2014.

[92] "Gartner says global smartphone demand was weak in third quarter of 2019," Nov 2019. [Online]. Available: `https://www.gartner.com/en/newsroom/press-releases/2019-11-26-gartner-says-global-smartphone-demand-was-weak-in-thi`.

[93] "IMEI Database Overview," May 2012. [Online]. Available: `https://www.gsma.com/latinamerica/wp-content/uploads/2012/05/IMEI-Database-Overview.pdf`.

[94] "Technological Advisory Council (TAC) Mobile Device Theft Prevention (MDTP) Working Group," tech. rep., Federal Communications Commission, Dec 2018.

[95] D. Protsenko, "Overview of national initiatives and solutions to combat counterfeit mobile devices," 2016. ITU Workshop on combating counterfeit using conformance and interoperability solutions, Geneva, Switzerland.

[96] " Framework for combating the importation, supply and use of counterfeit substandard terminals in the EACO member states," tech. rep., June 2016. 18th EACO Meeting, Kigali, Rawanda.

[97] "GSMA Membership Fees." [Online]. Available: https://www.gsma.com/membership/membership-fees/.

[98] "GSMA IMEI Black List," Dec 2017. [Online]. Available: https://imeishowcase.gsma.com/en#!/imeiblacklist.

[99] " The Economic Impact of Intellectual Property Rights (IPR) Infringement in Smartphone Sector ," tech. rep., European Union Intellectual Property office, Feb 2017. [Online]. Available: https://euipo.europa.eu/tunnel-web/secure/webdav/guest/document_library/observatory/resources/research-and-studies/ip_infringement/study11/smartphone_sector_en.pdf.

[100] " ETSI TS 129 272: Universal Mobile Telecommunications System (UMTS); LTE; Evolved Packet System (EPS); Mobility Management Entity (MME) and Serving GPRS Support Node (SGSN) related interfaces based on Diameter protocol ," tech. rep., European Telecommunications Standards Institute - ETSI, Oct 2014.

[101] " ETSI TS 129 511 5G System; Equipment Identity Register Services," tech. rep., European Telecommunications Standards Institute - ETSI, Oct 2018.

[102] A. S. Omar and O. Basir, "Smart phone anti-counterfeiting system using a decentralized identity management framework," in *2019 IEEE Canadian Conference of*

Electrical and Computer Engineering, CCECE 2019, Edmonton, AB, Canada, May 5-8, 2019, pp. 1–5, IEEE, 2019.

[103] A. S. Omar and O. Basir, "Decentralized Identifiers and Verifiable Credentials for Smartphone Anti-Counterfeiting and Decentralized IMEI Database," vol. 104, pp. 246–251, 01 2020.

[104] "Verifiable Organizations Network: Global Digital Trust for Organizations." `https://vonx.io/`.

[105] "Hyperledger Telecom SIG: IoT Subgroup Wiki." [Online]. Available: `https://wiki.hyperledger.org/display/TCSIG/IOT+Sub+Group`.

[106] "Impact of the Drug Supply Chain Security Act on Pharmacy Management: 2015 to 2023," tech. rep., American Society of Hospital Pharmacists (ASHP) Foundation, Mar 2015. [Online]. Available: `https://www.ashp.org/-/media/assets/practice-management/docs/sppm-dsc-dscsa-ashp-resource-paper.ashx?la=en`.

[107] "DSCSA 2020: Why Scanning 2D Barcodes is the Best Option for Pharmacies," Aug 2019. [Online]. Available: `https://www.tracelink.com/insights/dscsa-2020-why-scanning-2d-barcodes-is-the-best-option-for-pharmacies`.

[108] "2016 Top Markets Report Pharmaceuticals: Overview and Key Findings," tech. rep. [Online]. Available: note=[Online]. Available: `https://legacy.trade.gov/topmarkets/pdf/Pharmaceuticals_Executive_Summary.pdf`, institution = U.S. Department of Commerce - International Trade Administration, year = 2016,.

[109] J. Sabartova and A. Toumi, "Survey of the quality of selected antimalarial medicines circulating in six countries of sub-Saharan Africa," tech. rep., World Health Organi-

zation (WHO), Jan 2011. [Online]. Available: https://www.who.int/medicines/publications/WHO_QAMSA_report.pdf?ua=1.

[110] G. J. Buckley and L. O. Gostin, "Weaknesses in the drug distribution chain," in *Countering the Problem of Falsified and Substandard Drugs*, ch. 5, pp. 197–254, Washington D.C: National Academies Press (US), 2013.

[111] Council of European Union, "DIRECTIVE 2011/62/EU OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL," 2011. [Online]. Available: https://ec.europa.eu/health/sites/health/files/files/eudralex/vol-1/dir_2011_62/dir_2011_62_en.pdf,.

[112] U.S Food and Drug Administration, " Drug Supply Chain Security Act (DSCSA)," 2011. [Online]. Available: https://www.gpo.gov/fdsys/pkg/PLAW-113publ54/pdf/PLAW-113publ54.pdf.

[113] " IBM Food Trust. A new era for the world's food supply. ," Oct 2018. [Online]. Available: https://www.ibm.com/blockchain/solutions/food-trust.

[114] "TradeLens Platform: A Smarter Way to Engage in Trade." [Online]. Available: https://www.tradelens.com//.

[115] "LVMH, ConsenSys, Microsoft announce AURA, to power luxury industry with blockchain tech," May 2019. [Online]. Available: https://finance.yahoo.com/news/lvmh-consensys-microsoft-announce-aura-100008984.html.

[116] S. Uhlmann, " Reducing Counterfeit Products with Blockchains," Master's thesis, University of Zurich, 2017.

[117] "The MediLedger Project An Open and Decentralized Network for the Pharmaceutical Supply Chain." [Online]. Available: `[Online].Available:https://www.mediledger.com/`.

[118] A. S. Omar and O. Basir, "Secure Anti-Counterfeiting Pharmaceuticals Supply Chain System Using Composable Non-Fungible Tokens," in *Blockchain for Cybersecurity and Privacy: Architectures, Challenges, and Applications* (Y. Maleh, M. Shojafar, M. Alazab, and I. Romdhani, eds.), ch. 11, CRC Press, 2020.

[119] M. Lockyer, N. Mudge, and J. Schalm, "Erc-998 composable non-fungible token standard," tech. rep., Ethereum Foundation, July 2018. [Online]. Available: `https://github.com/ethereum/EIPs/blob/master/EIPS/eip-998.md`.

[120] M. Walshe, G. Epiphaniou, H. Al-Khateeb, M. Hammoudeh, V. Katos, and A. Dehghantanha, "Non-Interactive Zero Knowledge Proofs for the Authentication of IoT Devices in Reduced Connectivity Environments," *Ad Hoc Networks*, vol. 95, p. 101988, 08 2019.

[121] C. H. Suen, R. K. Ko, Y. S. Tan, P. Jagadpramana, and B. S. Lee, "S2Logger: End-to-End Data Tracking Mechanism for Cloud Data Provenance," in *2013 12th IEEE International Conference on Trust, Security and Privacy in Computing and Communications*, pp. 594–602, July 2013.

[122] S. Sultana, G. Ghinita, E. Bertino, and M. Shehab, "A Lightweight Secure Scheme for Detecting Provenance Forgery and Packet DropAttacks in Wireless Sensor Networks," *IEEE Transactions on Dependable and Secure Computing*, vol. 12, pp. 256–269, May 2015.

[123] X. Liang, S. Shetty, D. Tosh, C. Kamhoua, K. Kwiat, and L. Njilla, "ProvChain: A Blockchain-Based Data Provenance Architecture in Cloud Environment with Enhanced Privacy and Availability," in *2017 17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID)*, (Los Alamitos, CA, USA), pp. 468–477, IEEE Computer Society, may 2017.

[124] "Who announces covid-19 outbreak a pandemic." [Online]. Available: http://www.euro.who.int/en/health-topics/health-emergencies/coronavirus-covid-19/news/news/2020/3/who-announces-covid-19-outbreak-a-pandemic.

[125] G. Kampf, D. Todt, S. Pfaender, and E. Steinmann, "Persistence of coronaviruses on inanimate surfaces and its inactivation with biocidal agents," pp. 246–251, 01 2020.

[126] "Cellphone tracking could help stem the spread of coronavirus. is privacy the price?." [Online]. Available: https://www.sciencemag.org/news/2020/03/cellphone-tracking-could-help-stem-spread-coronavirus-privacy-price.

[127] "Private kit: Safe paths." [Online]. Available: http://safepaths.mit.edu/.

[128] "Halting covid-19: The benefits and risks of digital contact tracing." [Online]. Available: https://spectrum.ieee.org/the-human-os/biomedical/ethics/halting-covid19-benefits-risks-digital-contact-tracing.

[129] David Chadwick and Dave Longley and Manu Sporny and Oliver Terbu and Dmitri Zagidulin and Brent Zundel , "Verifiable Credentials Implementation Guidelines 1.0: Implementation guidance for Verifiable Credentials," tech. rep., World Wide Web Consortium (W3C), Nov 2019.

[130] S. Gusmeroli, S. Piccione, and D. Rotondi, "Iot access control issues: A capability based approach," in *2012 Sixth International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing*, pp. 787–792, July 2012.

[131] "The Falsified Medicines Directive," tech. rep., European Union (EU), Jun 2011. [Online]. Available: `https://ec.europa.eu/health/sites/health/files/files/eudralex/vol-1/dir_2011_62/dir_2011_62_en.pdf`.