Universitá degli Studi di Messina

Dottorato di Ricerca in Ingegneria Civile Ambientale e della Sicurezza

Curriculum Scienze e tecnologie, materiali, energia e sistemi complessi

per il calcolo distribuito e le reti SSD ING/INF05

Ciclo XXXV

Blockchain for Smart Cities: systems, infrastructures and citizens

AUTHOR:

ARMANDO RUGGERI

HEAD OF THE DOCTORAL SCHOOL:

Prof. Dr. Gaetano Bosurgi

ADVISOR:

Prof. Dr. Massimo Villari

Accademic Year 2021-2022

Abstract

The present PhD thesis investigates the application of new technological achievement in many social aspects of a modern city evolving to become a Smart City.

One fundamental aspect of digital communication is to ensure a secure and private channel between people or devices. Existing solutions have severe limitations in terms of security and resistance to distributed attacks, and this thesis proposes new approaches based on Blockchain technology to ensure trustiness in communications.

In particular, the healthcare sector has been investigated to identify some of the situations that can be prevented to improve patient's health, considering the clinical condition that could have been prevented by knowing the risk of concomitant therapies thanks to the use of different technologies applied in healthcare. In fact, it is possible to provide feedback on the prescription of drugs at a patient with specific diseases through a Big Data analysis algorithm able to check drugs and diseases relationships and detect possible failures in drugs prescriptions. In general, this PhD thesis collects several research works with the main goal of covering many aspects that can be beneficial for Citizen's quality of life, both in terms of innovation and security.

Keywords: Blockchain, Security, IoT, Smart Cities, e-health

Contents

List of Figures

List of Tables

Earlier Publications

This thesis is the outcome of the doctoral degree started three years ago. It is based on selected works (listed here below) already published, or under review, in scientific conferences proceedings and journals. Parts of these papers are contained in verbatim.

[1] Armando Ruggeri, Maria Fazio, Antonio Celesti, and Massimo Villari. Blockchain-based healthcare workflows in federated hospital clouds. In European Conference on Service-Oriented and Cloud Computing, pages 113-121. Springer, 2020

[2] Antonio Celesti, Armando Ruggeri, Maria Fazio, Antonino Galletta, Massimo Villari, and Agata Romano. Blockchain-based healthcare workflow for tele-medical laboratory in federated hospital iot clouds. In Blockchain in the Internet of Things: Opportunities, Challenges and Solutions. MDPI, 2020

[3] Armando Ruggeri, Maria Fazio, Antonino Galletta, Antonio Celesti, and Massimo Villari. A decision support system for therapy prescription in a hospital centre. In 2020 IEEE Symposium on Computers and Communications (ISCC), pages 1-4, 2020

[4] Antonino Galletta, Armando Ruggeri, Maria Fazio, Gianluca Dini, and Massimo Villari. Mesmart-pro: Advanced processing at the edge for smart urban monitoring and reconfigurable services. Journal of Sensor and Actuator Networks, 9(4), 2020

[5] Armando Ruggeri, Antonio Celesti, Maria Fazio, Antonino Galletta, and Massimo Villari. Bcb-x3dh: a blockchain based improved version of the extended triple diffie-hellman protocol. In 2020 Second IEEE International Conference on Trust, Privacy and Security in Intelligent Systems and Applications (TPS-ISA), pages 73-78, 2020

[6] Armando Ruggeri, Antonino Galletta, Antonio Celesti, Maria Fazio, and Massimo Villari. An innovative blockchain based application of the extended triple diffie-hellman protocol for iot. In 2021 8th International Conference on Future Internet of Things and Cloud (FiCloud), pages 278-284, 2021

[7] Armando Ruggeri, Antonio Celesti, Maria Fazio, and Massimo Villari. An innovative blockchain-based orchestrator for osmotic computing. Journal of Grid Computing, 20(1):1-17, 2022

[8] Alessio Catalfamo, Armando Ruggeri, Antonio Celesti, Maria Fazio, and Massimo Villari. A microservices and blockchain based one time password (mbb-otp) protocol for security-enhanced authentication. In 2021 IEEE Symposium on Computers and Communications (ISCC), pages 1-6. IEEE, 2021

[9] Armando Ruggeri, Rosa Di Salvo, Maria Fazio, Antonio Celesti, and Massimo Villari. Blockchain-based strategy to avoid fake ai in ehealth scenarios with reinforcement learning. In 2021 IEEE Symposium on Computers and Communications (ISCC), pages 1-7, 2021

[10] Lorenzo Carnevale, Armando Ruggeri, Francesco Martella, Antonio Celesti, Maria Fazio and Massimo Villari. Multi Hop Reconfiguration of End-Devices in Heterogeneous Edge-IoT Mesh Networks. In 2021 IEEE Symposium on Computers and Communications (ISCC), 2021, pp. 1-6.

[11] Armando Ruggeri and Massimo Villari. Improving the Key Exchange Process of the eXtended Triple Diffie-Hellman Protocol with Blockchain. In European Conference on Service-Oriented and Cloud Computing, PhD Symposium, 2022 (Accepted).

[12] Armando Ruggeri, Antonino Galletta, Lorenzo Carnevale, and Massimo Villari. An Energy Efficiency Analysis of the Blockchain-Based eXtended Triple Diffie-Hellman Protocol for IoT. In 2022 IEEE Symposium on Computers and Communications (ISCC). (Accepted)

CHAPTER 1

Introduction

With the latest demand of devices connected to Internet and the increasing need of on-demand services and real-time tracking, security is one of the biggest concerns for the diffusion of online services. If in the past security was concerning the access to the data, moving from different levels of password security algorithms to more advanced data biometric access, the new important need is to certify data exchanged, making it not mutable and access-controlled. Metropolitan Cities are evolving to accommodate the need of a connected world and it is becoming crucial to guarantee that data is authentic, reliable and certificated.

Since the advent of Bitcoin in 2009 and Smart Contracts in 2015, many sectors of public and private interest have studied for Blockchain applications, and General Data Protection Regulation (GDPR) helped to improve researches on user's privacy data protection.

In the following chapters of this thesis, the basic concepts and characteristics of innovative services aimed at securing communication guaranteeing data reliability will be discussed. The research questions investigated are grouped in three main topics and are specified as follows:

From Metropolitan City to Smart City:

- **Question 1:** How can a Metropolitan City evolve to become an innovative Smart City using the new technological achievement?
- **Question 2:** How can a device change its behavior autonomously accordingly to Smart City needs?

Increase security in public connections through the Blockchain:

- **Question 3:** How can citizens of a Smart City securely access to Municipality services?
- **Question 4:** How can online communication be securely and efficiently achieved, without an upfront investment, making it resistant to distributed cyber-attacks?

Improving the healthcare sector with the Blockchain:

- **Question 5:** How can the healthcare sector be improved, with a global and shared cooperation between Medical Doctors and new technologies to protect the sharing of patient's data?
- **Question 6:** How can Doctors and Patients be incentivized to scrupulously follow procedures and treatments?

1.1 Scientific contributions

In this Section the research questions previously described are answered.

From Metropolitan City to Smart City

Question 1: How can a Metropolitan City evolve to become an innovative Smart City using the new technological achievement?

Contribution 1: A new paradigm for municipal collaboration between the government and businesses has emerged with the adoption of Information and Communication Technology (ICT) applications for the creation of innovative and sustainable Smart Cities. By encouraging and enabling businesses to invest their resources and expertise in the cities and by enhancing the prosperity and contentment of their residents, Smart Cities contribute to social stability and economic growth. It is crucial to investigate the main aspects that can improve Citizens' quality of life, starting from but not limited to Hospitals and the healthcare sector in general. Several studies are reported in the following Chapters of this Thesis with the objective to improve efficiency by reducing errors and establishing strict protocols ensuring accountability. Nowadays this is possible with the recent technological achievements and a concrete application is close.

Question 2: How can a device change its behavior autonomously accordingly to Smart City needs?

Contribution 2: A new approach based on Blockchain has been studied and tested, with the aim to leverage the flexibility and robustness of Smart Contracts.

Specifically, it is proposed the BlockChain-Based Function-as-a-Service (BCB-FaaS) framework to allow combining the well-known FaaS paradigm with the non-repudiabilty features of the Blockchain.

Details related to this contribution and experiments are discussed in Sections 3.1, 3.2, and 3.3. This research has originally been presented in [4, 7, 10].

Increase security in public connections through the Blockchain

Question 3: How can citizens of a Smart City securely access to Municipality services?

Contribution 3: A Microservices and Blockchain based One Time Password (MBB-OTP) protocol for security enhanced authentication has been designed and prototyped, supporting a decentralized two-factor authentication (2FA) system that generates OTPs.

The main purpose is to approach the generation and distribution of OTPs in a decentralized fashion to reduce the dependence from any Single Point of Failure (SPoF) and make the Digital Identity management system more robust against possible attacks. Blockchain is a consolidated solution for ensuring the integrity of data in a distributed network, and it can be successfully adopted to replace the traditional central trusted authority, that is responsible for the OTP distribution, with a cooperative trusted environment that implements the same functionalities. With the MBB-OTP the risk of distributed cyber-attacks is reduced: by splitting the 2FA service into different and independent microservices, in case of violation of one of them, the attacker would get totally decontextualized information without affecting the end-user security, and it mitigates DoS attacks using the microservices architecture principles: a compromised microservice can be easily replaced by a non-corrupted one.

Details related to this contribution and experiments are discussed in Section 4.1. This research has originally been presented in [8].

Question 4: How can online communication be securely and efficiently achieved, without an upfront investment, making it resistant to distributed cyber-attacks?

Contribution 4: The BlockChain-Based Extended Triple Diffie-Hellman (BCB-X3DH) protocol has been proposed, allowing to combine the X3DH protocol with the intrinsic features of data non-repudiation and immutability that are typical of Smart Contracts. In this protocol, the centralized server used to perform agreements among parties is replaced by a distributed Blockchain network. The objective is to integrate a Blockchain based technology, leveraging the intrinsic features of data non-repudiation and immutability of Smart Contracts, combined with the well-known X3DH security mechanisms.

In particular, it is considered a Smart City scenario, composed of several IoT constrained devices such as sensing devices battery-powered, rechargeable through solar panel, testing the BCB-X3DH protocol applicability for low-resources devices.

Details related to this contribution and experiments are discussed in Sections 4.2 and 4.3. This research has originally been presented in [5, 13].

Improving the healthcare sector with the Blockchain

Question 5: How can the healthcare sector be improved with new technologies to protect the sharing of patient's data?

Contribution 5: A modern healthcare workflow has been designed to support Medical Doctors around the world with a global and shared cooperation, supported by automated decision support algorithms.

The goal is to harmonize health procedures with new technologies to guarantee patients' safety, verifying that the prescribed therapy does not conflict with other diseases or drugs used by the patient and, in the history of the hospital's clinical records, the same therapy did not lead to death. When physicians visit a patient they can add a prescription treatment indicating the disease to cure in the form of ICD9-ICD10, a medicine identified by ATC and a description with dosage and mode of usage.

Details related to this contribution and experiments are discussed in Sections 5.1, 5.2 and 5.3. This research has originally been presented in [1, 2, 3].

Question 6: How can Doctors and Patients be incentivized to scrupulously follow procedures and treatments?

Contribution 6: Focusing on Reinforcement Learning via Markov Decision Process formulation, it has been considered an agent acting in his environment motivated by the achievement

of the maximum individual objective by appropriate incentives. To achieve this goal, a group of selected Medical Doctors defines a transition matrix with a set of rules, scores and rewards for the treatment of a selected group of patients for the trials.

The environment is defined as space where the agent can take actions, get rewards, and learn, to reach the optimal stage which guarantees the achievement of maximum rewards over time. Reinforcement Learning is the area of AI concerned with how intelligent agents take the most profitable actions based on training the agents themselves to account for observations and rewards from the environment until the task goal is fulfilled.

Details related to this contribution and experiments are discussed in Section 5.4. This research has originally been presented in [9].

1.2 Structure of the Thesis

In Chapter 2, the basic technologies used on the rest of the thesis are analyzed.
Chapter 3 demonstrates how IoT devices are required to achieve a real Smart City scenario. In particular Section 3.1 proposes a comparison of a traffic monitoring camera and an IoT device equipped with a non-expensive camera to monitor the traffic in a metropolitan city during peak hours, to underlay the efficiency in terms of cost and flexibility that can be obtained with an open-source device, and Section 3.2 highlights new improvements to reconfigure end-devices in heterogeneous Edge-IoT mesh networks. Finally, Section 3.3 proposes a new approach for service orchestration based on Blockchain, able to modify the behavior of IoT devices and microservices on demand with a secured and anti-tampering approach.

Chapter 4 discusses the details of the research for new strategies to increase security in public connections through the use of Blockchain technology. In particular, Section 4.1 explores the potential of Blockchain to guarantee a secure One Time Password (OTP) generation with a seed stored in the network. Section 4.2 proposes an improved version of the Extended Triple Diffie-Hellman Protocol based on Blockchain, and Section 4.3 extends it for low resource and general IoT devices that are usually limited in spendable resources to generate secret keys. Finally, Section 4.4 validates the protocol in terms of energy consumed for small devices using an energy sensor [14].

Chapter 5 describes in details the contributions of the most recent research activities in the healthcare scenario. In particular, Section 5.1 discusses the use of a decision support system to improve prediction of clinical analyses. Section 5.2 presents the first approach of an innovative healthcare workflow supported by Blockchain technology, and Section 5.3

integrates it with a data certification system. Furthermore, to mitigate the risk of fake data in machine learning training models, Markov Decision Processes are utilized in Section 5.4.

Each Section presents the same structure, as it follows:

1. introduction to the problem;
2. review of the state of the art;
3. discussion about background and specific technologies;
4. design and implementation of the system;
5. performance evaluation;
6. remarks for the specific contribution.

Finally, Chapter 6 concludes the thesis providing lights on the future.

CHAPTER 2

Background

This chapter provides background information about technologies used as basis of this thesis, with particular attention to the Blockchain and why it is radically changing all industries.

Internet of Things

The term IoT (Internet of Things), coined in 1999 by Kevin Aston, identifies all those electronic devices equipped with a computing capacity, capable of acquiring and processing data independently. In recent years IoT devices played an integral part of smart environments for an increasingly connected world.

In general, IoT devices are used in different applications and heterogeneous environments, for examples at home with voice assistants, household appliances and devices connected to their local network by means of control panels for monitoring heating systems and much more.

Within the scenario of Smart Cities and Industry 4.0, the IoT is a fundamental element for improving the quality of life of users and supporting company policies and maintenance actions in production environments.

There are no limits where IoT is used: medical devices, automotive safety, agriculture and building construction are just some of the areas that are growing thanks to the innovative applications of these small computers.

During the design phase of an IoT-based smart system, it is of fundamental importance to

appropriately choose the type of hardware, based on costs, computational resources and the functions that the IoT devices themselves must be able to perform. In general, IoT devices are based on microcontroller and microprocessor technology:

- **Microcontrollers:** are devices characterized by a very simple architecture and allow a control of the digital and analog signals acquired by the sensors connected or integrated to them. A microcontroller device is not particularly performing as it has a low computational and memory capacity, and for these reasons they are used in real-time applications, which do not require the storage of acquired data.
- **Microprocessors:** are integrated boards with a high amount of memory and computational resources and they are designed for data processing and instruction execution. By means of a microprocessor it is possible to run specifically designed operating systems and create applications that can be used both in heterogeneous environments with an internet connection or a local network.

IoT systems security

The presence of IoT systems in heterogeneous environments, connected directly to the Internet, may lead to vulnerabilities exploits and possible cyber-attacks. In recent years there has been a substantial increase in the number of IoT devices in sectors such as Industry 4.0, smart cities, healthcare, smart homes and many others, and the need to ensure security protocols has increased.

With respect to conventional computing systems, IoT devices are subject to numerous attacks and risks for several reasons:

1. IoT systems do not have a well-defined perimeter and are constantly changing due to the mobility of devices and users;
2. IoT systems are highly heterogeneous with respect to means of communication and protocols, platforms and devices;
3. IoT devices could be autonomous entities that control other IoT devices;
4. IoT systems may include "things" not designed to be connected to the Internet;
5. IoT systems, or parts of them, may be physically unprotected and / or controlled by different parties;

6. Unlike smartphone applications, which require installation authorization and many user interactions, granular authorization requests may not be possible in IoT systems due to the large number of devices.

The communication of IoT devices within a network must take place by means of cryptographic tools useful for encrypting the data to be transmitted, in order to protect them from unauthorized third-party entities. A clear example is that of low-cost home automation systems in smart homes, where the negligible costs are due to a failure to secure communications between the devices themselves. To solve these problems it is necessary to use certified and validated IoT devices within the network in which they operate.

In general, three levels of security are identified:

- **Local Security** is the most widely used security system locally and consists in the recognition of identities by means of passwords. Changing the default passwords is highly recommended. One solution for vendors is to provide devices with randomly generated passwords, and disable services that are not used, such as SSH, X Server and Samba. Within this level it is recommended to avoid over-the-air device administration.

- **Network Security** is the system that uses the communication protocols considered safe from a theoretical point of view, compared to self-writing communication protocols. The protocols HTTPS, MQTT / SSL and XMPP are considered safe for Internet communication of IoT devices.

- **Software Security** is a primary aspect when developing applications for IoT devices. Software deployed over remote IoT can be certified.

2.1 Digital certificates

A digital certificate is a structured file, presented as an electronic document, its main purpose is to guarantee the identity of a subject within the network. Through certificates, it is possible to transmit encrypted data over the network that can only be read by those who have a specific certificate and the permissions to read the information contained therein, ensuring that the data comes from a reliable sender.

In network security and public key cryptography, the digital certificate is used to associate additional information to a public key that reinforces the identity and identification of the subject, whether it is a physical entity, a company, an organization, an application, a website or communication networks, or, more generally, a computer.

A digital certificate is composed of:

- the public key information;
- information on the identity of the owner;
- the digital signature of an entity that verified the contents.

The certificate can not be tampered or duplicated, and it can be verified online. To obtain a reliable certificate, users must rely on third-party authority. This entity is called the Certification Authority (CA). The CA uses the digital signature to protect the authenticity of a certificate.

The Certification Authority is a third party subject, considered reliable for the recognition of subjects in possession of a digital certificate.

Before being able to issue a certificate, the CA must generate and sign a certificate for itself that certifies its identity, within the organization in which it operates. Each issued certificate is signed by the CA, through its own private key.

The CA is one of the active and fundamental entities within a public key infrastructure (PKI).

Public Key Infrastructure (PKI)

The Public Key Infrastructure refers to those resources and entities required to operate with digital certificates and manage data encryption with a public key. This infrastructure is composed of entities, each with its own role, which follow certain rules and protocols. These procedures take place by means of hardware and software resources that allow to perform the following operations on certificates: creation, management, distribution, usage, storage or conservation, and revoke.

The goal of the PKI is to secure the transmission of data over the network, as they contain sensitive information that must not be read by those without authorizations. In general, the authentication system based on a keyword (password) is not sufficient to guarantee the identity of the reader, and in many cases, it becomes risky to communicate personal or confidential data to subjects whose identity is not guaranteed. The PKI is required to identify the parties involved in a communication and to verify that such information is transferred securely. The PKI, therefore, lays the foundations for providing "trust services", in which all actions can be considered reliable as they are performed by recognized entities.

A trusted service respects the three main properties of Internet Security, known as the CIA triad:

1. **Confidentiality:** the data is encrypted to make it secret and to be shared on the network so that the content of the payload is unreadable. The use of this property is evident in Transport Layer Security (TLS).

2. **Integrity:** through hashing functions, it is guaranteed that the data was not alternated at the time of delivery to the recipient.

3. **Authenticity:** it is verified that the sender is a verified entity and the owner of the data transferred on the network.

2.2 Blockchain

Blockchain has been discussed for the first time in 1991, when two American researchers were looking for a way to approve and maintain the integrity of a digital document, without the possibility of it being backdated or modified. In the following years there have been many attempts to insert a digital currency within the network, however these systems suffered from the so-called double spendig, or the possibility of spending the exact same coin twice, similar to distributing a copy of a file. In 2009 a solution was found and the Blockchain was used to create the first digital cryptocurrency: the Bitcoin [15].

The Blockchain is a distributed, decentralized digital archive shared with anyone who wants to be part of the network to read or add data stored on it. Given the decentralized nature, the resources stored in this repository are difficult to destroy, as the data should be eliminated from every computer that is part of the network.

From a practical point of view, each block of the chain is composed of the following information:

1. Data: the content depends on the nature for which the Blockchain was used. In the case of Bitcoin, information about the sender, recipient and amount of Bitcoin transferred is stored.

2. Hash: is a sequence of characters that identifies the block, and is closely linked to the content of the block, that is the stored data, so it can be considered as its fingerprint. If the data changes, the hash will change too.

3. Hash of the previous block: the presence of this additional hash guarantees the chain structure, thanks to the hash's characteristic of being unique based on the content of the block. If the data of a previous block is changed, then the hash of the subsequent blocks must in turn be modified. This ensures the validity of the block sequence. The first block is called "Genesis Block" and it's the only block without the hash of the parent block.

The structure of a block can vary between different blockchains, but few essential attributes are always present:

- Block header:
  - Pointer to the previous block
  - Nonce
  - Timestamp
  - Merkle Root
- Block body:
  - Transaction 1
  - Transaction 2
  - Transaction N

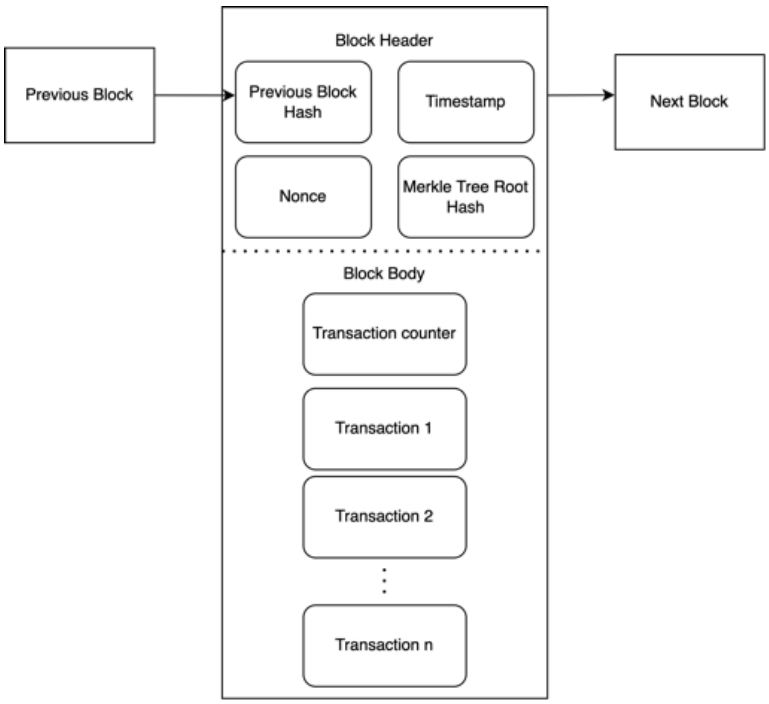


Figure 2.1: Example block structure

Each block receives a unique 64 alphanumeric ID, known as transaction hash, calculated with the algorithm SHA-256.

As it can be observed from Figure 2.1, the block header contains the hash of the previous block, which connects all the blocks together, mathematically chaining them. The hash is a fingerprint, and it is used to maintain all transactions in sequence.

Hash functions

Hash functions are not a new concept, and they were originally proposed in 1950s. This old innovation creates unidirectional functions that cannot be deciphered: through a mathematical algorithm, the hash function takes as input a string of any size and returns a string of bits of fixed size. The bit string lenght depends on the algorithm used, and it represents the encrypted data.

The Secure Hash Algorithm (SHA) is one of the hash functions used in Blockchains. SHA-256 is a common algorithm that generates a unique, fixed-size 256-bit (32-byte) hash. A practical example can be seen in Figure 2.2.

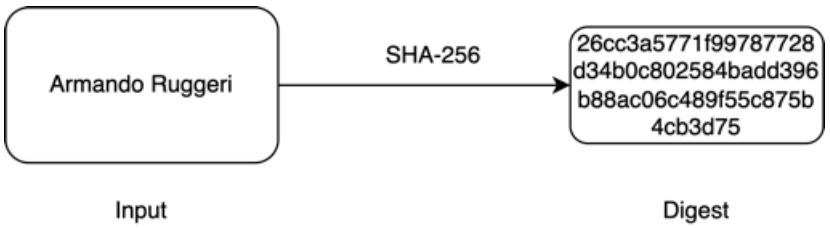


Figure 2.2: SHA-256 function example

A cryptographic hash function is a function

$$h : \{0,1\}^* \rightarrow \{0,1\}^n$$

that given a string m of length * any as input, associates in output a message digest $h(m)$ of fixed length.

These functions are particularly useful in cryptography for the following properties:

- Given any $m, h(m)$ must be calculable effectively.
- Preimage resistant: resistance to counter-images, given any y it must be computationally impossible to find a n such that: $h(m) = y$.

- Strong resistance to collisions: it is impossible to find $m1, m2$ such that $h(m1) = h(m2)$ with $m1 \neq m2$

To guarantee that the information transmitted trough an unsafe network is intact and unaltered, due to physical layer or intentional attack or tamper, the cryptographic hash functions help to solve this problem.

One of the main uses of hash functions is therefore to obscure some data in order to limit the use that can be made of it. For example databases do not store user passwords but a password hash. In this way the server can verify that a user enter the correct password by checking the password hash in a manner fast, but anyone trying to access the password database displays only hashes, useless for the purpose of gaining credentials.

The hash functions cryptographic systems are also used by the Bitcoin system to check that transactions are performed correctly, since they are designed so that a minimal difference in input radically changes the result.

When a transaction occurs the hash of the previous transaction is calculated and stored as part of the message itself. This guarantees that a change in the transaction is practically impossible, since the hashes would no longer match.

Another fundamental use of the cryptographic hash functions in the Bitcoin system is the generation of new coins. In fact, bitcoins are generated continuously and are given in ownership to those nodes who can solve the mathematical problem of the calculation of a counter-image, described above.

The term SHA, Secure Hash Algorithm, means one family of cryptographic hash functions developed by the National Security Agency since 1993.

In the Bitcoin system, two hash cryptographic functions are used for address creation: SHA-256 and RIPEMD-160. The number on the name represents the bit length of the function output (RIPEMD-160 is a cryptographic hashing algorithm that produces a hash output of 160 bits, thus being shorter than a hash of 256 bit). The RIPEMD algorithms have been developed in contrast to those devised by the National Security Agency, however Bitcoin uses them both to make the system more stable and secure.

In Bitcoin normally hash calculations are performed in two phases: the first with SHA-256, and the second with SHA-256 or RIPEMD-160, depending on the desired length.

Merkle Tree

A Merkle Tree or hash tree is a binary tree, whose ramifications are two at each step, in which each node is labeled with the hash of its child nodes. Merkle proposed in his PhD thesis in 1979 the idea of a hash tree that would allow the efficient and secure verification of the contents contained therein. The tree structure allows the verification of a block through a quantity of data proportional to the logarithm of the number of tree nodes. As it can be seen in the figure 2.3, the hashes of all the blocks, taken from one list, are paired each time until remains only one hash, called top hash.

Hash 0 is the hash of resulting from Hash 0-0 and Hash 0-1 concatenation. Precisely, $Hash0 = hash(Hash0 - 0 + Hash0 - 1)$.

To verify the integrity of a single block it is therefore necessary to consider a tree subset. For example, to check the integrity of block 1, the nodes Hash 0-0, Hash 0-1, Hash 0, Hash 1 and Top Hash must be considered Hash, without considering nodes Hash 1-0 and Hash 1-1. In the particular Bitcoin system each leaf of the tree consists of a double SHA-256 hash function, i.e. the SHA-256 hash of the current block which contains inside the SHA-256 hash of the previous block.

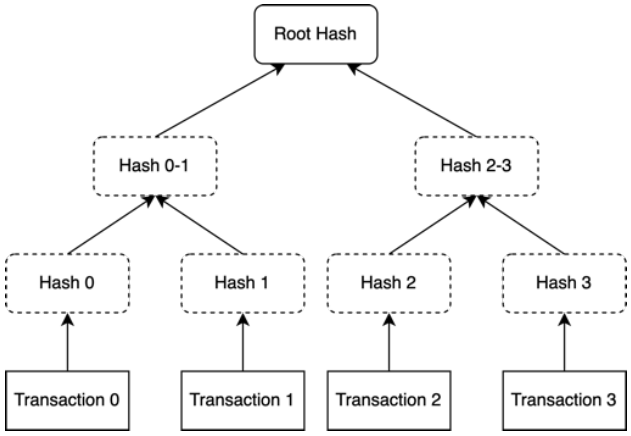


Figure 2.3: Simple structure of Merkle Tree

It is important to consider that one of the main advantages of the Merkle trees is that to calculate the hash of a given block it is not necessary to calculate the hash of the entire tree, but only the nodes along the branch, up to the root node. In this way the number of the hash calculations required decreases logarithmically on the number of blocks of total data.

This type of data structure is very resistant to tampering and also guarantees its integrity in the order of the transactions, since, if one were to change the order of a transaction, it would change everything. resulting hash.

The Merkle root in a Blockchain is present within the block header and is the hash of all transactions contained within the block itself. This means that instead of verifying all transactions one by one, with Merkle root verification alone, it is possible to verify all existing transactions.

A node has to check if a certain transaction has taken place. At this point, two things need to be verified: the transaction as part of the block and the block as part of the blockchain. To do this, a node does not need to download all the transactions of a block, but it can simply ask the network for information about the hash of the block and the hash of the transaction. The peers on the network that have the information they need can respond with the Merkle root of that transaction and the respective block. This might raise some questions about the reliability of the data provided, but we know that hash functions are one-way. Therefore in no way can an adversary node falsify transactions that match a given hash value.

The use of Merkle trees is not limited to blockchains alone: they are widely used in many other applications such as BitTorrent and NoSQL database such as Cassandra.

An interesting role that they play within the blockchain are the nodes participating in the network. They can be both miners, who have the task of creating new blocks and therefore new cryptocurrency, both signers of the blocks, who validate and digitally sign transactions. By using consensus algorithms, however, the blockchain technology will select one of the nodes participating in the network; the latter will add the block to the blockchain.

We have heard of blockchain for the first time, with the arrival of the cryptocurrency Bitcoin, which is exploited only for financial transactions between users of that same network. Bitcoin was therefore the first successful implementation of cryptocurrencies. It is therefore classifiable as the first generation of blockchain technology, blockchain 1.0. All alternative cryptocurrencies, such as Bitcoin, fall into this category.

It is also possible to recognize a second generation used for financial services and smart contracts. In this release, the Ethereum and Hyperledger blockchains can be included, which are two examples of blockchain platforms considered part of the blockchain 2.0.

Consensus algorithm

In order for a Public Blockchain to guarantee the properties of immutability, privacy, security and transparency, without the presence of a central authority that verifies the transactions, a protocol is required that performs the work of validator within the network.

This protocol is a consensus mechanism through which the nodes belonging to the Blockchain network are able to agree on the current state of the distributed ledger and to

approve or reject it. Through it, it is possible to ensure that the new block added to the chain is the only one whose version of the block's content reflects reality, based mainly on the number of nodes that consider it as such.

The approval, collaboration and cooperation activities between the nodes depends on the consensus algorithm. The most known algorithms are presented below:

Proof of Work (PoW): The problem of Proof of Work is a cryptographic problem that requires a huge number of attempts and can only be solved by the method of Brute Force, which makes it impossible to predict which user will find the solution before the others.

A Proof Of Work is a complex task to calculate since it must meet certain requirements and is expensive from the point of view of time and resource calculation required. It is also calculated by a random search process that has a low probability of success.

The Bitcoin system uses a version of the Hashcash Proof of Work, proposed by Adam Back in 1997, and exploits the apparent randomness of functions cryptographic hash. An algorithm converts an arbitrary data into a number hexadecimal and, if the initial datum is increased and proposed to the calculation of the hash, a completely different result is produced.

This operation is performed until a result less than T value set by the network is found. The smaller is the value, the more difficult and expensive is the operation of generating a new block.

The solution to the Proof Of Work problem is found by finding:

$$x \quad \text{such that} \quad H(x) \leq T$$

where H represents a cryptographic hash function.

The protocol followed by the miners to solve the problem is the following:

1. Increase each time by 1 a random number, present in the block header, which is called Nonce
2. Perform a SHA256 hash function of some parts of the block (called block header
3. Verify that the calculated hash $H \leq T$, than:

   If the value is less, the user has solved the problem and its block is added to the chain

   If not, the block is rejected by the system and starts again from point 1.

The difficulty of the Proof Of Work is adjusted to limit the speed with which new blocks can be generated by the network, and then added to the Blockchain: if the blocks were created too quickly, the Blockchain would contain many bifurcations, which would make it difficult for the nodes to choose the branch on which to work.

On the other hand, if the creation were too slow, it would take a lot time to confirm transactions. The nodes of the Bitcoin system use the target T to automatically adjust the speed of block creation, so that it takes an average of 10 minutes each time for a new block to be created.

The target T is a 256-bit number shared by all clients. The maximum target used by Bitcoin is 0x00000000FFFF0000000000000000000000000000000000000000000000000000, and represents the lowest difficulty of creating a block which can be found.

Difficulty D is a measure of how difficult it is, from computationally prospective, to find a hash value less than a fixed target value, defined as the relationship between the maximum possible target and the current target.

The difficulty is changed every 2016 blocks created (about 2 weeks), and is based on the average time used to create the last block. Since each hash value is a number of 256 bits, there are 2256 possible values hash.

Solving the Proof Of Work problem does not just involve an expense of time, but also of electricity, which allow computers to process the problem and perform calculations continuously. To entice users a using their own computers to validate the blocks, the Bitcoin system bestows a certain amount of BTC to those who produce a valid block.

Initially, the reward was 50 bitcoins per block, but if that reward it would always remain the same, the currency in circulation would increase infinitely over time and there would be continuous inflation. To avoid this problem the system is programmed to generate money according to a geometric series, until the total number of Bitcoins reaches 21 million. Bitcoin halves the reward for every 210,000 mine blocks, or about every 4 years.

Proof of Stake (PoS): Proof-of-stake is a distributed consensus protocol that is based on the principle that each user is required to prove to own a certain amount of cryptocurrency. Before a block can be added to the Blockchain, the creator must be chosen. There are various methods to make the choice:

- Random selection: the next account that can generate the block is chosen by a

random function. The disadvantage of this method lies in the fact that its result is easily understood by the nodes of the network.

- Selection based on seniority: mixes random selection with the concept of "seniority". The account that has a lot of coins that are not spent for a long time is more likely to be chosen to sign the next block. When an amount of coins has been used to sign a block, it starts over with "zero seniority" and must wait 30 days before it can be reconsidered to sign a block, thus allowing other users to participate in the selection.
- Selection based on vote: the creators of the blocks are chosen by a vote made by a number of delegates.

Compared to Proof-of-Work, which bases its operation on energy consumption, and therefore on hardware resources, Proof-of-Stake focuses the security of the entire system on the account holding the coin responsible for creating the block.

Proof of authority (PoA): Proof-of-authority is an algorithm used in the Blockchain that quickly processes transactions through an identity-based mechanism. In the PoA-based Blockchain network, transactions and blocks are validated by verified accounts, known as validators. The validator nodes also have a reputation associated with the account and are called upon to validate the transactions from an authority node. By doing so, the validators are motivated to support the transaction process in order to maintain a high reputation. The identity of the validator is made public in a register that can be consulted by anyone who is part of the network. The PoA allows each validator to approve only non-consecutive blocks, minimizing the risk of creating significant damage to the system. However, it is necessary to manage and secure the authority node so that it is not compromised.

New strategies are proposed as the solution for securing data collected by IoT devices such as images, videos, audio content and biometric data, often stored in central servers. The storage on these computers is not exempt from possible cyber attacks that cause the loss or theft of sensitive data.

With Blockchain technologies and by means of consent algorithms, users belonging to the network can verify the data so that this responsibility does not fall into the hands of a single company. One of the downside to Blockchain technology is the space available to store this data, which requires a considerable amount of memory and a high CPU usage for the

consent algorithm, as this data is publicly exposed. With Fusion Chain, a lightweight and decentralized Blockchain, it is possible to support IoT devices, solving the storage problem using IPFS, and high computing power through the PBFT. In these solutions, data security is guaranteed by asymmetric encryption using PKI.

Block Validation

Nodes verify the entire Blockchain on the basis of the Merkle Tree algorithm and collect new transactions generated but not yet validated. Using cryptographic hash function the output is estimated based on the target value. The first node to complete the mining process (i.e. resolve the block hash algorithm) sends it to the network and it gets added to the chain.

Considering X as fixed hash function of the network and X as pending transactions and n represents the value of nonce. The output hash must start with zeros and must be less than the target value, the problem is to determine the value of the nonce to make this happen. The number of zeros at the beginning of the output is the difficulty to solve the block. The difficulty is due to the fact that the cryptographic function of hash generates random numbers, in fact, the variation of even a single bit in the function input leads to a completely different output with respect to the one calculated without the variation; in this way, it becomes practically impossible to predict the nonce.

When the hash function is calculated and the block is solved, the hash is the reference to the previous block. After the block has been resolved, the network automatically adjusts the nonce value. The entire block validation process is called "mining".

It is also possible to validate many blocks simultaneously, creating a fork in the chain. When this happens, as soon as a block is added in one of the fork, all miners move to the longest chain leaving the other block as orphan.

Mining

Mining is the process that creates new validated blocks and adds them to the Blockchain. The nodes that carry out the mining process are called miners. Each miner performs a series of steps that lead to producing a new block:

1. Collect and decide which information, collected by the peer-to-peer network, include in a new block

2. Verify that all transactions included in the block are valid.

3. Select the most recent block in the longest branch of the block chain and inserts a hash of that block into the new block it is creating.

4. Try to solve the Proof of Work for the new block and at the same time observe which other new blocks are created by other nodes.

   If it finds a solution to the Proof of Work problem, the new block is added to the chain chosen and shared on the network.

   If another node first solves the problem of the Proof of Work, his block is subjected to validity checks. If it is valid, it is added to a local copy of the Blockchain and shared in network, otherwise it is discarded

As miners work in parallel, new blocks can be created simultaneously, with most of their time spent solving the proof of work problem, it happens that there are different versions of the Blockchain at a certain instant. This happens because new blocks are created with regularity every 10 minutes roughly, during this time all nodes look to solve the proof of work.

When a node finds a valid solution it gains a certain number of bitcoins in reward, provided by the protocol, shares the block in the network and the nodes that receive the new block, after verifying it, add it to their chain, starting to work on a new block.

The Internet is a revolutionary technology that has changed the way we see the world. It had such an impact transforming the daily life of the single person to the financial world. In fact, banking systems evolved to the point of offering new services, allowing transactions from everywhere in the world.

Blockchain technology gave birth to a new vision of the world, both in private and financial area, eliminating the need of an intermediary. This innovation concerns many aspects, including data storage and software applications management.

At first sight, Blockchain technology seems just an extension of traditional databases. Although it is invisible to the users, it performs operations that affect most online services, but it requires a often centralized third party management.

Centralized systems, in fact, are administered by central authorities. This feature facilitates the design and is also easier to maintain. However, there are some important limitations, such as instability, vulnerability to attacks, therefore less secure and scalability issues.

Blockchain technology, on the other hand, is changing this concept by fueling a new generation of applications based on decentralized systems. In fact, intermediaries are not

required, reducing the dependence from centralized services. Furthermore, each node has equal role and rights (see Figure 2.4).

Decentralized systems can be complex to implement and maintain, but there are important advantages. Being a decentralized system, there is no central point of failure, making it more stable and tolerant to failures. Furthermore, the data is distributed throughout the system. The system is thus more resistant to attacks.

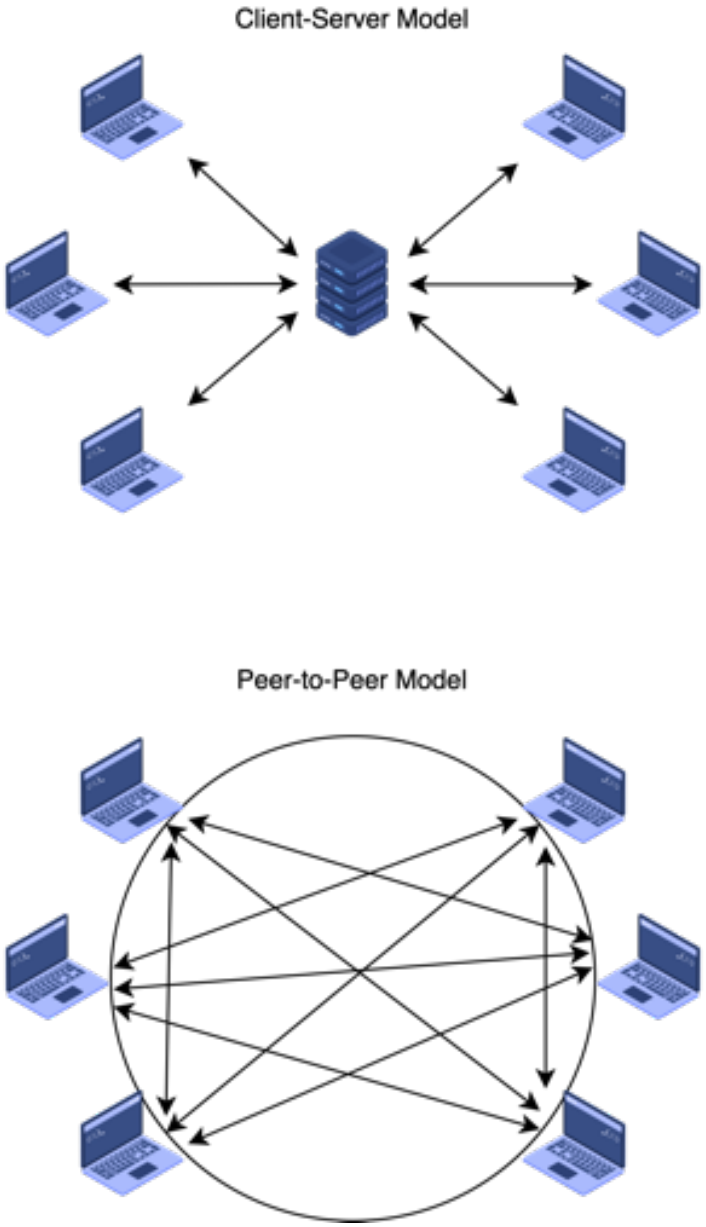


Figure 2.4: Client-Sever Model vs Peer-to-peer Model

The key features of this technology derive from the use of an anonymous, peer-to-peer, append-only network, immutable and updatable only by agreement between peers based on public-private key cryptography.

Cryptography is a fundamental component as it provides the blockchain with the confidentiality of transactions and the maintenance of intact data. This immediately results into:

- data that cannot be tampered with;
- the authenticity of the sender is verifiable by the recipient;
- the sender cannot deny sending.

It is possible to introduce digital signatures, based on public key cryptography. It is composed of two keys, one public and one private, related to each other. In fact, once a data is encrypted with one of the two keys (for example the public key), the data can only be decrypted using the other key, (the private one). (see Figure 2.5). This feature helps to recognize the owner of a certain digital asset.

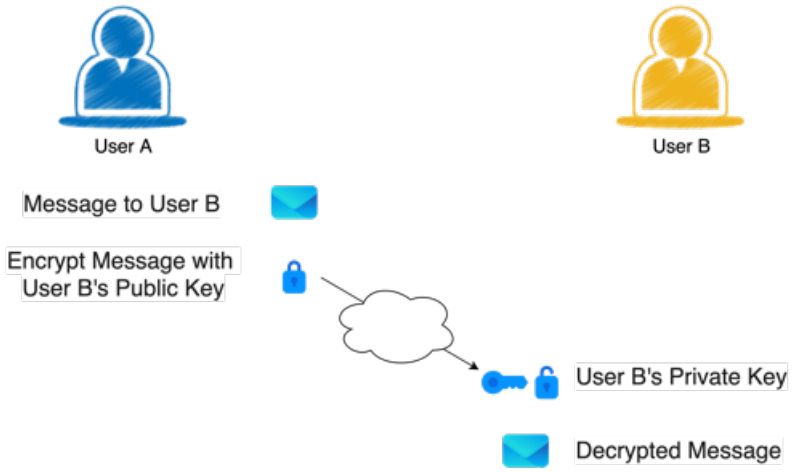


Figure 2.5: Public key cryptography example

Analyzing the definition just provided on blockchain technology, thanks to the a peer-to-peer network any centralized control is passed, since all participants communicates directly in the anonymous network. This concept has been already applied with BitTorrent software, which allows to exchange files directly with other users running the same software. The Blockchain technology implements the concept of distributed ledger, as all the records are distributed among all the peers participating in the network, allowing each peer to keep a copy complete Blockchain data.

One of the main properties of this technology is the append-only mechanism. Data can only be added to the Blockchain at the end of the chain. This means that it is not possible to alter some data already stored in a previous block, which allows the Blockchain to be

considered practically immutable. However, a very rare scenario could arise in which a single entity, or group, participating in the network manages to gain more than 51% of the peers [16]. In such scenario, the entity would have enough computing power to be able to modify or exclude data already present in the Blockchain. At the time of writing, the number of current active nodes in Bitcoin Blockchain are roughly 13,000, which makes this type of attack practically unfeasible.

2.3 Types of Blockchain

There are three main types of Blockchain public, permissioned and private there is no strict distinction, it is possible to combine various aspects of the different types in order to create custom registers for specific applications.

Public or permissionless Blockchain

The public or permissionless Blockchains are defined in this way because no authorization is required to access the network, perform transactions or participate in the verification and creation of a new block.

The most famous are certainly Bitcoin and Ethereum, where there are no restrictions or access conditions. Anyone can take part. This is a completely decentralized structure, as there is no central body that manages access authorizations. These are shared among all nodes in the same way. No user of the network has privileges over others, no one can control the information stored on it, modify it or delete it, and no one can alter the protocol that determines the operation of this technology.

The concepts of permissionless and publicness are closely related to each other. It would be a contradiction to have a private Blockchain where, however, an authorization is not required to access the recorded data; for this reason, all those that do not require approval are defined as public.

Although the data recorded on these Blockchains are public, these are encrypted to maintain a sufficient level of privacy. For example, all Bitcoin nodes know the wallet addresses of other users and the transactions that took place between them. In principle, these addresses are simply pseudonyms and, unless they are traced back to the identity of the person in the real world who owns them, a sufficient level of privacy is guaranteed. One way to further protect one's identity is to use more than a single wallet address.

The main concern related to public Blockchains is the issue of scalability, or the ability

of a system to improve as the number of participants increases. This type of network is not a scalable technology: as the number of nodes increases, the speed of transactions remains unchanged but increases the stability of the system, which becomes safer.

The central property of this type of Blockchain is that the set of nodes, amongst which consensus over the state of the chain should be reached, is known as proof-of-work (POW) Blockchains.

The public Blockchain is based on proof of work (PoW) consensus algorithms which are open source and anyone can participate in to read/write or audit the transaction in the public Blockchain without permission. Being transparent and open to public any individual can download the start running the public node on their local devices, send the transaction, validate it, and track it at any given time on the Blockchain.

The decision-making process is normally done by number decentralized consensus algorithms such as PoW as mentioned above and Proof of Stake (PoS).

Public or permissionless Blockchain possesses the potential to revolutionize the current business models through disintermediation. Bitcoin, Etherum, Dash, Litecoin, etc. are the prime example of cryptocurrencies based on the public Blockchain.

Private or permissioned Blockchain

The permissioned Blockchain is normally developed by the private organization or an individual where not anyone can read/write/audit the transaction.

Generally, write/audit permission is kept centralized while read permission may be public or restricted to the limited people. All of the decision-making processes such as giving mining right are done by the central trusted party. The permissioned Blockchain is way more advanced compared to public Blockchain when it comes to scalability, security and data privacy.

Private Blockchains tend to be smaller and do not utilize a token. Their membership is closely controlled. These types of Blockchains are favored by consortium that have trusted members and trade confidential information. All types of Blockchains use cryptography to allow each participant on any given network to manage the ledger in a secure way without the need for a central authority to enforce the rules. The removal of central authority from database structure is one of the most important and powerful aspects of Blockchains.

The Blockchain permissioned are subject to a central authority that determines who can access it. In addition to defining who is authorized to be part of the network, this authority defines which roles a user can play within it, also defining rules on the visibility

of recorded data. The Blockchain permissioned then introduce the concept of governance and centralization in a network that is born as absolutely decentralized and distributed. Commonly called the Blockchain of the Consortium, instead of allowing any person with an Internet connection to participate in the verification of the transaction process, entrusts the task to some selected nodes deemed worthy of trust.

Private Blockchains share many features with permissioned ones. These are private and non-visible networks that sacrifice decentralization, security and immutability in exchange for storage space, speed of execution and cost reduction. This type of Blockchain is controlled by an organization, considered highly reliable by users, which determines who can access or not the network and read the data recorded in it.

The network's proprietary organization also has the power to modify the Blockchain's operating rules, rejecting certain transactions based on established rules and regulations. The fact that it is necessary to be invited and authorized to access it guarantees a greater level of privacy to the users and determines the secrecy of the information contained.

Private Blockchains can be considered the fastest and the cheapest, since the transactions are verified by a limited number of nodes, thus reducing the timing; therefore the transaction fees are significantly lower than those of the public Blockchains.

The most common combinations:

- **Public permisionless Blockchain** whose most famous examples are Bitcoin and Ethereum. This type of network allows access to every user who decides to connect and participate, generating new transactions, performing the task of miners or simply reading the register of stored transactions. In this system the miners are anonymous, so they are not considered reliable individuals.

- **Public permissioned Blockchain** such as Ripple and Hyperledger Fabric. It is a network that operates on behalf of a community that shares a common interest, where access to the role of miner is limited to a small number of individuals considered trustworthy. The level of reading of the register and participation in the generation of new transactions may be subject to limitations or not depending on the organization that controls the Blockchain.

- **Private permissioned Blockchain** like Chain and Bankchain. In this case, only authorized and authenticated users can access the network since the Blockchain in question operates exclusively within the limits of a well-defined community where all participants are known. Usually, at the head of these systems are financial institutions or

government agencies that define who can access the network or not. This means that all the miners are considered trustworthy.

Hyperledger

A huge contribution to the diffusion of open-source Blockchain comes from the Hyperledger Project supported by the Linux Foundation and several other contributors such as IBM, Intel and SAP. The project consists of different sub projects, each with a different specialization: Burrow, Iroha, Indy, Fabric and Sawtooth.

Hyperledger Sawtooth is the only Hyperledger project which does not support Byzantine Fault Tolerance but relies on the unique Proof of Elapsed Time (PoET) consensus algorithm, which is by far the most efficient technology so far in the blockchain. This guarantees high transaction speed when using Sawtooth-based software with low hardware resources.

For comparison, the Bitcoin's Proof-of-Work consensus protocol uses three-four hundred times the energy and computing power the PoET needs.

Also, Sawtooth relies on the Python programming language, which makes the end software way easier to deploy and service, and it can be combined with Ethereum smart contracts using a tool called Seth.

Out of the five, Fabric is the most talked about (and generally referred to as Hyperledger itself), firstly it was the first project to be accepted by the Linux Foundation, and also thanks to IBM's marketing efforts.

One of its powerful main aspect is the lack of need for a specific programming language or its own virtual machine. Instead its Smart contract logic (here called chaincode) can be written with any programming language, but the majority of those written so far have JavaScript or Go. It is currently supported by tutorials and documentation published by IBM.

Hyperledger Fabric is built on a modular architecture that separates transaction processing into three phases: distributed logic processing and agreement ("chaincode"), transaction ordering, and transaction validation and commitment. This is implemented to obtain a minor number of steps for verification across the nodes, performance optimization and scalability.

Endorser peer In Hyperledger Fabric there are different types of peers: standard peer, endorser peer and anchor peer. Only endorser peer are used when a transaction invocation request occurs, and they are required to validate the transaction and execute the chaincode. At the end of these steps the endorser peer can validate or refuse the transaction. It is important to consider that only the endorser peer executes the chaincode, so it is not required to install

the chaincode in all the peers of the network, thus resulting in better scalability.

Anchor peer is configured at the time of Channel configuration and its duty is to send updates in broadcasts to the other peers in the organization.

Orderer Ordering service provides a shared communication channel to clients and peers, offering a broadcast service for messages containing transactions. The orderer is responsible of maintaining the correct logical order while outputting the messages to all the peers connected.

Hyperledger Fabric Workflow A participant in the member Organization invokes a transaction request through the client application. Client application broadcasts the transaction invocation request to the Endorser peer. Endorser peer checks the Certificate details and others to validate the transaction. Then it executes the chaincode (ie. Smart Contract) and returns the Endorsement responses to the Client. Endorser peer sends transaction approval or rejection as part of the endorsement response. Client now sends the approved transaction to the Orderer peer for this to be properly ordered and be included in a block. Orderer node includes the transaction into a block and forward the block to the Anchor nodes of different member Organizations of the Hyperledger Fabric network. Anchor nodes then broadcast the block to the other peers inside their own organization. These individual peers then update their local ledger with the latest block. Thus all the network gets the ledger synced.

To provide a visual representation of the transaction flow the following figure can be followed starting at the left:

1. The transaction proposal is submitted by an application to an endorsing peer.
2. The Endorsement policies outline how many and/or what combination of endorsers are required to sign a proposal. The endorser executes the chaincode to simulate the proposal in the network peer, creating a read/write set.
3. Then the endorsing peers send back the signed proposal responses (endorsements) to the application.
4. The application submits the transactions and signatures to the ordering service, which
5. Creates a batch, or block, of transactions and delivers them to committing peers.
6. Committing peer receives a batch of transactions and

7. Validates that the endorsement policy was met and checks in the read/write sets to detect conflicting transactions. If both checks pass, the block is committed to the ledger, and the state updates for each transaction are reflected in the state database.

Ethereum

The second Blockchain to have been a great success after Bitcoin is without doubt Ethereum. It was born in 2013 as an alternative to Bitcoin, proposing one substantial difference in the language used.

The novelty of Ethereum lies in the possibility of building decentralized applications thanks to its complete Turing language, whilst in the first version of Bitcoin there was no concept of scripting language.

These applications are called smart contracts. The term "smart" means that the contract in question, intended as one any action is self-executable if certain conditions are met. Code used to write applications belongs to a high level language: Solidity.

To avoid that the programs have infinite loop of execution of the code, the developers have implemented an exhaustion mechanism resources. The resource in question is called *gas* and acts as if it were the fuel of the contract.

Ether

Ethereum users can interact on a peer to peer network using the computational resources of the network called Ether (Ethereum virtual currency). Ether is concretely a token that is treated as cryptocurrency exchanges with the ticker symbol of ETC.

Ethereum Virtual Machine EVM

Ethereum Virtual Machine (EVM) is the Ethereum engine that is represented by a runtime environment for the development and management of Smart Contracts in Ethereum.

The EVM operates in a completely safe way, as it is completely separate from the network. The code managed by the Virtual Machine does not have access to the network and the Smart Contracts themselves are independent each other.

Solidity

Solidity is an object-oriented programming language for writing Smart Contracts, native to Ethereum. It is a static programming language designed for the realization of Smart

Contracts running on EVMs.

Solidity is currently the primary language on Ethereum as well as on other private blockchains running on platforms that compete with Ethereum, such as Monax and its Hyperledger Burrow blockchain, which uses Tendermint for consensus.

The Contract Application Binary Interface (ABI) is the standard way to interact with contracts in the Ethereum ecosystem, both from outside the blockchain and for contract-to-contract interaction. Data is encoded according to its type, and the encoding is not self describing and thus requires a schema in order to decode.

The first four bytes of the call data for a function call specifies the function to be called. It is the first (left, high-order in big-endian) four bytes of the Keccak-256 (SHA-3) hash of the signature of the function. The signature is defined as the canonical expression of the basic prototype without data location specifier, i.e. the function name with the parenthesised list of parameter types. Parameter types are split by a single comma - no spaces are used.

Remix

Remix is a powerful, open source tool that helps writing Solidity contracts straight from the browser. It is written in JavaScript and supports both usage in the browser and locally. Remix also supports testing, debugging and deploying of smart contracts.

Ganache

Ganache, formerly known as Testrpc, is a virtual Blockchain that sets up 10 predefined Ethereum addresses, each with private keys and 100 ETHER simulated. In Ganache there is no "mining" but it confirms any type of transaction that is performed. Through this simulated Blockchain it is possible to write unit tests for the code and execute them on the same, implement smart contracts, call function and simulate new tests by returning the addresses to the initial state of 100 Ether.

Ganache is available in two versions: CLI and UI. Ganache CLI is the latest version of TestRPC, a fast and customizable Blockchain emulator. Through this it is possible to make calls to the Blockchain without the overhead costs of executing a true Ethereum node. The main features of this virtual Blockchain are reported.

1. Transactions executed instantly;

2. No transaction costs.

3. Accounts can be restored and instantiated with a fixed quantity of Ether;
4. Mining activities are not necessary;
5. The gas price and the extraction speed can be changed;
6. A convenient GUI gives you an overview of your testchain events;

Truffle

Truffle provides a system for managing the compilation and deployment artifacts for each network.

Truffle is a development environment, a test and pipeline framework for any Blockchain that uses EVM. Truffle is managed in the Terminal, and as such has a series of useful commands to use in the different phases of development of a DAP. With the truffle it is possible to get:

1. Built-in smart contract compilation, linking, deployment and binary management;
2. Configurable build pipeline with support for custom build processes;
3. Scriptable deployment and migrations framework;
4. Network management for deploying to many public and private networks;
5. Interactive console for direct contract communication;
6. Instant rebuilding of assets during development;
7. External script runner that executes scripts within a Truffle environment;

2.4 Applications of Blockchain

Since this technology was born it has undergone a strong evolution and the applications connected to it have increased, since it is evaluated as one of the most promising technologies in the IT field surely many of the applications concerning the future of Blockchain are yet to be discovered, certainly the application that made it famous was the one linked to the development of cryptocurrencies.

Cryptocurrencies Cryptocurrency is a digital asset used as a medium forexchange of goods or money transactions based on cryptographic algorithms to ensure immutability of the data, control of the access and verification of the transaction. The core of this approach is the decentralization of the control, as opposed to central bank control as for the standard currencies and payments.Bitcoin has been the first example of cryptocurrency, and since 2009 more than 2000 cryptocurrencies (known as Alt-Coin, as an alternative to BitCoin) have been released.

Financial Services The Blockchain was born to offer a valid alternative to the traditional banking system. Decentralizing financial services is theoretically a vast improvement over the centralized banking system, many believe that this technology will change these institutions compared to how we see them before Satoshi Nakamoto created a digital currency that could be used only once, solving the problem of double spending for a long time it represented an obstacle to digital banking, there was no way to avoid centralized banking without storing cash.

Smart Contracts Smart contracts are a developed in different programming languages accordingly to the Blockchain used that are actually stored inside the Blockchain. The example below is to better understand what a smart contract is.

To provide an example to better understand what a smart contract is, consider a car dealer with two main actors: the car seller and the buyer. The seller needs to demonstrate that he owns the car or that he is delegated to sell it; the buyer needs to have sufficient founds in the bank account to buy the car. With a Smart Contract it would be possible to quickly perform these verifications without a third party intermediary, holding the funds until both conditions are validated. As all information are recorded in the Blockchain, these are immutable and trusted.

Immutable means that once a Smart Contract is created, it can never be change again. Distributed means that the output of the Smart Contract is validated by everyone on the network, this means that a single person cannot force the contract because other people on the network will spot this attempt and mark it as invalid.

The following Chapters focus on the application of the above mentioned technologies in modern research contexts.

CHAPTER 3

IoT Applications in Smart Cities

Internet of Things has revolutionized the way services are distributed in smart environments, approaching the computation where data are generated. However, IoT devices have limited resources and reconfiguring them can be very difficult. In these cases, Edge computing represents a challenging solution supporting IoT with flexible management of resources. It has been investigated how pushing computation activities from Edge to IoT, changing the behavior of IoT nodes according to application or system requirements.

3.1 IoT e Smart urban mobility

With reference to the MeSmart project, the Municipality of Messina is making a great investments to deploy several types of cameras and digital devices across the city for carrying out different tasks related to mobility management, such as traffic flow monitoring, number plate recognition, video surveillance etc. To this aim, exploiting specific devices for each task increases infrastructure and management costs, reducing flexibility. On the contrary, using general-purpose devices customized to accomplish multiple tasks at the same time can be a more efficient solution. Another important approach that can improve the efficiency of mobility services is moving computation tasks at the Edge of the managed system instead of in remote centralized serves, so reducing delays in event detection and processing and making the system more scalable. This research investigates the adoption of Edge devices connected to high-resolution cameras to create a general-purpose solution for performing

different tasks. In particular, the Function as a Service (FaaS) paradigm is adopted to easily configure the Edge device and set up new services. The key results of our work is deploying versatile, scalable and adaptable systems able to respond to smart city's needs, even if such needs change during the time. The proposed solution has been tested setting up a vehicle counting solution based on OpenCV, and automatically deploying necessary functions into an Edge device. From experimental results, it results that computing performance at the Edge overtakes the performance of a device specifically designed for vehicle counting under certain conditions and thanks to our reconfigurable functions.

Smart Cities represent a new way to live the urban environment, involving social and technological aspects. In this context, systems based on IoT (Internet of Things) play an important role to allows citizens to interact with the environment and to benefit of advanced services, such as video surveillance, intelligent traffic lightning, air quality sensing and noise estimation, to obtain intelligent means of transport and autonomous vehicles management [17]. From a technological point of view, using sensors and actuators to automatize services becomes really strategic, but managing, configuring and optimizing the digital infrastructures to adapt their behaviour to the specific needs of the context is a big challenge. The Municipality of Messina is hardly working to make Massina a smart city, in order to improve the quality of life of its citizens and the quality of experience in accessing public services.

Messina is located in Sicily, an Italian island in the Mediterranean area of Europe. It can be considered as the "door" of Sicily because only a small portion of sea about 3 km large (Strait of Messina) separates Messina (and the rest of Sicily) from the Italian peninsula. The port of Messina in a strategic position in the center of the city and act as a multimodal hub for the metropolitan/regional network for handling freight, transport passengers from and to the rest of Italy. According to the most recent statistics, in 2019 more than 10 millions of passengers travelled through the Strait of Messina[1]. Messina is also tourist destination with about 400.000 cruise passengers in its port per year, and it is one of the 15 biggest commercial ports in Italy with about 18 million tonnes of goods per year. The intense activity of multimodal transportation systems in the city center causes many problems to the urban mobility. For that, the Municipality of Messina is investigating innovative solutions for mobility management, and this paper provides a contribution to its current activity in this field.

The Municipality of Messina is making a significant investment on transport services,

[1]https://www.messinaoggi.it/website/2020/02/04/ap-cresce-il-traffico-passeggeri-e-dei-croceristi-nello-stretto-bilancio-
2019-piu-che-positivo

buying several IP high-resolution cameras for video surveillance, Traficams for traffic flow monitoring, cameras for vehicle plate recognition and so on. However, buying specific devices for each service implies high costs, both in terms of equipment purchase and in the management of a highly heterogeneous infrastructure. Furthermore, these devices may present several drawbacks. For instance, Traficams are quite expensive (the cost of each camera is about 2.000 €) and are not able to accomplish multiple tasks. Also, the resolution is quite low, therefore, at a certain time of the day, vehicle detection works poorly. On the contrary, using general-purpose devices customizing them to accomplish multiple tasks can be a more efficient and cheap approach. Also, moving computation tasks at the Edge of the system instead of in remote centralized serves can make the system more scalable reducing both delays in event detection and bandwidth usage. Our idea is to exploit general-purpose devices (e.g., high-resolution cameras) and employ Edge devices (e.g. Raspberry Pi 4 whose cost is only 70 €) to accomplish different user-defined tasks such as person counting, object identification, vehicle number plate recognition, traffic monitoring etc.

From a technical point of view, we propose a system based on the Function as a Service (FaaS) computational paradigm. FaaS allows us to define several minimal functions (e.g., vehicle counting or vehicle number plate recognition) and to run one or more instances of them on the same device at the same time in order to implement scalable application-s/services. In this paper, we present a vehicle counting system we developed to be run on a Raspberry Pi mod 4 equipped with a USB webcam with 1920×1080 resolution, and we compare the performance of the proposed approach with the performance of a Traficam device. Vehicle counting systems have been addressed in other works such as [18], [19], [20] and [21] however, to the best of our knowledge, it is the first time that FaaS is adopted on Edge devices for such purposes. We made different types of analysis to test both the feasibility of our Edge-based vehicle counting system and the applicability of these devices for heterogeneous purposes. Together with the vehicle counting system, we ran other concurrent services performing different tasks, such as OCR for vehicle plate recognition, to set up a useful service for mobility management, and we also considered several connected cameras.

The contribution of this paper can be summarised as follows:

- investigating how to set-up of a large scale general-purpose and distributed environment able to deploy on-demand services;
- proposing a vehicle counting system based on the FaaS paradigm and Edge computing to setup concurrent services;

- evaluating the performance of our system based on general-purpose devices also in terms of scalability whenever multiple services run simultaneously;
- discussing how our approach can provide real advantages in comparison with solutions based on specific monitoring devices.

3.1.1 State of the art

In this section, we report a review of related works in the context of Edge Computing and Smart Cities, and Edge Computing and Video Analysis, with particular attention to safety and traffic management goals. Some works are available in the literature on these topics, however only a few deal with the capabilities of dynamically changing the computational behaviour at the edge. Some of these works try to lightweight the algorithms and computational load. Moreover, edge computing is often considered as an extension of Cloud computations with limited resources (eg. see [22, 23, 24, 25]). Our work is different because we aim to adapt edge computing and load on purpose the conventional algorithms executed on the cloud.

Smart cities and traffic monitoring

A real-time traffic monitoring using IoT-based is discussed in [18]. Ultrasonic sensors are used to detect vehicle traffic levels at the lanes and to send them to the server for the analysis. A similar work is presented in [19]. The authors, by using Zigbee sensors send the traffic movements to a control center. A traffic signal automation system is presented in [20]. The proposed system, by using IR sensors is able to calculate the traffic density and to switch the traffic lights accordingly. A very interesting Edge-based video analytic system for traffic monitoring is presented in [21]. The proposed systems are very interesting, however, relying on the server for the analysis, in case of disconnection they, contrary to our system, are not able to perform the analysis. Furthermore, our system is based on FaaS that allows it to be easily reconfigured to accomplish multiple tasks. The works presented above are designed and implemented for a specific goal, therefore, to add or remove functionalities imply the redesign of the whole system.

Edge Computing and Smart Cities

An interesting Survey has been carried out in Edge Computing Enabled Smart Cities: A Comprehensive Survey [26], authors highlight the role of edge computing to realize the vision of smart cities analyzing the evolution of edge computing paradigms, with the objective to

classify the literature by devising a comprehensive and meticulous taxonomy. They identify and discuss key requirements, and enumerate recently reported synergies of edge computing enabled smart cities. Finally, several indispensable open challenges along with their causes and guidelines are discussed, serving as future research directions. Intelligent Offloading for Collaborative Smart City Services in Edge Computing [27] claims the weakness of long service response time and low QoS in scenarios with clouds. The authors remark that edge computing is nowadays integrated with the smart city to promote the inherent shortcomings of terminals in cities, to this they designed an intelligent offloading method for collaborative smart city services, named IOM. They try to achieve a trade-off among minimizing service response time, optimizing energy consumption and maintaining load balance while guaranteeing the privacy preservation during service offloading. A comprehensive and good analysis with mathematical models has been done.

Edge Computing and Video Analysis

The collaborative cloud and edge for object tracking is presented in [22], where ML algorithms are described in the approach of a partial processing of video acquisition on the edge, as compared with higher complexity of processing on the cloud. More results using different tracking strategies are reported. However, this work uses edge computing for limited purposes.

Edge networks created among devices are used in cooperative caching and video characteristics in [28] where opportunistic algorithms for sharing chunks of videos are considered. Here, we are much more in the domain of video analysis in terms of energy consumption during the sharing. At the Edge, Device-to-Device (D2D) is powered from batteries, since their simulation models try to understand better policies in different scenarios.

The Adaptive Wireless Video Streaming Based on Edge Computing cited in [29] is aligned with our approach that is the use of the emerging edge computing paradigm by deploying edge transcoding servers close to base stations. Their idea is to adopt the Dynamic Adaptive Streaming over HTTP (DASH) for edge transcoding where servers cooperate with the base station to transcode videos at a finer granularity according to the obtained users' channel conditions, which smartly adjusts the transcoding strategy to tackle with time-varying wireless channels. Varying at the edge computations is also our goal in this work.

Edge Computing and Video Analysis for Safety

A Lightweight Deep Learning based Intelligent Edge Surveillance system is presented from the authors in [24]. They report a lightweight neural network at the edge node, in particular, they use a modified version of MobileNetV2-SSD for the IIoT applications, which combine edge computing and cloud computing. It is interesting to see how a lightweight computation is achieved at the edge nodes. It can represent our future work in this perspective for intelligent behaviours at the edge.

Also, the deep learning-based couple-like cooperative computing method for IoT-based Intelligent Surveillance Systems is presented in [30] uses intelligent at the edge. Here the authors leverage the NVIDIA embedded development board, Jetson TX2; this board intended for artificial intelligence based on the NVIDIA Pascal system, whereas its compact size and powerful computing capabilities make it simple to integrate into various edge devices. This work detects helmet-wearing and confirms the workers' identities. In the future, they want to combine this method with some other to achieve cross-camera detection.

The research presented in [31] is in line with our current outcomes of the presented paper about the emergence of edge computing is highly promising in video pre-processing with an edge camera. Authors claim that a video surveillance system is a killer application for edge computing. They propose an edge computing enabled video usefulness system aimed at large-scale video surveillance systems. They provide the early failure detection and bandwidth improvement. Their model can locate a failure and send it to end-users on the fly. Their experimental results demonstrate the approaches in video usefulness model accurately detect failures that were collected from a video surveillance system with approximately 4000 cameras. This represents a useful case study for at edge computations.

Finally, in [23] authors recognize that many smart video surveillance approaches are proposed for object detection and tracking by using Artificial Intelligence (AI) and Machine Learning (ML) algorithms. However, it is still hard to migrate those computing and data-intensive tasks from cloud to edge due to the high computational requirement. Their hybrid Lightweight Tracking Algorithm to Enable Smart Surveillance as an edge Service describes intelligent surveillance as an edge service by proposing a hybrid lightweight tracking algorithm named Kerman (Kernelized Kalman filter) that is a decision tree based hybrid Kernelized Correlation Filter (KCF) algorithm proposed for human object tracking, which is coupled with a lightweight Convolutional Neural Network (L-CNN) for high performance. Their proposed Kerman algorithm has been implemented on a couple of single board computers (SBC) as

edge devices and validated using real-world surveillance video streams. The experimental results are promising that the Kerman algorithm is able to track the object of interest with a decent accuracy at a resource consumption affordable by edge devices.

Edge Computing and Video Analysis for Traffic

An interesting work has been conducted in [32] the Real-Time traffic light control system based on background updating and where the at edge detection is performed, similarly to what we describe in the following sections of this paper. However, they focus the excellent analysis on how to Adjust the traffic light based on traffic volume, no consideration is driven about the computation capabilities of the edge node, as we made.

Similarly to the above presented work, in [33] the video-Based Traffic Flow Monitoring algorithm for single-phase position at the intersection is treated. Their innovation is based on new algorithms to obtain background difference to guarantee edge information and extract a complete moving target, using Canny operator-based edge detection to preserve weak edges and taking advantages of median filter to remove noise obtaining a clearer processed image. No consideration is performed like in our case about the computation capabilities.

The difficulties of our analysis reside in the different setup necessary to put in place with all weather and illumination conditions. The work in [34] is interesting because opportunely address these issues driving video mining.

Edge Computing and Function as a Service

The benefits of a serverless architecture are discussed in [35]. The authors propose a FaaS platform for Edge computing and propose several application scenarios to test the applicability. The authors in [36] discussed the existing FaaS technologies and highlighted the use of the WebAssembly paradigm to enable the computation at the Edge. A federation of serverless Edge-based systems is proposed in [37]. In particular, the authors proposed a prototype for the oil extraction and analysed the performance of the proposed system. Qualitative and quantitative analyses of four open-source FaaS frameworks are discussed in [38]. In order to compare these frameworks, the authors deployed several services that push environmental parameters to a serverless edge-based platform. The results showed that all frameworks have similar results.

3.1.2 Motivation

Edge Computing is becoming a new challenging approach to provide advances in distributed and scalable systems. Providing services closer to end users is a great opportunity, especially when low network latency is a mandatory requirement and connections to the remote computing systems (e.g., Cloud datacenters [25]) are not always available. Intelligent systems at the edge help to foresee new opportunities also in smart cities, where the efficient management of computation and storage resources becomes essential for improving the quality of life of citizens.

Video and Audio processing at the Edge: an example of Virtual Device

The key idea of our work is to use cameras for multiple purposes, such as safety and security. The Police Department must have continuous access to live videos to constantly supervise the metropolitan area. An important aspect in this context is the concept of *Virtual Device* (VD): it represents a smart abstracted service built on-the-fly that, leveraging the device resources up to their physical limits, is capable to virtually configure new devices to satisfy the needs of multiple stakeholders. Let's assume that some high-resolution cameras are monitoring a public square with different perspectives. A VD could add Computer Vision capabilities to the cameras, like objects detection and tracking, or a 360 degrees global view, that can be achieved only considering all the cameras as a whole. Similarly, the deployment of many microphones for analyzing noise pollution can be virtualized in one VD able to provide, in real-time, the noise analysis in a public area. A new VD can be created merging the above functionalities to identify crowds or to analyze vehicles traffic. These examples show how, using the same physical infrastructure, it is possible to set up different virtual infrastructures according to the specific applicative needs.

3.1.3 Design

The reference architecture diagram for service provisioning on VD is shown in Figure 3.1. A general purpose Edge device can be equipped with several sensors or peripheral hardware to provide differentiated services, such as a high-resolution camera, audio capture systems and temperature, humidity or air quality sensors. Several configurable functionalities (implemented according to the FaaS paradigm) can vary from moving object detection to person counting, from vehicle traffic monitoring to license plate recognition, and can be activated on-demand. In particular, services are deployed using containers to take the

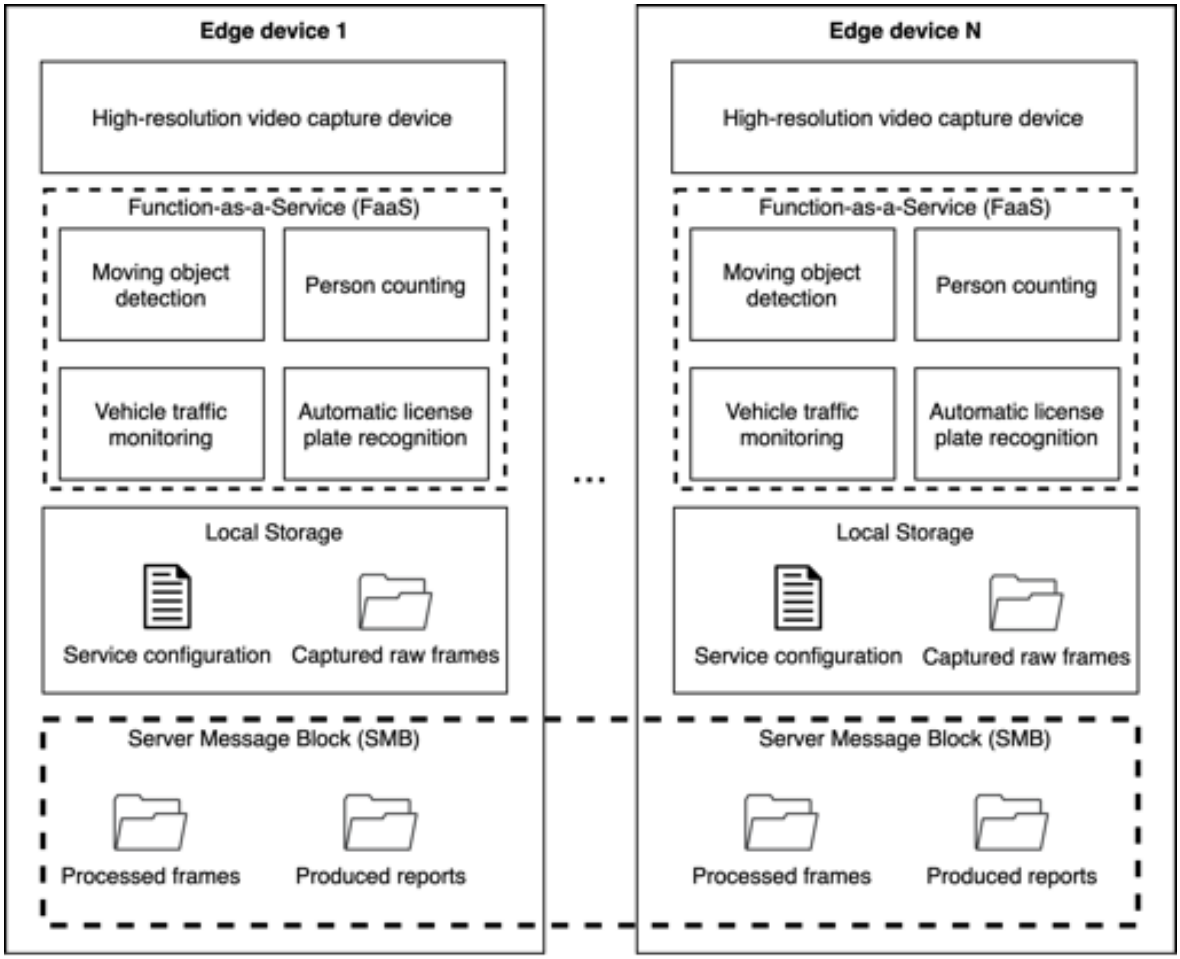


Figure 3.1: Architecture diagram service provisioning on VD.

advantages of virtualization technology allowing service portability, resiliency and automatic updates. The FaaS approach is adopted for an easy configuration on-the-fly of the device, especially if switching among different services is required during the time. A configuration file for each service is kept in a local storage area on the device, as well as the raw frames captured from the camera without any elaboration applied. Processed frames and reports are accessible remotely using a Server Message Block (SMB) protocol. This configuration can be replicated for any number of Edge devices available, for example, to cover the city center area. To secure the communication between the Edge device and the endpoint, a Virtual Private Network (VPN) is configured.

In this paper, we present how the proposed solution has been used to support urban mobility management in the city of Messina. In the video processing use case discussed in Section 3.1.2, we consider a generic video capturing device connected to a Raspberry Pi 4 which can monitor, record and store single frames or video streams. The last year, Raspberry Pi 4 has been released in the market with impressive hardware characteristics in the MPUs.

Nowadays it is possible to buy a fully equipped RPI 4 (4 cores cpu, 8GB of ram, giga-Ethernet, USB3 and USB-C, WIFI and BLE, etc.) for only 70 €. This allow us to implement an innovative, scalable and flexible solution at low cost. The specific workflow we followed for audio and video processing (and that will be further discussed in the next section) has been performed through the following steps:

1. **Requirement gathering**: based on high level municipality needs, the Edge device is equipped with different hardware systems (camera, microphone, sensors);
2. **System configuration**: based on specific municipality needs, a configuration file is arranged to indicate the necessary setup information, such as camera resolution, fps, running time and the output report to be produced;
3. **System initialisation**: this phase depends on the type of service that has to be executed. For instance, considering a traffic monitoring service, a certain number of frames are captured to build the *background layer*, the minimum size to identify a moving object is adjusted and the exit area is configured (this will be discussed in detail in Section 3.1.4); in case of vehicle license plate recognition, the area of interest is cropped to reduce the required computational effort;
4. **Service execution**: the configured service is run on the Edge device and reports are produced;
5. **Report analysis**: at any time, the operator can analyse reports produced to plan the required actions for a safer city.

In the next Section, implementation details for the described contexts are provided.

3.1.4 Implementation

For the implementation of the proposed system, we used the Python3 programming language and the most popular OpenCV (Open Computer Vision) image processing library. This library allows us to apply several filters to the frames acquired by the video source, whether captured via live streaming, video recording or static image, processing each frame as a single independent image.

With reference to the video processing at the Edge scenario, we aimed at controlling vehicles crossing a specific area in the city. To this purpose, a webcam is placed to monitor a specific road junction of the city. An example of a frame captured with a 640x480 resolution is

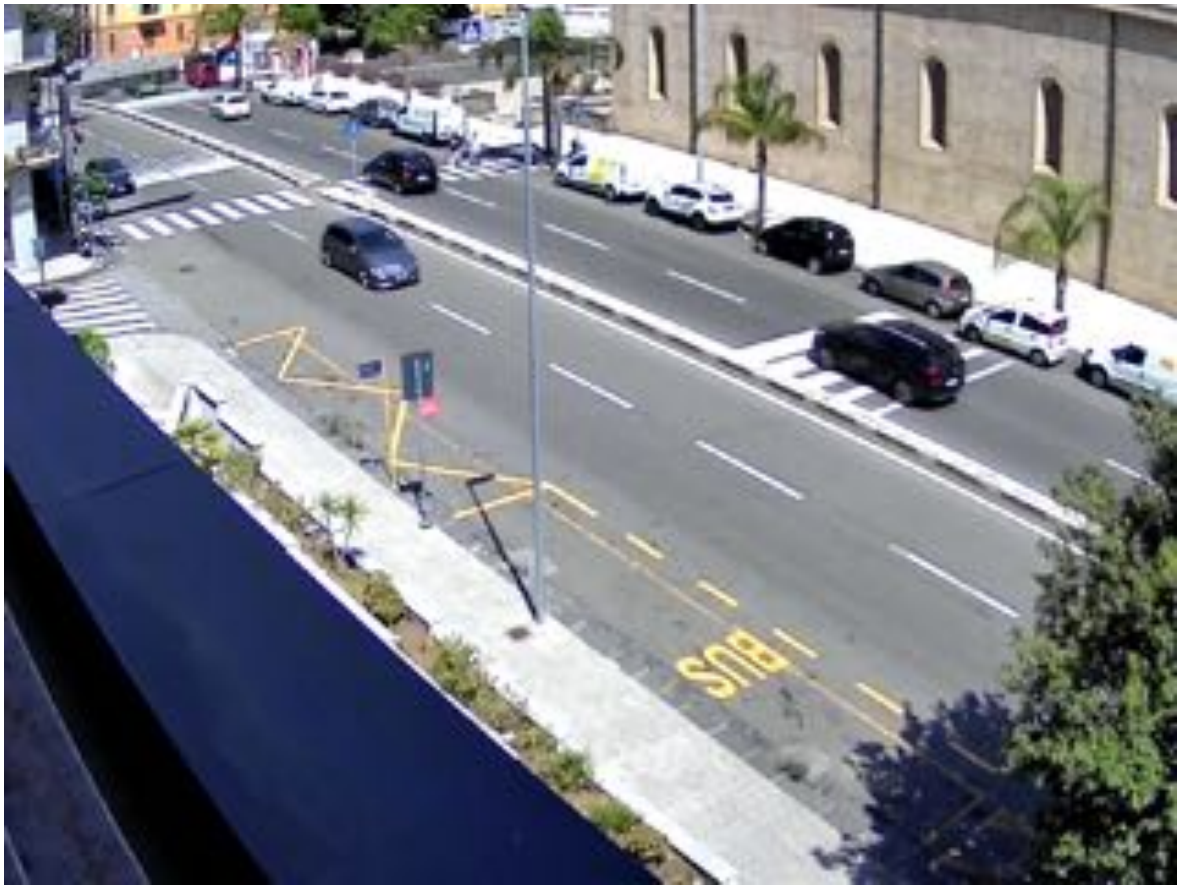


Figure 3.2: Sample frame captured at 640x480 resolution.

shown in Figure 3.2. A dedicated Web Server Gateway Interface (WSGI) has been developed with the Python web application framework Flask and can be accessed to look at the captured live stream of the camera device. All the service functionalities have been configured as FaaS to leverage the flexibility of on-demand capabilities based on municipality needs and can be easily switched. All the captured raw frames are saved in a local storage folder and kept for one week (the storage persistence is configurable accordingly to the memory capacity of the Edge device). The processed frames and reports are accessible remotely through the File Transfer Protocol (FTP) or accessing the directory with a network protocol like Samba. The number and effectiveness of the applied manipulation filters can be configured on-the-fly to improve the captured image based on the changing environmental conditions, such as day-night shift or sunny-rainy day. It is important to note that processing time and resource usage at the Edge are not affected by the changing of such filters.

To monitor the number of vehicles moving along a road, we use the Moving Object Detection (MOD) algorithms, and, in particular, MOD algorithms for background subtraction. In particular, during the initialization phase, the system is configured to analyse a specific

number of frames. In our experiments, this number is set to 500, but it can be calibrated. This step is required to identify the background composed by atmospheric conditions and natural or artificial ambient light. Frames are acquired and then interpolated to build the so-called background_layer. To track moving objects on the video, it is necessary to subtract the background_layer from the current_frame. The result of the applied filters is a black and white image where only non-static objects are present, as illustrated in Figure 3.3.

In case of noise caused by the poor image quality and non-static background objects (e.g. car headlights, clouds or tree leaves), these can be removed or reduced by using additional filtering techniques. To obtain a clear frame, the following filters can be applied: *Opening*, *Closing* and *Dilate*. As it can be seen in Figure 3.3a, tree leaves and cars are identified as objects non-belonging to the background_layer, and these are shown as white pixels in contrast with the static background composed by black pixels. This first level of noise can be reduced by applying the Opening filter (Figure 3.3b), that is a combination of the Erosion and Dilation filters. It usually remove small noise applying erosion to the resulting frame so the size of the foreground object decreases. On the contrary, the dilate filter used in our experimentation increases the white region in the image so that the foreground object increases to reshape the object size that previously shrank during erosion. The second filtering step is obtained by applying the Closing filter (Figure 3.3c): it uses the reverse approach respect to the Opening filter, where the Dilation algorithm is followed by the Erosion one. It is useful for closing small holes inside the foreground objects and helps to further clean the frame thus to have a more precise object identification. As last filtering step, the Dilate filter is applied to join remaining gaps and adjacent objects (Figure 3.3d).

To identify the moving objects we are interested in (e.g., cars, trucks,...), we need to isolate them from other mobile objects that are not of interest for the urban mobility management use case (e.g., people, cycles, tram wagons,...). To this aim, we track the blob in every subsequent frame to validate if the centroid of the minimum surrounding box is moving in the specific area or direction we want to monitor, such as a road junction or intersection. To do so, we setup a virtual area in the captured frame and we monitor if a moving object hits that area. The result of this analysis step is shown in Figure 3.4, where it is possible to distinguish blue boxes around cars (that are the moving objects we are tracking), red dot indicating the direction of moving objects (they come from the comparison of consecutive frames) and two green areas that represent the gates where the objects moving in the two opposite directions are counted.

The pseudo-code of the described algorithm is provided in the Listing 1.

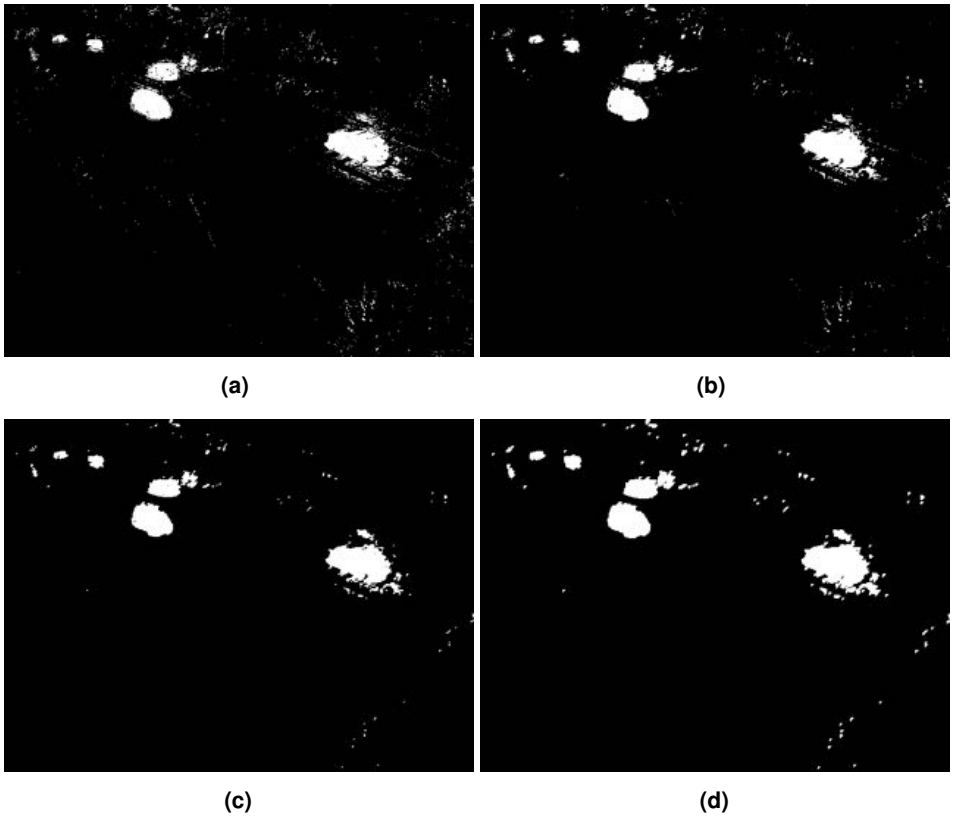


Figure 3.3: **a)** Background layer with no filter applied; **b)** Opening filter applied to remove background noise; **c)** Closing filter applied to fill object gaps; **d)** Dilate filter applied to join adjacent objects.

Algorithm 1 Moving Object Detection Algorithm Pseudo-code

Class *VehicleCounter*

method $movingObjectDetection(n\_background\_frames, pixel\_coordinates, video\_source)$

$obj\_coordinates \leftarrow \emptyset$

$moving\_objects \leftarrow \emptyset$

$detected\_objects \leftarrow \emptyset$

$background \leftarrow initBackground(n\_background\_frames)$

$monitored\_area \leftarrow [pixel\_coordinates]$

$video\_stream \leftarrow openVideoStream(video\_source)$

for all $frame \in video\_stream$ **do**

$objects \leftarrow (frame - background)$

$objects \leftarrow applyFilters(objects, [Open, Close, Dilate])$

for all $object \in objects$ **do**

$obj\_coordinates \leftarrow (object, object.position)$

if $object \in detected\_objects$ **then**

for all $position \in obj\_coordinates[object]$ **do**

$moving\_object \leftarrow moving\_object \cup position$

end for

else

$detected\_objects \leftarrow detected\_objects \cup object$

The described analysis can be slightly modified if different requirement are specified, such as if we want to monitor the number of vehicles in a given time slot, or to verify if a vehicle enters a restricted or forbidden transit area or it is moving in the wrong direction. All the above mentioned scenarios can be easy configured changing the software function deployed into the Edge device. This is performed exploiting a FaaS approach. For example, it permits to easy switch the service configuration during different hours of the day (e.g. road access forbidden during the night and allowed during the day) or depending on special requirements (e.g. road traffic changes during a riot).

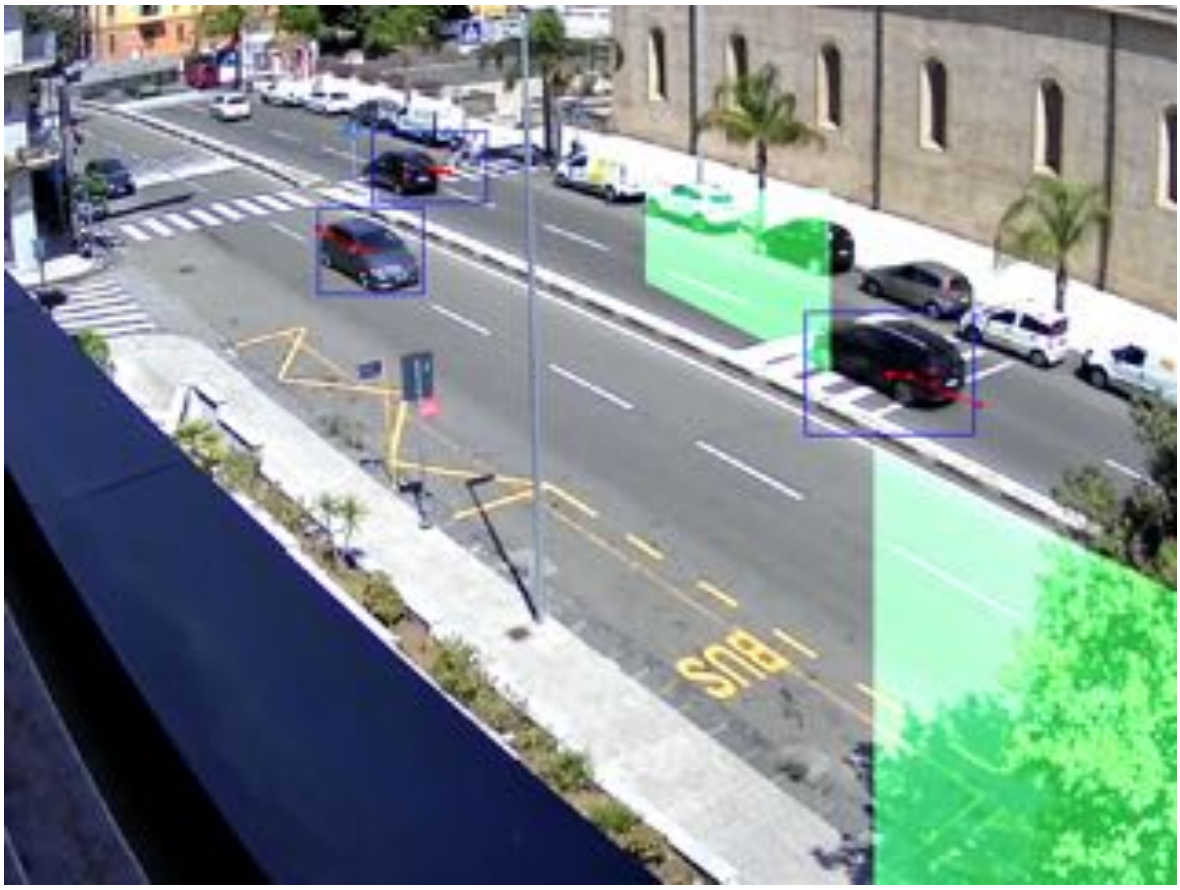


Figure 3.4: Captured frame with moving objects identified. Objects hitting the green area are counted as moving objects.

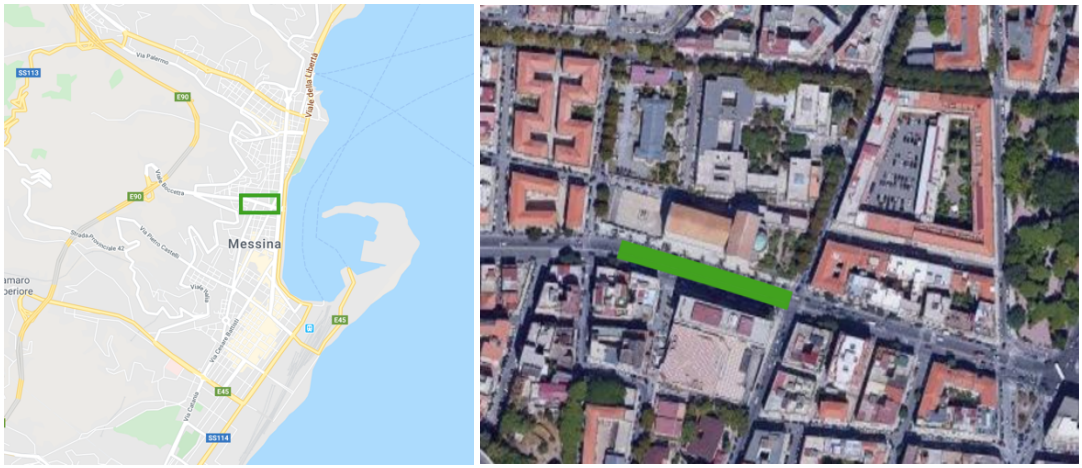
3.1.5 Performance Evaluation

In this section, we evaluate performance of edge devices to boost cameras from simple surveillance device to general-purposes device able to monitor the road traffic. In particular, we make two different types of analyses to test both the feasibility and the applicability of these devices for such purposes. More specifically, we make both a quantitative analysis comparing the performance, in terms of vehicle counting, of a Traficam with a Raspberry

Pi 4 and a qualitative analysis in order to test the flexibility of edge devices for supporting different tasks and workloads.

Reference scenario

As we discussed in previous sections, within the Mesmart project, the Municipality of Messina is monitoring the traffic of main roads. In our paper, we consider as reference scenario Viale Boccetta, one of the main roads that link the highway to the touristic harbour. Figure 3.5a displays the map of Messina, while Figure 3.5b shows a zoom in the area of Viale Boccetta. In particular, Figure 3.5b shows the monitored zone presented in this paper in green colour.



(a) **(b)**

Figure 3.5: a) Plan of the City of Messina (left); **b)** detail of the monitored zone (right).

In this scenario, in order to test the feasibility and the applicability of edge devices for vehicle counting, we compared performances of a Traficam with a Raspberry Pi 4, equipped with a Full HD webcam, installed at "PalaAntonello". Hardware and software characteristics of these devices are summarised in Table 3.1:

Investigation on Video Functionalities

In this section, we investigate the service quality of a general-purpose platform in comparison with a system built and dedicated to the specific task of vehicle counting. In particular, considering a one week monitoring, we divided the business days into six different time slots:

1. night: 01:30 - 5:30;

Table 3.1: Testbed characteristics.

Parameter	Traficam	Raspberry
CMOS type	1/4" color	1/3" color 2 MP
Resolution	640 × 480 pixels (VGA)	1920 × 1080 pixels (Full HD)
Frame Rate	25 fps	30 fps
Detection Zones	24	unlimited
CPU	N/A	Quad core Cortex-A72 (ARM v8) 64-bit SoC @ 1.5GHz
RAM	N/A	4GB
OS	GNU	Raspbian

2. sunrise: 5:30 - 9:30;
3. morning: 09:30 - 13:30;
4. afternoon: 13:30 - 17:30;
5. sunset: 17:30 - 21:30;
6. evening: 21:30 - 01:30.

These time slots have been chosen to take in account different light conditions on the road. Per each time slot, we calculated the average number of vehicles that transit in both directions (West-East and East-West). In order to validate the outcome of the deployed system, per each time slot and direction, we observed several samples of the recorded video in order to calculate the number of false-positive, false-negative and the accuracy of both the systems.

In Figure 3.6, the behaviour during the time slot "night" is shown. As the reader can observe, the performance of both devices is comparable. However, observing the video, we noticed that the accuracy of both devices is quite low. This is because the resolution of the Traficam is quite low and, therefore, some vehicles, especially the dark ones, are not recognised. With reference to the Raspberry Pi device, our algorithm is not able to distinguish vehicles from their own headlights and, thus, it does not increment the vehicle counter. With the increasing of illumination, the behaviour is different. Starting from 04:30 a.m., the number of vehicles detected from the raspberry, especially in the direction W-E, increases. This is much more evident during the time slot "sunrise" shown in Figure 3.7. In fact, issues due to headlights disappear and the raspberry is able to detect vehicles with an accuracy of 90%. During this time slot, light conditions are still not sufficient for Traficam. In fact, the number of vehicles detected by the Traficam becomes comparable to the Raspberry's one only at the end of this time slot, starting from about 9:00 a.m.

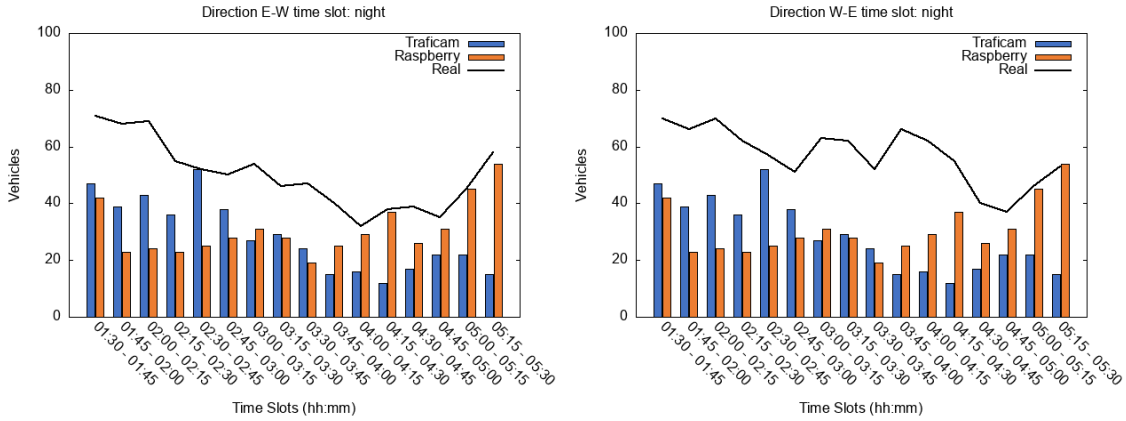


Figure 3.6: Comparison of Traficam (in blue) and Edge device (in orange) for vehicle counting during the night time slot. On left part of the figure the direction E-W is shown, on the right part the direction W-E is displayed.

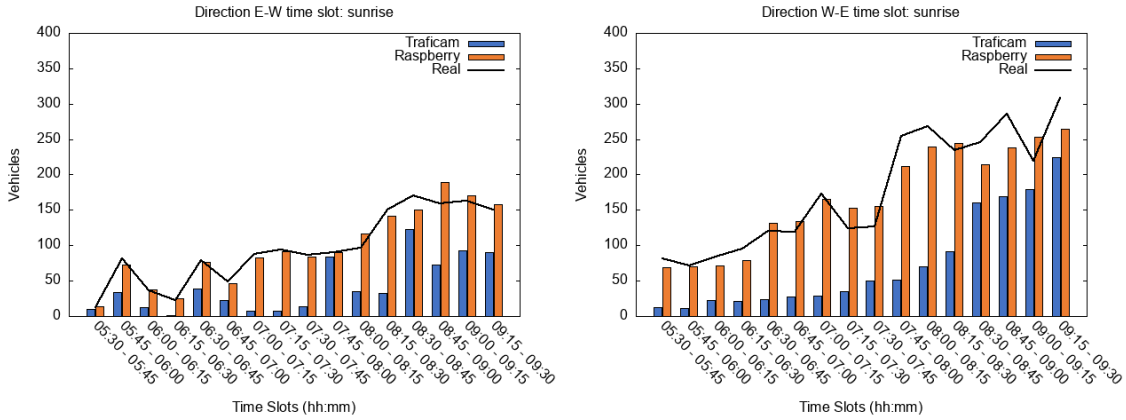


Figure 3.7: Comparison of Traficam (in blue) and Edge device (in orange) for vehicle counting during the sunrise time slot. On left part of the figure the direction E-W is shown, on the right part the direction is W-E.

In the time slot "morning", as the reader can observe in Figure 3.8, we can distinguish different behaviours. In particular, chart "E-W" shows two different behaviours depending on the position of the sun. From 9:30 to 10:15 the Raspberry-based system is able to detect more vehicles with respect to the Traficam. In contrast, since 10:15 the light conditions are good for both devices. There are only a few discrepancies on the number of vehicles detected by Traficam and Raspberry respectively. This is due to the presence of big vehicles such as trucks or buses and cars crossing two lanes. Considering the direction "W-E" is it possible to observe three different behaviours: i) From 9:30 to 10:00 the Raspberry-based system is able to detect more vehicles with respect to the Traficam; ii) considering the time period 10:00 - 11:30 the light conditions are good for both devices and we can note only a few discrepancies; iii) instead, starting from 11:30, the light conditions are not ideal for the Traficam, in fact, as we can observe in the figure, there are considerable differences between the results of both

devices. As we will discuss for the "afternoon" time slot, these discrepancies depend on the light conditions.

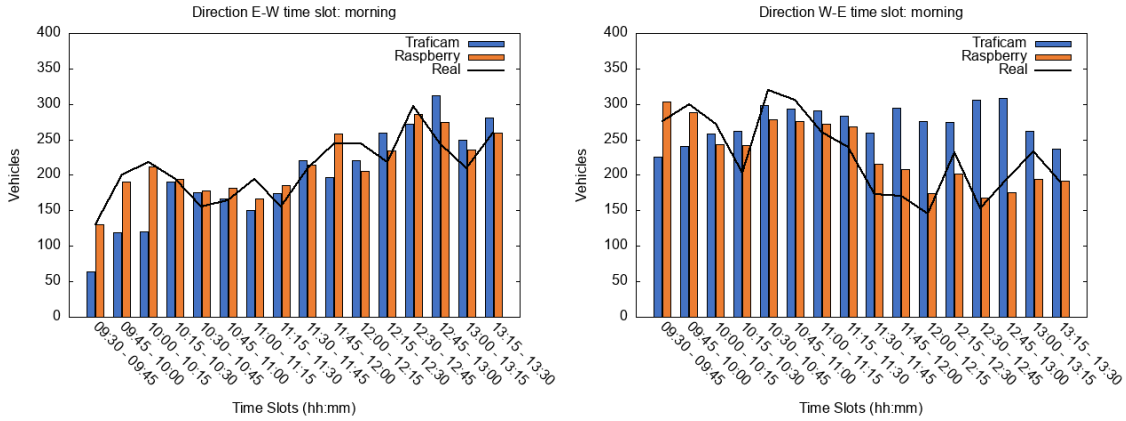


Figure 3.8: Comparison of Traficam (in blue) and Edge device (in orange) for vehicle counting during the morning time slot. On left part of the figure the direction E-W is shown, on the right part the direction is W-E.

During the time slot "afternoon", as the reader can observe in Figure 3.9, there is a big difference in vehicles counting especially for E-W direction at 2:00 p.m.. Observing the video we noticed that, due to light condition, one of the Traficam's detectors was blocked at the active position thus the vehicle counter was incremented constantly.

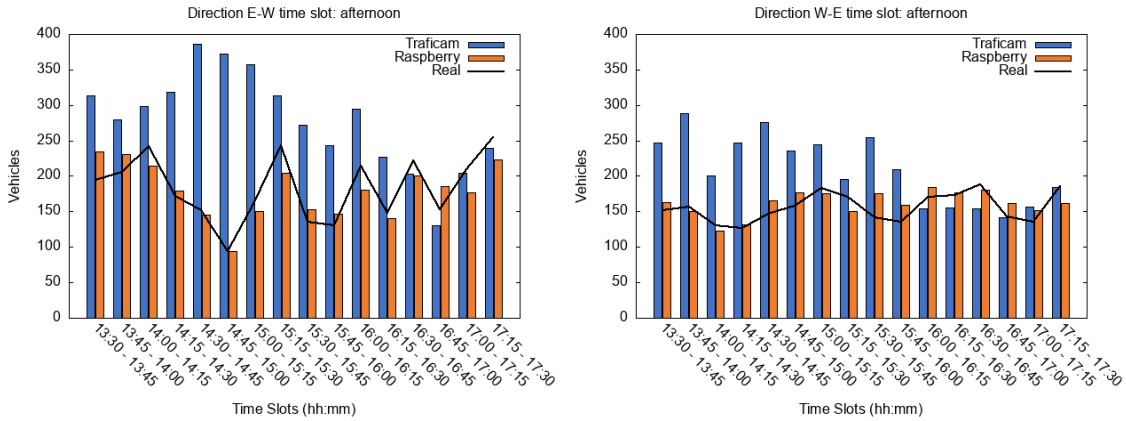


Figure 3.9: Comparison of Traficam (in blue) and Edge device (in orange) for vehicle counting during the afternoon time slot. On left part of the figure the direction E-W is shown, on the right part the direction is W-E.

In Figure 3.10 the behaviour of both devices during the time slot "sunset" is shown. The accuracy of both devices is quite good for a great part of the time slot. The only big difference is on the direction E-W after 8:00 p.m.. As the reader can observe in the figure, due to the absence of environmental light, the vehicle headlights become more prominent and the Raspberry is not able to distinguish the vehicles.

Finally, in Figure 3.11 the behaviour of both devices during the time slot "evening" is

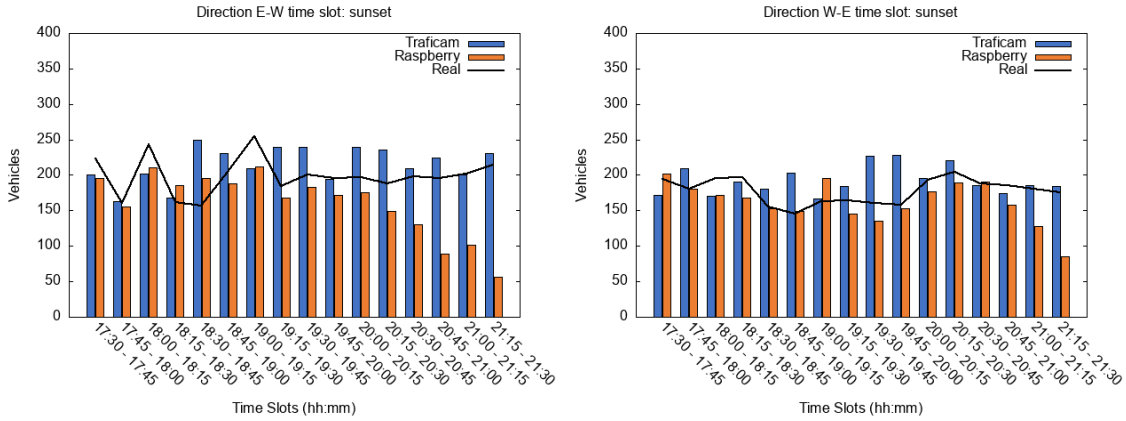


Figure 3.10: Comparison of Traficam (in blue) and Edge device (in orange) for vehicle counting during the sunset time slot. On left part of the figure the direction E-W is shown, on the right part the direction is W-E.

shown. During this time slot, we can observe two different behaviours: in the first part of the chart (till 10:30 P.M.) the behaviour is very similar to "sunset" time slot, in fact, the Traficam detects vehicles that are not visible for the Raspberry. Instead, in the second part of the chart, the behaviour of both devices is similar to "night" time slot. Both devices detect almost the same number vehicles but, in both cases, observing the video we can see that a great part of vehicles remains undetected.

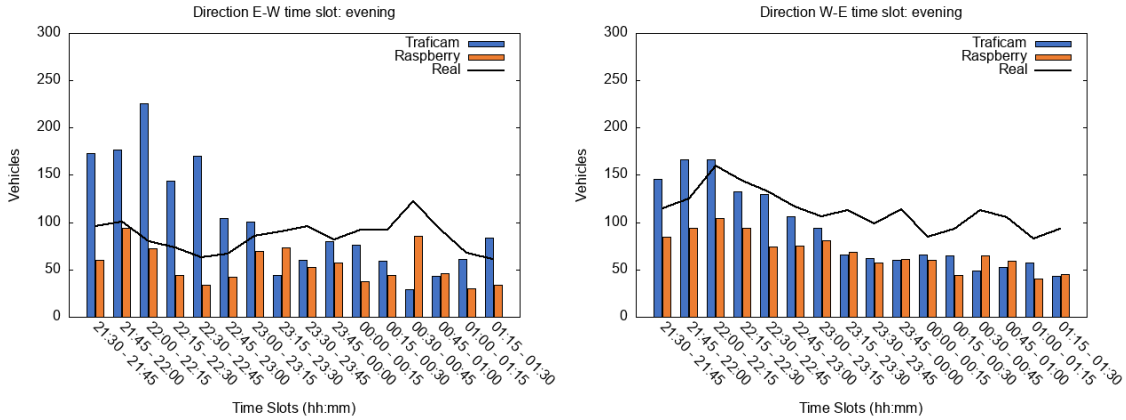


Figure 3.11: Comparison of Traficam (in blue) and Edge device (in orange) for vehicle counting during the evening time slot. On left part of the figure the direction E-W is shown, on the right part the direction is W-E.

Analyzing the outcomes of both devices, we found that the system based on Raspberry does not over-count the number of vehicles that pass through the monitored area. In particular, it performs better during the daytime reaching 95% of accuracy. During the nighttime, the system is not able to distinguish between vehicles and headlights therefore the accuracy of the system decreases. Different behaviour can be seen for the Traficam. In fact, due to the presence of multiple static detectors, it over-counts the number of vehicles especially around

2 P.M.

Performance assessment

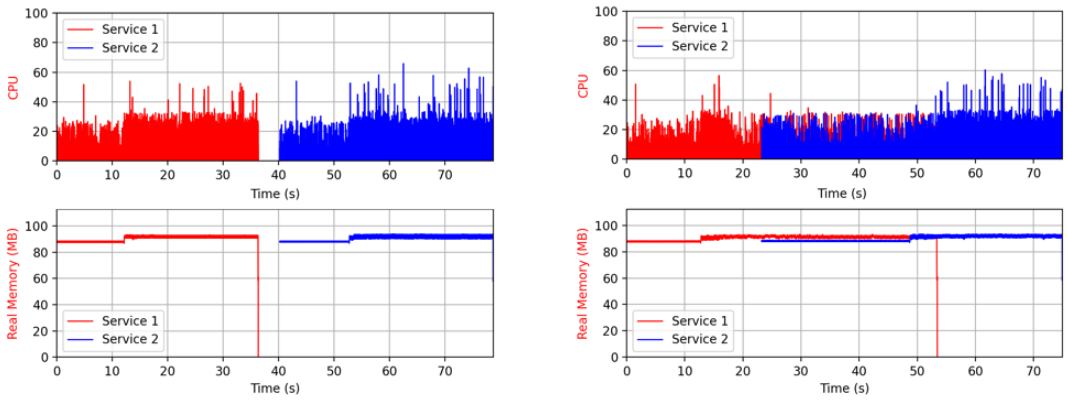


Figure 3.12: Resource comparison for two different services running on the general-purpose device. **a)** two services running with a minimum downtime; **b)** two services running simultaneously.

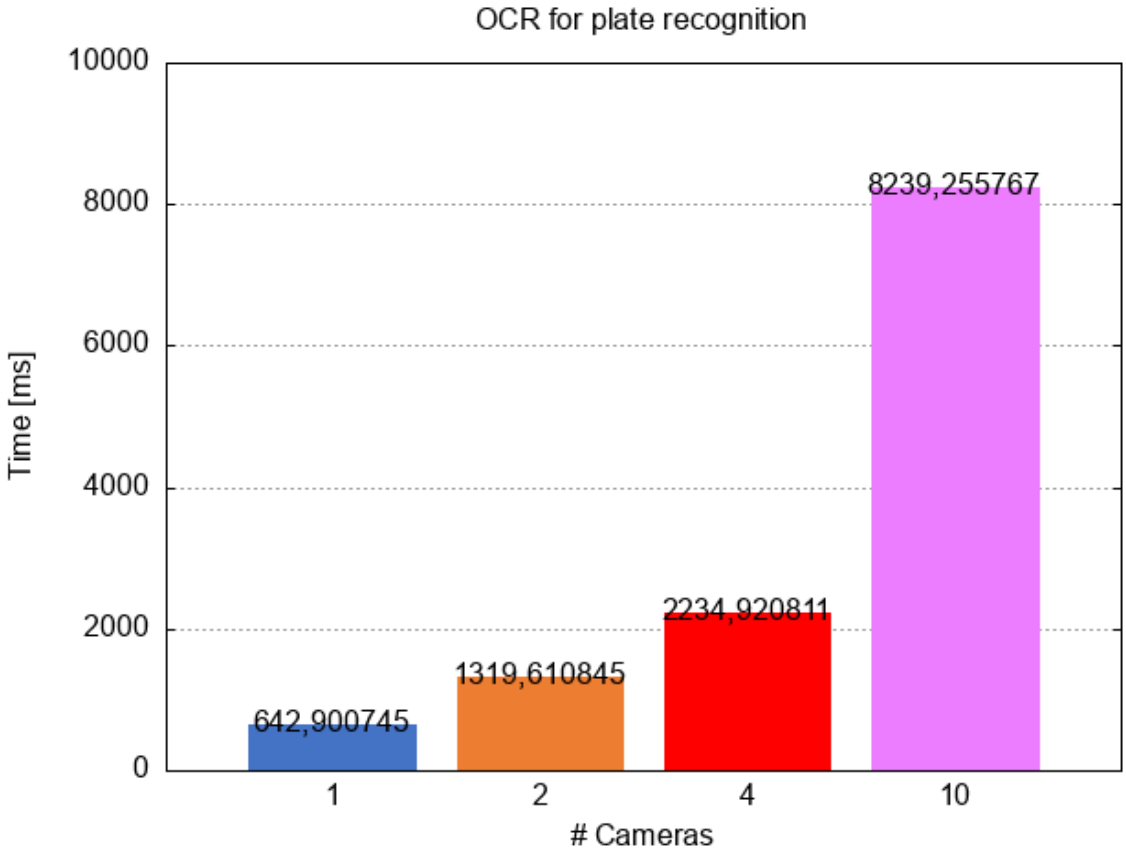


Figure 3.13: Execution time for performing the OCR for number plate recognition on Raspberry considering 1, 2, 4 and 10 cameras connected to the same device.

Performance assessment is aimed at understanding possible concrete usage at the Edge (eg. RPI 4), a platform conceived for hosting any kind of data processing (eg. Audio, Video, Sensing, etc) is able to fulfil several computation ad storage requirements. In Section 3.1.5 we investigated the service quality that a general-purpose platform can have in comparison with a system born and dedicated to a specific task. Here, we want to understand if a general-purpose platform can have enough hardware resources to come up with more tasks at the same time. We tested the device to run two different services of vehicle counting simulating the on-demand request that could be made by the Municipality stakeholders. In the first scenario, we simulate the case where there is a need to switch from one service to another. As indicated in Figure 3.12a the down-time is minimal and in just a few seconds the Service 2 is operative. The second scenario in Figure 3.12b presents a similar case but the two services co-exist for a certain period before Service 1 is switched off. As demonstrated, the device is able to manage both the services simultaneously.

In addition to the vehicle counting task, we stressed the Raspberry performing the OCR on frames for the number plate recognition considering 1, 2, 4 and 10 cameras simultaneously connected to the same devices. As the reader can observe in Figure 3.13, the execution time increases almost linearly within the number of cameras. Considering our urban scenario, we can consider acceptable the time required for processing up to 4 cameras on the same device.

3.1.6 Some remark

In this Section, starting from the requirements of the Italian project Mesmart, we investigated the use of a single edge device for accomplishing multiple tasks in the domain of video processing instead of buying a specific camera per each task. In particular, we implemented a FaaS solution that allows the users to run several tasks on the same device such as vehicle counting, number plate recognition, object identification etc. Our solution is composed of a full-HD webcam connected to the Raspberry Pi 4 running OpenCV algorithms. In order to validate our system, we made two different types of analysis: a qualitative analysis comparing the performance of the Raspberry with a Traficam, a device specifically designed for the vehicle counting, and a quantitative analysis running several concurrent tasks on Raspberry in order to discover if it has enough resources for managing multiple tasks. Results demonstrated that the Raspberry-based system, during the daytime, performs better than Traficam, in particular, it never over-counts vehicles, reaching an accuracy of around 95%. Different behaviour can be seen during the nighttime. In this case, both devices present almost the same performance in terms of vehicle counting, but the accuracy is quite low.

Finally, in order to test the applicability of edge devices for multiple purposes, in addition to vehicle counting, we ran the OCR for number plate recognition considering 1, 2, 4 and 10 concurrent cameras. From this analysis, we discovered that considering an urban scenario the Raspberry is able to manage up to 4 cameras. In future works, we plan to improve the vehicle counting technique in order to increase the accuracy of edge-based approach during the night time.

3.2 Edge-IoT mesh network

Internet of Things (IoT) has revolutionized how services are distributed in smart environments, approaching the computation where data are generated. However, IoT devices have limited computation and storage resources and, especially in large networks, reconfiguring them to undertake different tasks can be very difficult. In these cases, Edge computing represents a challenging solution to support IoT systems with more flexible management of resources. This research investigates how to push computation activities from the Edge to IoT, changing the behavior of IoT nodes according to application or system requirements. In particular, it is adopted the Multi-Hop-Over-The-Air (MH-OTA) update technology to enable the auto-configuration of IoT devices. Considering IoT nodes connected in a mesh network, a distributed and collaborative ecosystem is developed to perform an on-fly injection of code in IoT nodes, thus deploying new services whenever necessary and in an automatic way. A prototype of the proposed solution is implemented over a heterogeneous Edge-IoT mesh network, with particular attention to the dynamic configuration of end-devices with limited resources, such as the ones based on MicroController Units. Experiments are performed with the purpose to study the update phase of end-devices while they execute Digital Signal Processing.

3.2.1 Introduction

One of the major drivers of current digital transformation is undoubtedly the Internet of Things (IoT). This technology is revolutionizing our daily life by connecting several smart objects around us to provide innovative user-centric services. As reported by Strategy Analytics [39], smart devices deployed all around the world will be nearly 40 billion by 2025 and more than 1000 ZB of data will be generated by connected people, machines and things in the next years [40]. Furthermore, IoT is changing the way we interact with the surrounding environment, such as in smart homes, offices, exhibitions or cities, allowing us to access a

huge amount of information and, most of all, improving several aspects of our life, such as logistics, hospitality, healthcare systems, transportations, energy-saving industries and new solutions that need real-time reactions.

Most IoT applications follow the logical cycle shown in Figure 3.14, where four key functionalities are executed cyclically for: (i) collecting data from the environment (sensing), (ii) transfer data to a processing unit (communication), (iii) elaborating data (computation), and (iv) pushing a machine or device to operate (actuation). However, such functionality is not implemented in a single device. Indeed, sensing and actuation are performed by IoT systems, also called end-devices, whereas communication and computation are in charge of more powerful systems often labeled as Edge devices. The difference between both is more clear by taking a look at their hardware architectures:

- IoT devices are often built on embedded systems based on MicroController Units (MCUs) and are typically used in compact and low-energy equipment. MCU is a chip optimized to control electronic devices consisting of a single integrated circuit that is dedicated to perform a specific task. Examples of commercial devices include Arduino and ESP32.

- Edge devices are based on MicroProcessor Units (MPUs) and are typically used for low-complexity and low-latency computing purposes. MPU consists of a Central Processing Unit (CPU) linked with other surrounding chips that support various functions like memory, interfaces, and I/O. Examples of commercial devices include Raspberry PI 4 and Jetson Nano.

According to the most recent distributed and interconnected application deployment model, IoT solutions take support from Edge Computing systems deployed close to them. Edge computing is a computation paradigm arisen for moving Cloud computing benefits closer to IoT devices over a distributed network of microdata centers. It enables local computation and storage of critical data to face latency issues, lack of broadband connectivity, and reliability of IoT applications. Then, all the generated or partially processed data are sent to the Cloud for massive elaboration or long-term storage. Edge computing is able to reduce the amount of data exchanged with the Cloud, enabling computing data susceptible to latency very close to the source. However, designing system architectures spread over IoT, Edge and Cloud nodes is not trivial at all due to networking and consequent Quality of Service (QoS) issues. At the same time, the upcoming MCU is becoming even more powerful thanks to enhanced system-on-chips multi-cores with integrated high-speed WiFi and Bluetooth

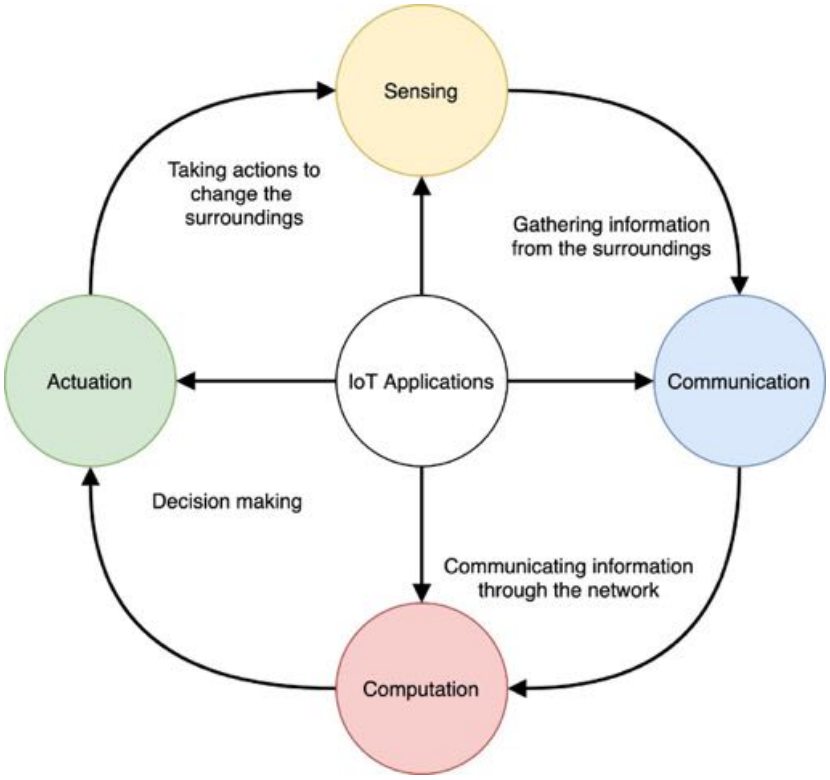


Figure 3.14: Overview of generic IoT applications' phases

Low Energy (BLE). For example, let us think about the recent Tensorflow announcement[2] related to the capability to make inference on MCU based device such as the Espressif ESP32, which is one of the cheaper devices currently available on the market. This trend is therefore promoting developers to increase the effort on building a more complex application on MCU-based IoT devices.

The new generation of IoT infrastructures can do massive usage of both cheap MPU and most powerful MCU devices useful to increase processing and storage resources at the Edge. All these components should be organized in a mesh network and should be able: to share data (both raw information collected from the surrounding environment and pre-processed ones collected from distributed storage systems); to communicate with each other for maintenance or reconfigurability purposes and to execute local processing. Edge nodes can also provide access to the Internet (e.g., to Cloud-based services) for implementing advanced and complex services and applications. Therefore, focusing on IoT based applications, the objective of this research is to discuss a technique to push processing from the traditional Cloud-based distributed systems (commonly deployed in High-Performance Computing (HPC) servers placed at "the center" of the network) to interconnected Edge/IoT end-devices, configured into a strongly dynamic BLE/WiFi mesh network. Another important aspect to

[2]blog.tensorflow.org/2020/08/announcing-tensorflow-lite-micro-esp32.html

consider about the new generation of IoT-based applications is that processing needs change over time. Reactive systems where actuations change the behaviour of the environment need to sense (and then process) data according to the specific conditions. For example, during the monitoring of a smart building, in case of fire, the rate of temperature measurements could be increased to analyze how the fire propagates, or in smart agriculture, the analysis of the soil could change according to the season or plantation. To address these needs, this research investigates heterogeneous Edge/IoT mesh networks, with special attention to the dynamic and on-fly configuration of nodes, with particular regard to MCU ones. Specifically, we adopted the Multi-Hop-Over-The-Air (MH-OTA) protocol to drive the reconfiguration of MCU nodes by a remote message, thus enabling the infrastructure to redeploy services whenever it is necessary.

In particular, we present the design and implementation of a system prototype of our proposed solution, also discussing some experimental results to assess it in a real heterogeneous mesh network. To summarize, the main contributions of this research are the following:

- discussing a computing paradigm-shifting, which gives more responsibilities to end-devices, intended from now on, and for the sake of simplicity, as Edge components;
- investigating cheap, low energy and dynamic mesh network fully deployed on the edge, including both MPU and MCU devices;
- analyzing how MH-OTA methods of distributing new software and configuration settings can support the dynamic behavior of update of nodes in the mesh network;
- evaluating the proposed approach for highlighting benefits in implementing Edge-IoT Mesh Network.

3.2.2 Related Works

In the recent past, many attempts to build IoT systems adopting customized approaches had mostly failed due to the intrinsic complexity of services deployed on distributed interconnected devices. In this context, the main building blocks of a general purpose IoT platform called Mainflux are discussed in [41]. Mainflux is built according to a microservice deployment style using a container virtualization approach. Binding is based on HTTP, MQTT, WebSockets and CoAP technologies. Although such a platform is interesting, it does not take advantage of the mesh network in terms of faster broadband, seamless roaming and setup simplicity. In fact, the advantages of hybrid mesh networks have been widely investigated in

the literature. A survey of the relevant technologies that may be suitable for mesh networking, either providing native support or being adapted subsequently was discussed in [42]. From the application point of view, it also identifies several application scenarios that can benefit from the deployment of hybrid mesh networks.

A piece of framework called "Wireless-Fog Mesh" for in-network computing of microservices in dynamic smart environments is discussed in [43]. Such a solution is based on a fog controller that exploits underutilized resources of mesh devices and network metrics for mapping a microservice to a fog node. The system is based on a DHT overlay of fog nodes also acting as distributed MQTT broker for disseminating data to fog nodes.

A wireless IoT hardware platform that uses LoRa as a communication protocol and is able to connect and manage several types of sensors was presented in [44]. Then, they investigate the advantages of combining into a mesh network devices with wide-area coverage (by means of LoRa) and devices characterized by ultralow-power consumption WPAN protocol (by means of the ANT technology). The cooperation of different IoT technologies in industrial environments, through a collaborative mesh network based on Bluetooth low energy (BLE) and long range wide-area network (LoRaWAN) is encouraged in [45]. In the proposed architecture, the IoT system is connected to a fog server responsible for preprocessing raw data that will be stored in a global cloud server, being available for consults at any time. In this paper, however, IoT devices are configured only to sense context data and send it to a back-end through the gateway for storage and processing.

An IoT mesh system where the physical network combines two subsystems: 1) a Bluetooth Low Energy Mesh network of boards equipped with sensors, and 2) a security monitoring system equipped with passive infrared sensors and cameras is investigated in [46]. These subsystems work together as deployed systems for a synology service. Even if the paper presents an interesting implementation of a heterogeneous IoT infrastructure, the behavior of IoT devices is static and configured a priori. Moving towards fully distributed approaches, [47] proposes a generalized framework for in-network computation, in which the data aggregation and processing steps are implemented through an artificial neural network distributed across the IoT mesh network. This work overcomes the limits of the in-network computation literature that focuses on simple functions. However, the required operations on the input data collectively perform by a predefined number of IoT neurons. The emphasis of this contribution is related to the data flow model for processing instead of the reconfigurability of processing tasks.

A lightweight virtualization, Information-Centric Networking (ICN), and service deploy-

ment algorithms to facilitate efficient service delivery in Community Mesh Networks, which are decentralized mesh networks built to satisfy the community's demand for Internet access and to provide services of local interest is discussed in [48]. The proposed decision engine selects the appropriate nodes for service instantiation based on constraints observed in network bandwidth, available hardware resources and network topology. To distribute the services over the network, the engine manages Docker containers which can be easily deployed at the edge and make use of the Named Data Networking solutions to distribute in-network container caching without any control entity. A microservice architecture model suitable for building IoT applications is discussed in [49]. Applications are composed of loosely coupled microservices. Such microservices are packed in Docker containers, deployed over a mesh of devices and the plenty of containers related to the same application are orchestrated using Kubernetes. These solutions are challenging but not applicable if there are MCU devices that can not hold containers.

The remote and automatic configuration of IoT devices especially in large networks is an open issue. In literature, some solutions deal with the automatic configuration of IoT devices. For example, an IoT protocol for the autoconfiguration of IoT devices in smart environments such as houses is discussed in [50]. However, the protocol addresses only the initial configuration of devices at the service setup stage. About the reconfiguration of IoTs devices and applications over-the-air (OTA), the evaluation of some solutions for the well-known embedded device Telosb with TinyOS is provided in [51]. TinyOS provides the necessary abstractions useful to install different tools such as Mires, Deluge, TinyLIME and so on. On the contrary, in our work, we operate at a lower layer considering devices that cannot execute an operating system. In our approach, the OTA protocol drives the reconfiguration of MCU nodes by a remote message.

3.2.3 Motivation

Computing is increasingly moving from the centralized Cloud to the decentralized and distributed Edge. Computing power also tends to grow, and it is not a foregone conclusion that forwarding data to Cloud is so essential for computation and data storage. However, poor network connectivity quality can be a big issue is, in fact, one of the best-known problems of Cloud-Edge-IoT interaction because reliable wireless connectivity is highly complex to be achieved, and dense device deployments further complicate operations. Moreover, the expectation around IoT devices with critical objectives is to work reliably, without errors, even in the most difficult environments. The continuous exchange of information between

devices in the same or different layers, therefore, increases the potential for error and network overload. Our vision leads us to think about independent solutions, much closer to data sources and timely drafting the observed event. As we have seen in recent years, many low-cost, low-energy devices are gaining computing power and connectivity. Each device, being it a micro data center or 'Things', can take charge of the computation, with the idealistic vision that each device can be as independent as it is collaborative.

The further challenge that arises is the management of a large variety of devices that are different from each other, use standards that do not always coincide, and have unique specific characteristics. As the scenario becomes heterogeneous as the complexity may increase exponentially when every custom Edge location is added. An innovative technology is needed for integrating end-devices together, allowing the provisioning of services and applications over them, and performing the efficient management of resources.

In this regard, the distributed, compact and low energy features of IoT infrastructures enable challenging deployments in both indoor and outdoor environments that would be uncomfortable or risky to be accessed by humans, such as forests [52], mountains, volcanoes [53, 54], mines, space and critical industrial areas. Considering possible quick changes in specific environments (e.g., fires, eruptions, melting glaciers, unexpected changes in weather, etc.), it could be necessary to track the events that are taking place from different perspectives, through a customized solution for collecting and processing data over a specific area. From a technical point a view, it means deploying distributed and re-configurable IoT infrastructures. However, the difficulty in accessing these areas makes it quite complex to reconfigure or deploy new services in scheduled times, let alone in time-critical needs. Thus, it is necessary to think about dynamic, multi-purpose remote IoT installations able to automatically react to environment changes.

Most of all, these technical advances, bring us to think about new opportunities in the setup of IoT infrastructures that take advantage of the new MCU potentialities, investigating also capabilities (and limits) of cooperation and dynamic reconfiguration of MCU devices. Moreover, we want to differently think about the end-devices, leaving out the idea to label them, using a dynamic approach where the end-devices can be part of the IoT or Edge system according to the constraints or current computation requirements. As result, these devices must be able to perform the full cycle shown in Figure 3.14.

This research presents a shift of computing paradigm into end-devices, even in the ones with low computation resources, in order to have more localized computing functionalities. In our idea, the new generation of IoT infrastructures can do massive usage of cheap MPU

devices and some MCU nodes useful to increase processing and storage resources at the Edge. All these components are organized in a mesh network and are able to collect and share both sensing raw data and stored pre-processed ones for maintenance or reconfigurability purposes, and to execute local processing. Edge nodes can also provide access to the Internet (e.g., to Cloud-based services) for implementing advanced and complex applications.

A prototype of the proposed heterogeneous Edge-IoT mesh network is implemented, with special attention to the dynamic and on-fly configuration of nodes, especially the MCU ones. In particular, we adopted the MH-OTA protocol to drive the reconfiguration of MCU nodes by a remote message, thus enabling the infrastructure to deploy new services whenever is needed and we provide some experimental results to evaluate the proposed solution in a real heterogeneous mesh network.

3.2.4 Materials and Methods

The paper aims at leveraging a MPU-MCU mesh network acting at the Edge with particular regard to the configuration of MCU nodes. Specifically, this Section is dedicated to a description of the different elements that were designed and deployed to enable the MH-OTA firmware update of all MCU nodes within the mesh network deployed on the Edge. Moreover, it will highlight the innovative aspects of our scientific work, that are:

- the distributed firmware Edge storage repository;
- the OTA update of several mesh network nodes driven by human needs.

Figure 3.15 shows the infrastructure overview of the proposed solution. Two mesh networks are firstly deployed on the Edge. One of them is dedicated to store firmware. We have decided to use MPU devices, such as Raspberry Pi 4, for accomplishing this goal and serving the firmware over the private network by means of a server web. The Raspberry PI 4 root node communicates with the root node of the second mesh network, which is dedicated to computation. In this case, the network is composed of MCU devices, such as ESP32 system-on-chips. Both networks elect the root nodes from time to time, allowing communication between them. The full cycle composed of sensing, communication, computation and actuation is fully functional over these two mesh networks.

A message protocol is instead used for enabling communication between the mesh networks and the service owner. New firmware is then uploaded into the MPU Edge network and served by server web; they are so installed via Multi-Hop-Over-The-Air (MH-OTA)

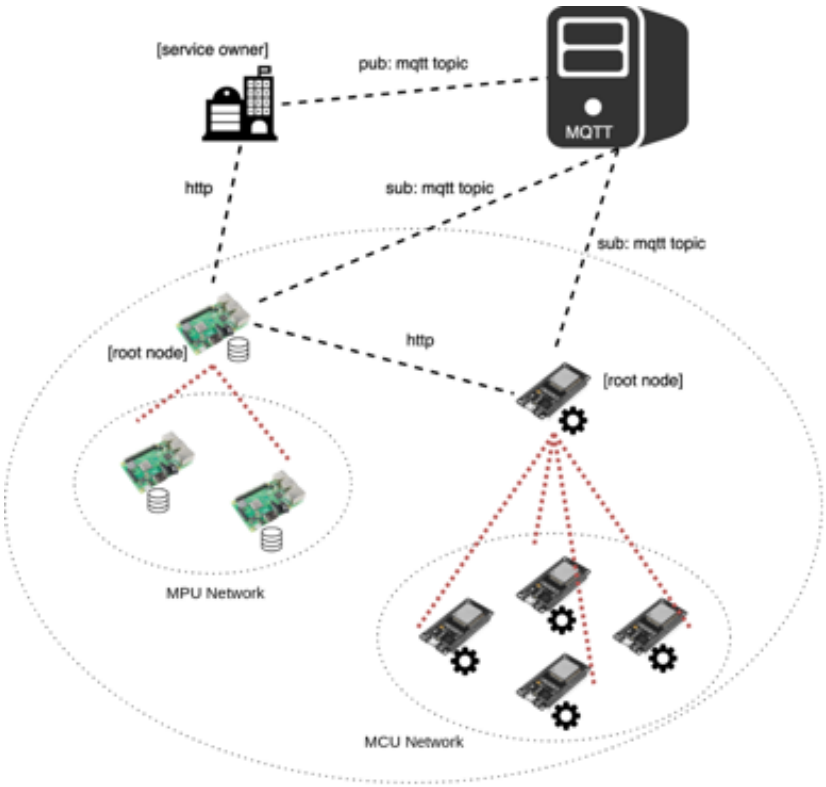


Figure 3.15: Infrastructure overview.

protocol on all MCU nodes, driven by a remote message. Our system, therefore, enables a full re-configuration of the MCU network, deploying new services remotely whenever is needed.

Distributed Firmware Repository

To allow data sharing between cluster nodes, GlusterFs has been chosen as a distributed and scalable Open-source volume management solution, thanks to the data maintenance and saving features in case of failures, using a unique hashtag for each file, stored within the file system itself. To use GlusterFS, it is required that each node, being this a client or a server node, has the service installed. Server nodes maintain the data in the form of volumes and act as a storage pool to expose the directories. Each client can then configure the service which is seen as a normal mounted volume by the operative system. GlusterFS supports various types of volumes as needed. Some types are suitable for scaling storage sizes, others for improving performance, and still others for optimizing both of them. Edge devices can benefit from the concept of High-Availability: a volume can be configured as a replica, to reduce the data loss risk, as copies are provided by each node. For example, with a configuration set on "replica 3" directive, each file is written three times across the nodes. Volumes can be configured as distributed when scalability is crucial and data loss is acceptable. To improve

data redundancy, high availability and high reliability there are several options: Replicated, Distributed replicated, Dispersed, and Distributed dispersed. For implementation details, please refer to official documentation [55].

For the scope of this research, all pros and cons have been carefully analyzed and it has been chosen the Distributed replicated approach, keeping in mind that the firmware file size to store is about 1 MegaByte, hence required storage space is easy to be obtained, and the critical requirement is to guarantee that the firmware is always available to be downloaded by IoT devices.

MPU-MCU Mesh Network Communication

The idea of updating the mesh network from the long-distance is of great importance when it is deployed on hard-to-reach locations. In this case, service owners need a reliable communication strategy for reaching the desired destination. As a consequence, we have designed to use a message protocol for exchanging information among the mesh and the external networks. MQTT is an OASIS standard messaging protocol designed for IoT integration. It is extremely lightweight and uses the publish/subscribe messaging transport, which is ideal for connecting remote devices with a small code footprint and minimal network bandwidth. It is low-energy and compared to RESTful protocol, MQTT helps to consume less power[3], turning it into a optimal fit for IoT devices. Moreover, a well-designed set of topics leads to good scalability policies.

Design choices have led us to create a topic based on the mesh network ID in the form of: */MESH-UUID/ota/update*. Proceeding in this way we have the opportunity to reach out to the root node of the specific mesh network, scaling over multiple ones whenever other networks are deployed. The payload is made standard thanks to a JavaScript Object Notation (JSON) encoding, allowing passing multiple arguments. To start the OTA update the following JSON document is required:

```
{
  "ota_endpoint": "http://PRIVATE_IP_ADDRESS/FIRMWARE_NAME"
}
```

Remembering that both the storage-based mesh network and the computation-based mesh network are located on the same private network, the endpoint sent over the message protocol is only accessible by them-self.

[3] www.bevywise.com/blog/mqtt-vs-rest-iot-implementation

Adoption of Smart MCU End-Devices

As already mentioned in Section **??**, this work aims at giving more responsibility to IoT end-devices for enabling the full cycle (sensing, networking, computation and actuation) shown in the Figure 3.14. The MCU choice was then fundamental for having a low energy device with good computing capacity. Moreover, taking in mind the wish of remotely injecting new services, software development and deployment flexibility was also strongly required. Correspondingly, we have chosen to use the Espressif Systems' ESP32 as an end-device, which is a low-cost, low-power system-on-chip microcontroller with integrated Wi-Fi and dual-mode Bluetooth. ESP32-WROOM-32 contains two low-power Xtensa 32-bit LX6 microprocessors individually controlled, and the CPU clock frequency is adjustable from 80 MHz to 240 MHz. The chip has also a low-power co-processor that can be used instead of the CPU to save power while performing tasks that do not require much computing power. All software components run on freeRTOS, i.e., a free real-time operating system with a lightweight open-source TCP/IP stack, such as LwIP.

Each MCU end device is part of a mesh network built atop the WiFi protocol. The mesh nodes simultaneously act both as the access point for enabling multiple downstream connections and a station for maintaining a single upstream connection, resulting in a tree network topology with a parent-child hierarchy consisting of multiple layers. Each node can transmit packets to other nodes through one or multiple hops but simultaneously serves as relays for other nodes. Therefore, the network configuration process becomes independent of routers.

The implementation we have built on top of the Espressif's Mesh Development Framework (ESP-MDF) follows the automatic root node election via the physical layer protocol. Indeed, when devices start up in the network, they look for others in the WiFi range. The network is then built layer by layer, starting from the root node, which will be elected according to the received WiFi power signal. Such a node serves as the only interface between the mesh network and the external IP network, especially the repository server web. The self-organizing property of the network lets itself elect a new root node when this is detected as broken.

Firmware Injection into MCU Devices using the MH-OTA

The firmware injection is what makes the entire mesh network re-programmable and suited for general-purpose applications. Such a mechanism is making possible by the MH-

OTA update, which allows an end-device to update itself and other nodes distant one or more hops, according to data received over WiFi in run time. The MCU end devices are configured with a partition table including two OTA partitions: one for applications and one for data. As a result of the MH-OTA update, the OTA data partition is updated to specify which OTA application partition should be booted next. As a fault tolerance system, the implementation has a rollback policy for always keeping the device working. Therefore, if the injected application has critical errors, the rollback policy allows starting the previous firmware version. The MH-OTA update starts from the root node, triggered by a MQTT

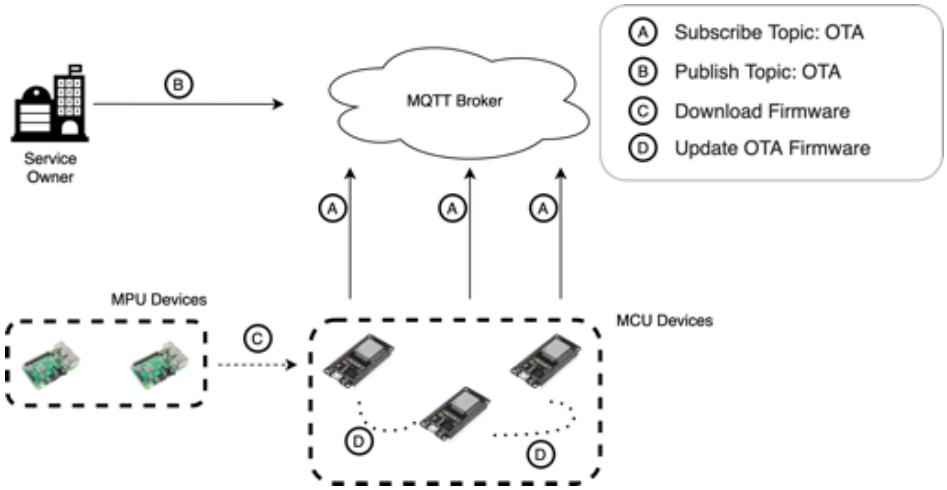


Figure 3.16: Device OTA Firmware update.

message sent by the service owner. The Mesh is indeed passive, waiting for an update pushed by the external network. However, it continuously listens for starting the process. The root node then follows the steps below for distributing the firmware over the mesh network:

1. requires the firmware contacting the server web hosted on the edge
2. writes the firmware on the unused OTA partition
3. splits the firmware in fragments and sends them to all nodes
4. when all fragments are received, asks all nodes to complete the update just restarting

3.2.5 Results and Discussion

The goal of the experiment is to understand how the MH-OTA update performs in terms of time while the involved node is also committed to computational tasks. Therefore, we implemented five firmware versions, in which the nodes perform tasks increasing the operation rate (0.1 ms, 1 ms, 10 ms, 100 ms, 1000 ms). Another firmware deactivates any

task, leaving the node idle and representing the benchmark of the experiment. The MH-OTA update message is sent following a task execution period of 10 seconds. All the experiments are repeated 20 times.

Through these experiments, we aim to study the Mesh Network behavior during the MH-OTA update while any node executes a digital signal processing (DSP), floating-point arithmetic operations such as sum, subtraction and multiplication. The FFT is one of the most commonly used operations in digital signal processing to provide a frequency spectrum analysis. The Fast Fourier Transform (FFT) computes the Discrete Fourier Transform of input much faster than computing it directly. In other words, the FFT reduces the number of computations needed for a problem of size N from $O(N^2)$ to $O(NlogN)$. [4]

The performance metrics of relevance for the experiments are the following:

average of download time is the time required for downloading the firmware within the root node. It starts when the trigger event happens and it ends when all bytes are downloaded and stored into the OTA application partition. The metrics gathered while the root node performs both the firmware download and a computation task are compared with the metric gathered when the root node is only committed in the downloading process (idle).

average of routing time is the time required to propagate the firmware from the root node to all the mesh network's nodes. It starts when the firmware is downloaded and stored into the OTA application partition and it ends when all firmware's frames are propagated through the mesh network. The metrics gathered while the nodes perform both the firmware propagation and a computation task are compared with the metric gathered when the node is only committed in the routing process (idle).

In order to carry out the above-mentioned experiments, the solution described in this paper has been tested over two networks, where we have changed the number of nodes and layers. Both experiments have the same storage mesh network, composed of two devices based on Raspberry Pi 4 Model B with a Quad-core Cortex-A72 (ARM v8) 64-bit SoC @ 1.5GHz, 4GB LPDDR4-3200 SDRAM, 2.4 GHz, and 5.0 GHz IEEE 802.11ac wireless. They are configured for using GlusterFS and share a repository of the firmware. The firmware size is 1052576 bytes, and it is served by a simple server web built on Python 3.7. The MQTT broker is installed on a server with an Intel(R) Xeon(R) E-2124G CPU @ 3.40GHz and 32GB SDRAM.

[4] https://towardsdatascience.com/fast-fourier-transform-937926e591cb

The computation mesh network is based on ESP32-WROOM-32 with two low-power Xtensa 32-bit LX6 microprocessors and a lightweight open-source TCP/IP stack.

The implementation is fully based on the Espressif Mesh Development Framework, a networking protocol built on top of the Wi-Fi protocol. It runs on a free operating system (freeRTOS) and executes the processes in tasks. That means, only one of them within the application can be executing at any point in time and the real-time RTOS scheduler is responsible for deciding which task this should be.

3.2.6 Two Layers Network

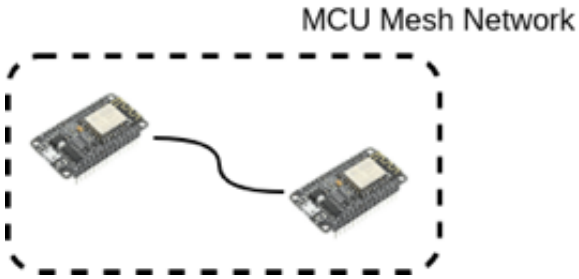
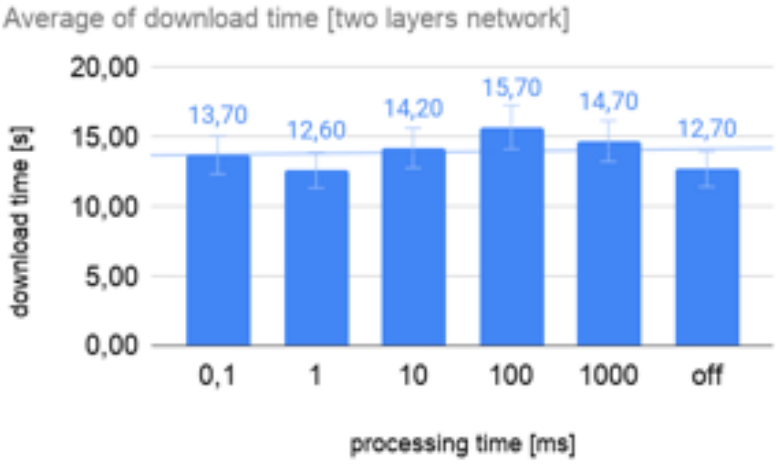


Figure 3.17: Example of two layers network. One of the nodes is elected as root in run-time.

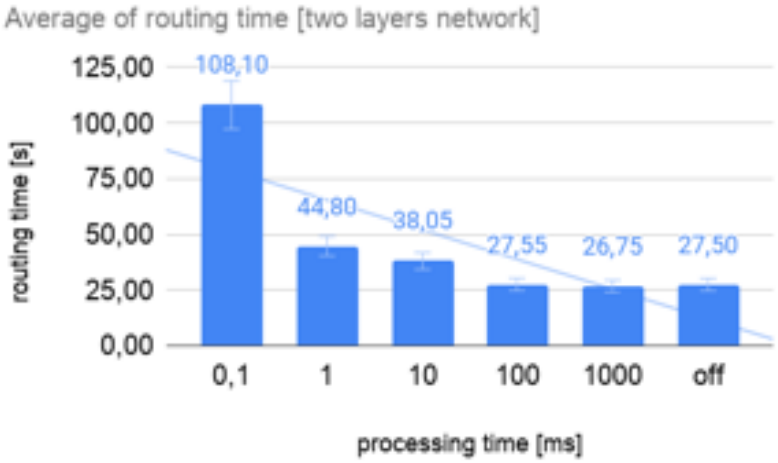
The first tested network is composed of two nodes organized in two layers as shown in Figure 3.17. One node is elected as root in run-time, whereas the second one is attached as a child. The routing table then follows a direct connection between the two nodes.

Figure 3.18a shows the average download time. On the x-axis, it reports the processing time used over the computation tasks, whereas on the y-axis it reports the download time in seconds. Considering also the confidence interval, the comparison of the download time among any processing time experiments, and mostly with the "off" experiment (when the root node is idle) shows a constant behavior. This means the download time is not affected negatively by the concurrent computation of specific tasks.

Figure 3.18b shows the average routing time. On the x-axis, it also reports the processing time used over the computation tasks, whereas on the y-axis it reports the routing time in seconds. The firmware is then split into chunks and sent over the network. The routing time shown in the Figure 3.18b has an exponential behavior. Indeed, by decreasing the task operation frequency, the routing happens faster. It is interesting to highlight the lower bound that starts with operations at 100 ms and continues with nodes in idle. This means the routing time is not affected by specific tasks when they have a period of operation greater than 100 ms. On the other hand, a real-time task (0,1 ms) affects the timing needed to propagate the



(a) Average of download time.



(b) Average of routing time.

Figure 3.18: Experiments of two layers network.

firmware all over the network.

3.2.7 Three Layers Network

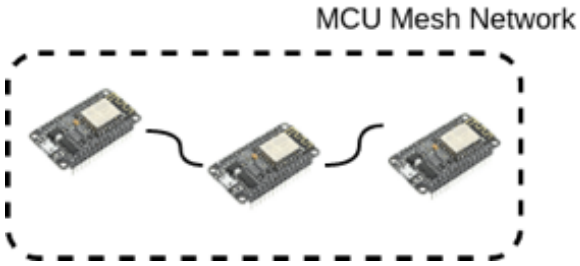


Figure 3.19: Example of three layers network. One of the nodes is elected as root in run-time.

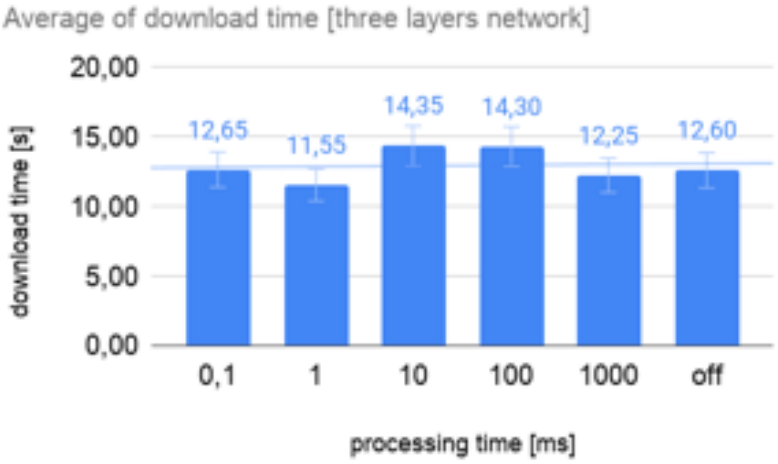
The second tested network is composed of three nodes organized in three layers as shown in Figure 3.19. One node is elected as root in run-time, whereas the second one is attached as a child. The third node is a leaf of this particular tree, which takes the shape of a chain. The routing table then has a node two hops distant from the root.

Figure 3.20a shows the average download time. On the x-axis, it reports the processing time used over the computation tasks, whereas on the y-axis it reports the download time in seconds. Figure 3.20b shows instead the average routing time. On the x-axis, it also reports the processing time used over the computation tasks, whereas on the y-axis it reports the routing time in seconds. The times shown both in Figures 3.20a and 3.20b confirm the trends highlighted on the previous experiment. The download time is again constant, as the linear approximation line shows in Figure 3.20a. Indeed, just as we expected, the firmware download is independent of the network size and configuration.

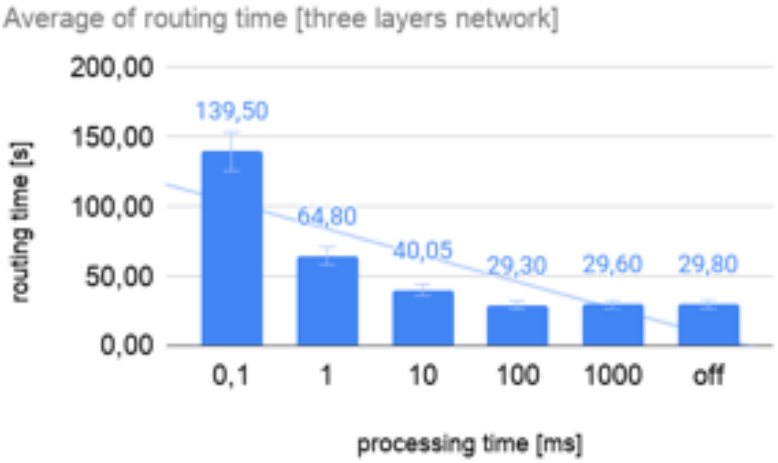
On the other hand, although the exponential behavior is still present, the amount of time has increased at any experiment, and mostly in experiments with a shorter period of operation. This means that by increasing the network size, and mostly the number of layers, we might expect the scalability deterioration. In this case, is important acting with suitable routing algorithms and topology. On the other hand, the lower routing time gathered when the nodes have a period of operation greater than 100 ms is a good starting point for building re-programmable nodes without interrupting the computation and staying within an acceptable time.

3.2.8 Future Works

The new computing paradigm discussed over these pages changes the way think about the end-devices. Leaving out the idea to label a device as part of IoT or Edge systems



(a) Average of download time.



(b) Average of routing time.

Figure 3.20: Experiments of three layers network.

from now on, what is taking importance is the capability to enable the full cycle of sensing, communication, computation and actuation, which is typical within any IoT system. Still, the capability to re-adapt not only the end-device itself but above all the entire mesh network according to events that are taking place on the field is also a valuable point of interest.

However, there are also reasons behind many research challenges in implementing that kind of Mesh network. The following list is intended to summarize only the issues that came to our mind during the writing of this work and is not intended as an exhaustive list. We will use the following list as a roadmap of our future works.

BLE Mesh Network The Mesh Network discussed so far is strongly based on WiFi protocol. Nevertheless, this solution does not really satisfy in full measure. First of all, the WiFi protocol is cost-inefficient. Such a device works well for short ranges, resulting in a largest number of nodes. If on the one hand, it increases the investment spending in network provisioning, on the other it makes the routing table more complex, because of the hops increment. Moreover, WiFi is not very power-efficient, usually lasting about 10 hours. It is against this background that we would like to move towards BLE. As ideal technology for industrial IoT use cases, it requires relatively low-cost and low-power scalability, reliability, and performance. It enables many-to-many device communications and is optimized for creating large-scale device networks.

Hybrid Mesh Network The proposed solution keeps two different mesh networks alive in a collaborative context, which perform two different jobs according to their hardware characteristics. We are talking about the mesh network built around microprocessors, such as the Raspberry Pi 4 used in our experiments, and the MCU network made up of ESP32-based devices. This difference is due to the better storage and web services capacity of the microprocessors, as well as the speed of execution of the MCU devices when specific applications are required. This distinction, however, is a bit narrow. Our vision is to hybridize the mesh network by creating only one that includes both types of devices. What we expect is better communication between the parties, which does not necessarily have to go through the two root nodes (which creates an inevitable bottleneck when updating very large networks). The alternating distribution of storage and computation nodes, allows the web servers to be reached through multiple hops and not an external IP network. The result is an isolation of the network that relies on better security, organization, and response times.

Self-Configuration There are several applications of a mesh network where devices must

be uniquely recognized, also based on the distance from them. Let's assume that we install a mesh network for outdoor navigation, which does not necessarily have to be performed by a human being, but also by a machine, such as a drone or a self-driving car. Under such conditions, we want to avoid signal overlaps or dead zones, which could result in an outage of service. Obviously, moving the devices once installed may not be possible, but it is instead useful to modulate their transmission power in run-time. In fact, there may be various factors that influence the transmission and reception of wireless signals, whether they are generated by WiFi or Bluetooth. For example, varying environmental conditions result in an uncertainty of the transmitted signal. The power value is known as RSSI, a measurement of the power present in a received radio signal. This value should be modulated during the network configuration phase and updated from time to time following changes in the surrounding environment. We call this property self-configuration.

Self-Pulling Typically, the mesh network properties are limited to self-organizing and self-healing abilities. The first one allows the devices to independently organize themself on network topology (tree or graph), i.e. electing a root node and all the underlying layers; whereas the second one is the capability to network recovery, i.e. when a node unexpectedly broke down the network changes the routing table for reaching all underlying nodes. Already with the self-configuration we have tried to extend this concept, but herein we would like to go further. The proposed solution uses a message-driven approach for starting the MH-OTA update when it could be managed by the network itself. Let's imagine that a network is monitoring public order situations, such as terrorist attacks, in a large geographical area. As such, the network will be spread over several neighborhoods, using audio and video detection sensors for facial recognition, voice recognition, and suspicious movements and sounds. At a specific moment, a suspicious event within a neighborhood is confirmed through sound recognition. As a consequence, the network would like to re-configure all the nodes of a nearby neighborhood to dedicate all the computing resources to the same function and certify the presence of the event also in that geographical area. Certainly, a real-time system cannot afford to wait for human intervention for re-configuring the nodes. Therefore, the only viable way is updating the selected nodes with the firmware already present and served through the storage nodes. This property is called self-pulling and allows a mesh network to deploy new functionality based on an event.

3.2.9 Some remarks

This research deals with the recent advances in MCU computing power have changed the way we think about the end devices. They are usually labeled as non-smart IoT devices, confining their capacity within the sensing and actuation features. However, such devices are much more. The computing capability has increased over time, arriving at enabling inference on MCU. As a consequence, we have aimed to give more responsibility to them.

The approach proposed on these pages organizes the end-devices in mesh networks, a solution of collaborative routing that exploits their potential. In particular, we have seen what are the needs of a network deployed on inconvenient or hazardous locations, where the human cannot easily access the devices anymore and therefore change their behavior using the classic way.

On the other hand, the experiments carried on a mesh network based on ESP32 system-on-chip supported the thesis that such devices are able to perform specific operations while their behavior might be changed over time. The injection of the new firmware is indeed efficient in terms of routing time, considering operations executed with a frequency less than 10Hz.

3.3 Secure service orchestration through Blockchain

In the last decade, the increasing number of devices connected to the internet has allowed the development of new technologies for distributed computing. IoT devices have reached computing capacities superior to personal computers of a few years ago, and their small size, low cost and lower energy consumption have made them fundamental for the new technological evolution. This research analyzes and compares different solutions of serverless paradigms focused on dynamic behaviour adaptation, based on Function-as-a-Service (FaaS) framework. To allow a rapid interoperability of on-demand services, a solution based on Blockchain is proposed, named BCB-FaaS, to ensure trustiness and accountability of function configuration, providing a strong barrier to well-known cyber-attacks. In addition, a cost analysis has been performed demonstrating how Blockchain can be an economic alternative to secure decentralized communications.

In the modern landscape of smart cities, it is important to guarantee services (such as smart street lights [56], air quality [57], vehicle detection and plate recognition [58]) are always available. Several approaches have been presented to achieve this, but each comes with some limitation or compromise.

In recent years, the computing landscape has evolved to accommodate new requirements which were unpredictable during the personal computing era. In just 20 years, with the objective to increase system response and reduce communication latency, computation moved from mainframes and computing rooms towards Cloud Computing [59], Fog Computing [60] and lastly, Edge Computing [61].

The serverless architecture [62] was born as a technology for Cloud Computing and, in recent years, it became famous for Edge Computing. Serverless computing describes a programming model and architecture where the code is executed in the cloud without any control on the host system resources. This doesn't mean that there are no servers, but the application engineers can forget about the hardware details, focusing on the development of the code, leaving the technical aspects to the cloud service provider. Edge Computing is a distributed architecture created to solve problems that are usually a limitation of Cloud Computing, concerning the evolution of technologies and the development of new types of services. Edge Computing builds an architecture at the edge of the network that is not centralized, as in the case of Cloud Computing, and it is constituted by a cluster of nodes.

With the aim of linking together the above described architectures, in recent years Osmotic Computing has emerged as a computing paradigm focused at dynamically manage heterogeneous resources over Cloud, Edge and IoT resources to enable the Cloud-Edge-IoT continuum [63]. Specifically, it allows to perform a transparent deployment of distributed services over heterogeneous Cloud and Edge nodes, also guaranteeing data proximity to end users and IoT devices and optimizing storage, computation, and network capabilities.

What is important in this revolution is that new computing paradigms are context-aware: when configuring a new IoT device or a node in Edge Computing, the analysis of data collected by sensors is performed locally, without sending every single information to a central server as it happens with Cloud Computing paradigm. This allows to discard the non-relevant information and store, in a centralised manner, only the most significant evidence (e.g. avoid sending data from a humidity sensor if values remain constant, send it only when it moves out of a pre-defined threshold).

Enabling a device to quickly modify its behaviour from a defined task to something different can make a significant difference with respect to a system that is based on Cloud technology only. In the modern city context, it can be useful to load-balance a server if the number of requests increases, or to scale-up a face detection task in a public area in case of congestion at a certain time of the day (e.g. during peak hours when people leave the offices) or at certain conditions (e.g. during a pacific riot at the Town Hall).

To perform such a variety of tasks, the Function as a Service (FaaS) [64] computational paradigm can be adopted. FaaS allows to define several minimal applications and to run one or more instances of these on the same device at the same time.

Edge Computing and serverless model allow the creation of clusters that provide FaaS for the management of IoT devices and data collection and processing, bringing significant benefits to support the technological evolution [65]:

- Workload balance: an application uses the resources of the device in order to function; if a cluster is in proximity, it is possible to balance the workload allowing the execution of functions directly from connected devices via the FaaS framework to execute applications that would not be supported on a single IoT device.
- Latency optimization: to form a cluster, Edge devices are placed in proximity one to the other, hence it is not required to send all data to the Cloud to be routed to an adjacent device. This reduces network congestion, hence achieving low latency, optimizing data exchange.
- Data security: unlike the Cloud, where data travels throughout the network and can be intercepted or altered (e.g. MITM attack), in Edge Computing data remains local and safe from attacks.
- Data analysis: Edge Computing allows to reduce time and resources for data processing and analysis, performing a pre-analysis to identify the target of data collection in the shortest possible time.

Services are supported by a configuration file, containing details such as IP or port configuration and running service parameters. FaaS framework relies on two configuration file approaches: a local configuration file, generally YAML, or a secure remote server.

Both of these comes with limitations: a local file configuration is, by definition, immutable without access to the device. If the service needs to be reconfigured over time, it is required to access the serving device, physically or through a secure connection (eg. Secure SHell (SSH), SSH File Transfer (SFTP) or Secure Copy (SCP) protocols), and modify its configuration. Alternatively, a remote server can send an updated configuration file, but it might be vulnerable to well-known cyberattacks suck as Man-in-the-middle (MITM) [66] or Distributed Denial of Service (DDoS) [67], making it unusable and unreliable.

This research proposes a new approach based on Blockchain, to leverage the flexibility and robustness of Smart Contracts. Specifically, it is proposed the *BlockChain-Based Function-*

as-a-Service (BCB-FaaS) framework that allows combining the well-known FaaS paradigm with the intrinsic features of data non-repudiation and immutability that are typical of Smart Contracts. In simple words, in BCB-FaaS, the service configuration is replaced by a Smart Contract that is in charge of controlling all devices connected to the distributed Blockchain network.

Blockchain has been increasingly recognised as a technology able to address existing information access problems in different applications domains. Born in 2009 as the technology behind Bitcoin [15], it has completely revolutionised traditional encryption-based security systems, introducing a new approach applying hash-based encryption to link blocks of various content. Smart Contract is one of the major applications of Blockchain: it is a protocol aimed at digitally facilitating, verifying, and enforcing the negotiation of an agreement between parties without the need of a centralised certification third party.

The major contributions of this paper are as follows:

- analysis of Serverless paradigms has been performed, with a focus on dynamic behaviour change at the edge, combined with recent approaches aimed at securing communications with the use of Blockchain technology;
- a group of well-known cyber-attacks have been analyzed and their effect compared to highlight the current threats and possible solutions with regards to FaaS protocol;
- the BCB-FaaS framework is proposed to ensure trustiness and accountability of function configuration. A cost analysis has been performed demonstrating how Blockchain, despite the known cost to write transactions, is still the best alternative compared to secure server infrastructure.

3.3.1 Background and Related Work

This section illustrates some of the related works in the context of Cloud and Edge Computing, with particular attention to the capabilities of dynamically changing the computational behaviour at the edge. However, this research work does not consider Edge Computing as an extension of Cloud computations with limited resources. Instead, it aims at ensuring the confidentiality of service configuration when it is required by the context.

3.3.2 Serverless and Edge Computing

Several scientific initiatives have been summarized in a recent survey [26], demonstrating how Edge Computing is the necessary step towards the refinement of Smart Cities.

To achieve such a goal, serverless architecture and FaaS framework are acknowledged to be the right approach [35], and many different solutions have been tested to elect the most performing one [38], but experimental results show that performances are highly comparable.

A common problem of Smart City services in Edge Computing is the low latency and consequently, the drop of the Quality of Service in Cloud scenario [68, 27]: authors designed a new and diverse approach to improve efficiency in collaboration for Smart City services, minimizing response time while optimizing energy consumption, and they achieve it with the proposed Intelligent Offloading Method.

3.3.3 Secure Communication through Blockchain

Blockchain technology can be logically divided into four different layers [69]: data layer, consensus layer, network layer, and application layer. The data layer deals with the cryptographic functions, data structure and the mining process. The consensus layer involves different mechanisms (e.g., PoW, PoS, DPoS). The network layer deals with scalability and data privacy, which is still a potential threat due to the openness of Blockchain. The application layer can be summarized with the development of Smart Contracts.

Many architectural challenges have to be considered in applying Blockchain for the IoT [70] and to ensure security in industrial applications [71]. Despite the challenges, Blockchain technologies are highly promising for resolving security, privacy, and trust issues in multi-stakeholder application environments.

Blockchain has been increasingly recognized as a tool able to address existing open information access issues [72]. In fact, it has been demonstrated how to achieve possible transparency, security, privacy, efficiency and traceability while accessing services using Blockchain technology, and how Smart Contract can be adopted in many contexts. It is in fact becoming common to see Smart Contract applications for industry indicators, such as Service Level Agreement (SLA) [73], to achieve trustiness and transparency in different application domains [1, 74] and can be trusted and secured through several tools [75].

Table 3.2 summarizes the main literature reviews. Differently from the above mentioned recent scientific initiatives, this paper proposes a practical implementation of how Blockchain and Smart Contracts can be adapted to implement a trusted distributed system for serverless

applications of IoT and Smart Devices eliminating the disadvantage of SPoF and the risk of cyber threats.

3.3.4 Motivation

Central server architecture is limited in terms of security because it is a Single Point of Failure (SPoF) and it is susceptible to many cyber attacks. To name a few, Distributed Denial of Service (DDoS) [67] and Man-in-the-middle (MITM) [66] are well-known attacks that origin at the beginning of the IT era, but are still some of the most used and crucial types of attack.

DDOS attack consists of denying access to a service by limiting access to a machine, making the network incapable of providing normal service by targeting either the network bandwidth or its connectivity. Such attack aims at saturating the network capacity to deny access to legitimate consumers by sending to the victim host a huge number of data packets using simple and light-weight tools involving even unaware users to leverage the number of attacking machines to run a distributed version of this attack. Multiple solutions are available to limit such attacks [76], but these have an immediate correlation with the increase of cost to maintain the secure infrastructure [77].

A MITM attack, the malevolent user stays between the two parties of a conversation (being this human-to-human, human-to-machine, or machine-to-machine communication) and establishes independent connections with both the victims. The attacker takes control over the communication channel sending altered messages to the legitimate users, sending altered data or refusing to deliver data to one of the parties, causing the incomplete exchange of messages which can drastically alter the meaning of the communication.

In the case of the FaaS protocol, the attacked service can be the server that manages the service configuration. During a DDOS attack, the malevolent user can gain root permission to the system or database and add or remove data from the system making it difficult to identify what is legitimate and what is altered, making the entire service configuration unusable.

DDOS and MITM are just two examples of possible distributed attacks, but the same principle applies to almost all of them.

From the Information Security point of view, it is essential to guarantee that a system is reliable and its data immutable. There are in fact three aspects, known as the CIA triad (Confidentiality, Integrity and Availability) [78], to consider when designing a secure system:

- **Confidentiality** refers to the inability to access sensitive data by unauthorized personnel

Table 3.2: Summary of the literature review.

Title	Objectives	Advantages	Disadvantages
Towards a Serverless Platform for Edge Computing [35] (2019)	Present a FaaS platform implementation to deploy functionalities on fog nodes with limited resources.	Satisfy the requirements of different application scenarios and heterogeneous deployments of fog nodes.	The overhead introduced is a critical aspect that has not been addressed.
An Evaluation of Open Source Serverless Computing Frameworks Support at the Edge. [38] (2019)	Validate serverless architecture and FaaS frameworks through experimental results to elect the most performing one.	Provides a classification of best approaches to improve QoS for low-latency applications.	A framework comparison is proposed in terms of response time, throughput and success rate, but no technological progress has been made.
Addressing application latency requirements through edge scheduling. [68] (2019)	Design a new and diverse approach to improve efficiency in collaboration for Smart City services, minimizing response time while optimizing energy consumption, and they achieve it with the proposed Intelligent Offloading Method.	The proposed solution improves QoS and responsiveness.	The approach is relevant to a single use case and it is not guaranteed that the positive results are mantained in different application domains.
Intelligent Offloading for Collaborative Smart City Services in Edge Computing. [27] (2020)	Design an intelligent offloading method to improve QoS for Smart City servicese, optimizing energy consumption while guaranteeing the privacy preservation.	Privacy preservation is addressed.	The work proposed is based on offloading simulation, without a real Smart City use case.
Blockchain for the IoT: Opportunities and challenges. [70] (2018)	Describe the opportunities for applications of Blockchain for the IoT and examine the challenges involved in architecting Blockchain-based IoT applications.	A comprehensive overview of Blockchain technology with focus on IoT is presented.	No real use case application is implemented.

Title	Objectives	Advantages	Disadvantages
Towards Blockchain-Enabled Security Technique for Industrial Internet of Things Based Decentralized Applications. [71] (2020)	Develop a novel blockchain-enabled cyber-security framework and algorithm for industrial IoT.	The proposed approach is a potential candidate for the blockchain-driven IIoT system in terms of reliability, convergence, and interoperability.	A comparison between the proposed solution and the state of-the-art security based algorithm is not analyzed, and an objective evaluation can not be performed.
FHIRChain: Applying Blockchain to Securely and Scalably Share Clinical Data. [72] (2018)	Provide a prototype for collaborative clinical decision support dedicated to patients based on blockchain technology.	Highlights how to achieve transparency, security, privacy, efficiency and traceability while accessing services using Blockchain technology, and how Smart Contract can be adopted in many contexts.	The work proposed is based on several assumptions, and the evaluation of the quality of the work can not be fully appreciated.
Smart Contracts for Service-Level Agreements in Edge-to-Cloud Computing. [73] (2020)	Provide a new Smart Contract based architecture for Service Level Agreement evaluation.	Optimize resources and automated deployment of applications in a dynamic and decentralised way.	Consider only Ethereum as Blockchain architecture and it is not applicable for Private/Federated Blockchains.
Blockchain-Based Healthcare Workflows in Federated Hospital Clouds [1] (2020)	Propose a Software as a Service Hospital cloud to establish a federation to form a virtual healthcare team.	Healthcare environment privacy, data authenticity and protection is guaranteed by the Blockchain technology.	Only a subsection of the entire healthcare scenario is considered.
BCB-X3DH: a Blockchain Based Improved Version of the Extended Triple Diffie-Hellman Protocol [74] (2020)	Propose a new protocol to eliminate Single Point of Failure for Triple D-H key exchange algorithm.	Combine X3DH security mechanisms with the intrinsic features of data non-repudiation and immutability of Smart Contracts.	The implementation approach proposed can not be applied on resource constrained IoT devices.

or entities. If an attacker had access to the Blockchain, he would still not be able to view or extract information. All information relating to each transaction is in fact encrypted. Only those in possession of the corresponding private key can access it.

- **Integrity** refers to the inability to modify or destroy information by unauthorized users. The incorruptibility of the data refers to the fact that once a transaction has been validated and inserted in a block, it can no longer be cancelled or tampered with. The incorruptibility of the network is instead linked to the fact that the majority of nodes in the Blockchain must recognize each transaction as valid. With more than 2 thousands active nodes currently in the Ethereum Blockchain, the 51% attack [16] becomes practically unfeasible.

- **Availability** refers to being able to access information in a timely and reliable manner. A Blockchain cannot be compromised starting from a single node (no SPoF). Each node contains a copy of the Blockchain itself so the system would continue to function normally even if one of its components went offline for a period of time. This mechanism makes the Blockchain resistant to DDoS Attacks.

The reasons described in this section suggest that Blockchain can be a valid reliable and secure alternative to the traditional client-server architecture, offering intrinsic security features that would be expensive to achieve and maintain with a traditional implementation.

3.3.5 System Design

In this work, it is proposed an improved FaaS framework aimed at taking the advantage of inherit features of security and trustiness of a Blockchain network: the *BlockChain-Based Function-as-a-Service (BCB-FaaS)*.

The reference architecture diagram for Edge devices and FaaS protocol is shown in Figure 3.21. A general-purpose Edge device can be equipped with several sensors or peripheral hardware to provide different services, such as temperature, humidity or air quality sensors, high-resolution camera or audio capture systems. Several configurable functionalities (implemented according to the FaaS framework) can be activated on-demand and can involve remote picture capture, vehicle traffic monitoring, moving object detection, resource system management, and many more. In particular, services are deployed using containers to benefit from virtualization technology to simplify installation, configuration and maintenance, allowing service portability and resiliency. The FaaS approach is adopted for an easy on-the-fly

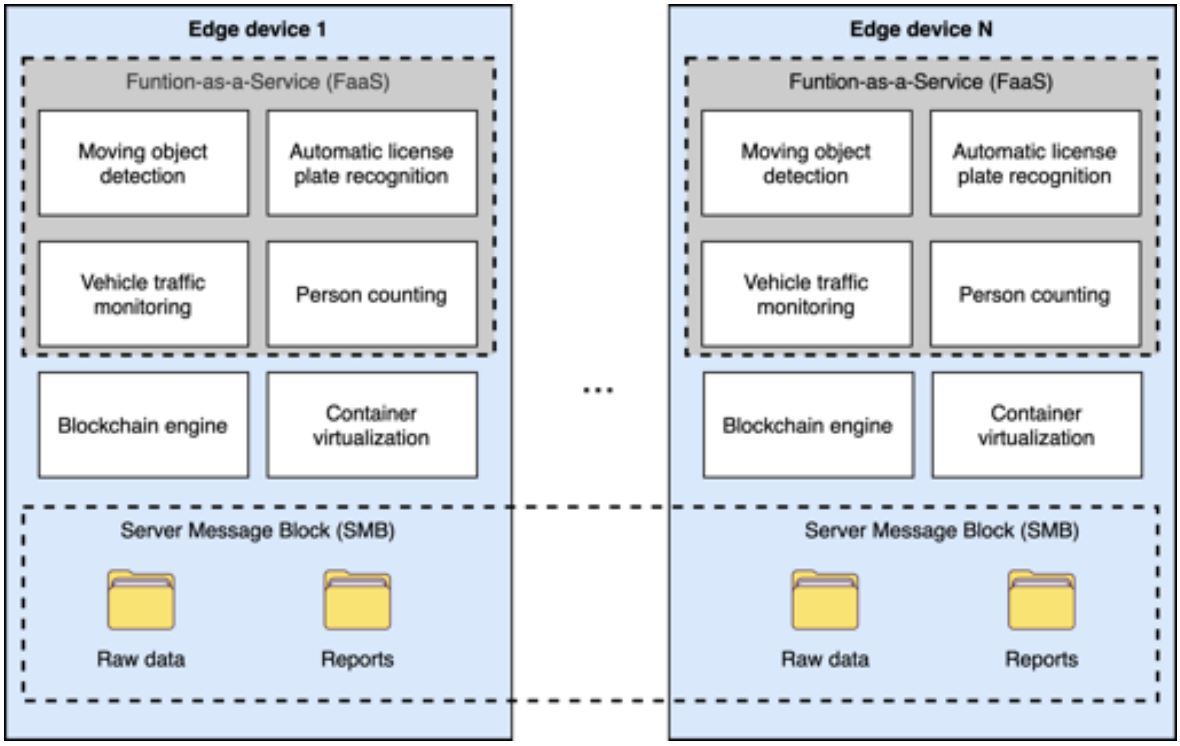


Figure 3.21: Architecture diagram for Edge devices and FaaS protocol.

device configuration, especially if switching among different services is required during the time.

The system aims to guarantee that all updates to service configuration are trackable and irreversible. To achieve such a goal, the Blockchain engine is responsible to store information on the Blockchain to certify data non-repudiation and immutability, guaranteeing accountability and authenticity, through the development and coding of a Smart Contract.

Raw data captured from devices and any report produced are accessible remotely using a Server Message Block (SMB) protocol. This configuration can be replicated for any number of Edge devices available, for example, to cover the city centre area. To secure the communication between the Edge device and the endpoint, a Virtual Private Network (VPN) can be configured.

The application of this analysis can vary from a scenario where it is required to monitor the number of vehicles in a given time slot or to verify if a vehicle enters a restricted or forbidden transit area or it is moving in the wrong traffic direction. Such scenarios can be configured with a FaaS approach to permit an easy switch of configuration during different hours of the day (e.g. road traffic forbidden in an area during night hours) or depending on special requirements (e.g. road traffic changes during a riot).

In the BCB-FaaS system, it is fundamental to guarantee the trustiness of the server on

which resides the service configuration. In fact, if an attacker takes control of the server, he/she can alter the service configuration or can arbitrarily refuse to accept any request or update.

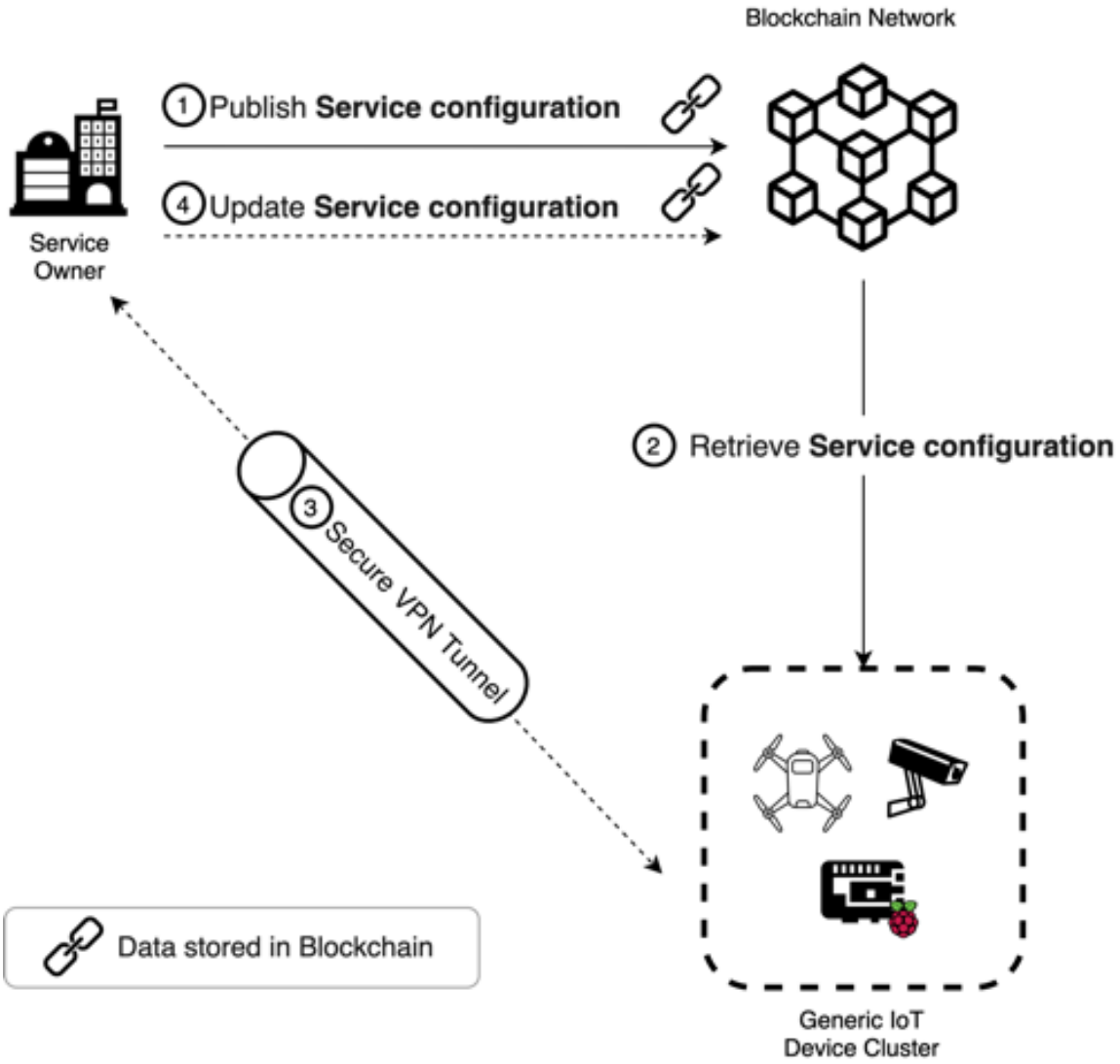


Figure 3.22: BCB-FaaS scheme.

The system described in Figure 3.22 highlights the following phases:

1. **Publish Service configuration:** The Service Owner publishes the Service configuration to the Blockchain network, containing all the attributes to run a service as it usually happens with the content of a .yaml file.

2. **Retrieve Service configuration:** Each IoT Device can be configured to request the Service configuration at any time interval to ensure to always have the latest available configuration. After downloading it, the FaaS framework continues its business as usual.

3. **Secure communication through VPN Tunnel:** If any data has to be shared between IoT Devices and Service Owner, a communication channel can be secured through VPN Tunnel for E2E encryption.

4. **Update Service configuration:** At any time, the Service Owner may upload new Service configuration detail to modify the desired behaviours.

The BCB-FaaS protocol can be seamlessly adopted in device-to-device communication scenarios including IoT where low-resource smart devices (e.g. drones, Raspberry Pi, cars, CCTV cameras, etc.) can act as light Blockchain nodes without significantly altering their system configurations.

The implementation details of the proposed architecture using a public Blockchain network is described in the next section.

3.3.6 Implementation

In this section, a prototype implementation of the BCB-FaaS framework is discussed. Ethereum [79] was chosen as the public Blockchain network since it is currently the most widely used Blockchain platform thanks to its flexibility in developing Distributed Apps (DApps) through Smart Contracts. Moreover, this has been chosen against private Blockchain implementation such as Hyperledger Fabric [80] or Sawtooth [81] since the greater number of nodes on the public Blockchain network improves the trustiness of the system.

All the components are configured as Docker containers to take advantages of virtualization technology allowing service portability and resiliency.

Docker

Following the Osmotic paradigm, the functionalities are deployed in distributed environments as a graph of MicroELementS (MELS) [82], including MicroServices for implementing specific functionalities, which can be deployed and migrated across the virtualized infrastructures, and MicroData to represent the information flows to and from IoT devices. Thanks to their flexible nature, the MELS can be deployed as lightweight containers such as Docker seamlessly on Cloud and Edge architectures.

Docker is a platform that allows to develop, share and run any application anywhere. Its expansion took place in a very short time, becoming one of the most widely used standards for software distribution. Before Docker, the development pipeline required the use of Virtual Machines (VM) most of the times. VM need their own operating system and dedicated

resources (such as CPU, RAM etc) and are usually slow to boot, and migrating a service or an entire VM from a Cloud platform to another requires a significant effort. With the advent of containerized platforms, the management effort considerably decreased, as each container does not require a complete operating system as all the containers run on a single host shared the same operating system, efficiently managing the system resources.

OpenFaaS

OpenFaaS is an application developed to facilitate the implementation of functions and microservices triggered by events. It works by encapsulating code or an already existing binary file in a Docker image to obtain a highly scalable endpoint, automatically managing system resources. The goal of OpenFaaS is to develop solutions in any programming language that apply the FaaS paradigm in any environment that has the necessary resources to run the platform, not necessarily having to approach Cloud-computing.

OpenFaaS is template based: it is possible to create functions choosing the language and the platform for the deployment, write the code and build the function. The OpenFaaS cluster has all the fundamental elements to be a service provider with a high fault-tolerance.

Through OpenFaaS it is possible to develop functions with small portions of code to be packaged in a Docker image to be implemented through a .yaml configuration file, using the Command Line Interface (CLI) to build an image and make it available for deployment. A sample .yaml file indicates the OpenFaaS gateway, the name of the function, the name of the template, the path of the handler, the name of the image and the maximum read and write timeouts before the function closes the service:

Ethereum Blockchain Smart Contract

In this paper, the implementation focuses on a web-based Application Program Interface (API) such as Infura [83] to interact with Ethereum to be more efficient in terms of resource because the only requirement for the IoT device is to be able to perform Secure HTTP requests, and this is feasible also for low-resource devices. The Smart Contract is written in Solidity programming language, and it is the core of the proposed implementation: it is designed to store the Service Owner ID (SOID), eventually the Unique device Identifier (UID) or a MAC address, and the Service configuration, in the form of free text.

The main $Configuration$ data structure of the Smart Contract is shown in Table 3.3.

The computational complexity of the provided Smart Contract can be calculated in terms

Table 3.3: Data structure of the implemented Smart Contract.

Attribute	Data Type	Description
SOID	string	Service Owner's identification code
UID	string	Unique device Identifier or device MAC address
Service configuration	string	OpenFaaS service configuration

of the cost for the transaction to be executed and mined to be stored in the Blockchain.

Generally, a simpler Smart Contract costs less in terms of the transaction fee, but multiple things have to be considered in designing an efficient Smart Contract [84]:

- **Data structure**: the Data structure used to store the attributes of the contract must use a proper data type, because waste in memory space has a direct result in the transaction cost.
- **Loops**: carefully consider when it is required for the contract to iterate through all the elements stored. This is one of the most expensive operations in a Smart Contract.
- **Deadlock**: avoid all entities on the contract are waiting for some status to change.
- **Mutual exclusion**: the contract must ensure that some actions cannot be executed simultaneously by multiple users or invocations.
- **Gas limit**: when deploying a Smart Contract it is required to provide a Gas limit value as the upper bound for a transaction calculation, in order to avoid infinite loops and miner's deadlock state.

The Smart Contract proposed in this paper has an average execution cost of 93.732 Gas, which is in line with the cost per transaction of the entire Ethereum network [85].

In order to publish the Service configuration in the Ethereum Blockchain network, the Service Owner needs to call an *external public function*.

Similarly, any IoT Device who requires to download the Service configuration from the Blockchain network only requires to know the Service Owner ID, which can be a simple alphanumeric string. Eventually, for large scale organizations that require to manage several IoT devices, it is possible to query a device-specific configuration using the device UID or the device MAC address.

It is important to note that the Service Owner pays the Blockchain transactions only when it is required to store or update data (i.e. storing Service Configuration), and do not pay any fee for data retrieval.

3.3.7 Experiments

The objective of experiments discussed in this section is to verify if the overhead introduced by Blockchain in this BCB-FaaS framework prototype implementation is acceptable in terms of both read and write access times, CPU usage and inbound and outbound network traffic, compared to a system based on a third-party configuration server. Specifically, we considered the following implementations:

- **Traditional FaaS approach**: A sample service configuration, in the form of a .yaml file, is prepared to allow a generic IoT device to perform a task and it is sent to a remote server. The IoT device requests this information to the server and performs the required tasks.

- **BCB-FaaS**: The service configuration is generated in the same manner as the first approach along with SOID and UID, and it is sent to a Blockchain node to be added in the queue for mining and immutable storage. Time-to-mine and Ethereum (ETH) cryptocoin currency transaction cost are subject to Ethereum network traffic: the more time is required to mine, the higher is the cost to be paid for a transaction. The IoT device queries a Blockchain node to retrieve this information and performs the required tasks.

The system prototype assessment was conducted analyzing the total execution time required to complete the described approaches, the CPU usage and the inbound and outbound network traffic. The IoT device used is a RaspberryPi model 4 equipped with a Quad-core Cortex-A72 (ARM v8) 64-bit SoC @ 1.5GHz and 4 GB RAM running Raspbian OS. The remote server is a Intel® Xeon® E3-12xx v2 @ 2.7GHz, 4 core CPU, 8 GB RAM running Ubuntu Server 18.04. Tests have been executed for 1000, 2000 and 3000 read and write operations, considering 95% confidence intervals and the average values. All tests have been performed using Ropsten Ethereum public Blockchain test network, leveraging 300+ available nodes with a real server load status. It must be considered that the Ethereum Blockchain Ropsten environment is based on Proof of Work (PoW) consensus protocol which makes it difficult to obtain scalability and system speed. A summary of testbed characteristics is reported in Table 3.4, experiments setups is reported in Table 3.5.

Figure 3.23 shows the execution time difference expressed in seconds between the two system implementations to publish and retrieve service configuration. On the x-axis, it is reported the different two steps required to perform the publish and retrieve workflow,

Table 3.4: Testbed characteristics.

Parameter	Server	Raspberry
CPU	Intel® Xeon® E3-12xx v2 @ 2.7GHz, 4 core CPU	Quad core Cortex-A72 (ARM v8) 64-bit SoC @ 1.5GHz
RAM	8GB	4GB
OS	Ubuntu Server 18.04	Raspbian

Table 3.5: Summary of experiments performed.

Parameter	Value
Test executed for each scenario	[1000, 2000, 3000]
Confidence interval	95%
Gas used by transaction	93,732
Gas price (Gwei)	45
Average cost per transaction (ETH)	0.004218

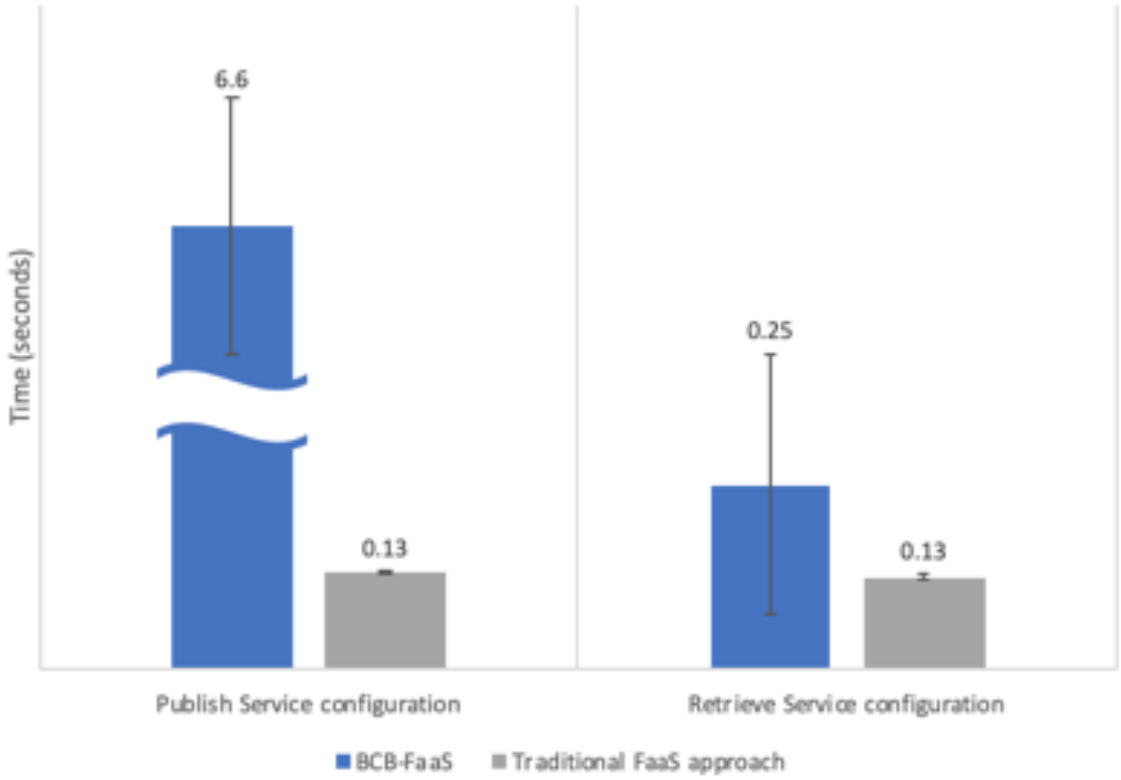


Figure 3.23: Execution time comparison between BCB-FaaS and traditional Faas approach implementations to publish and retrieve service configuration.

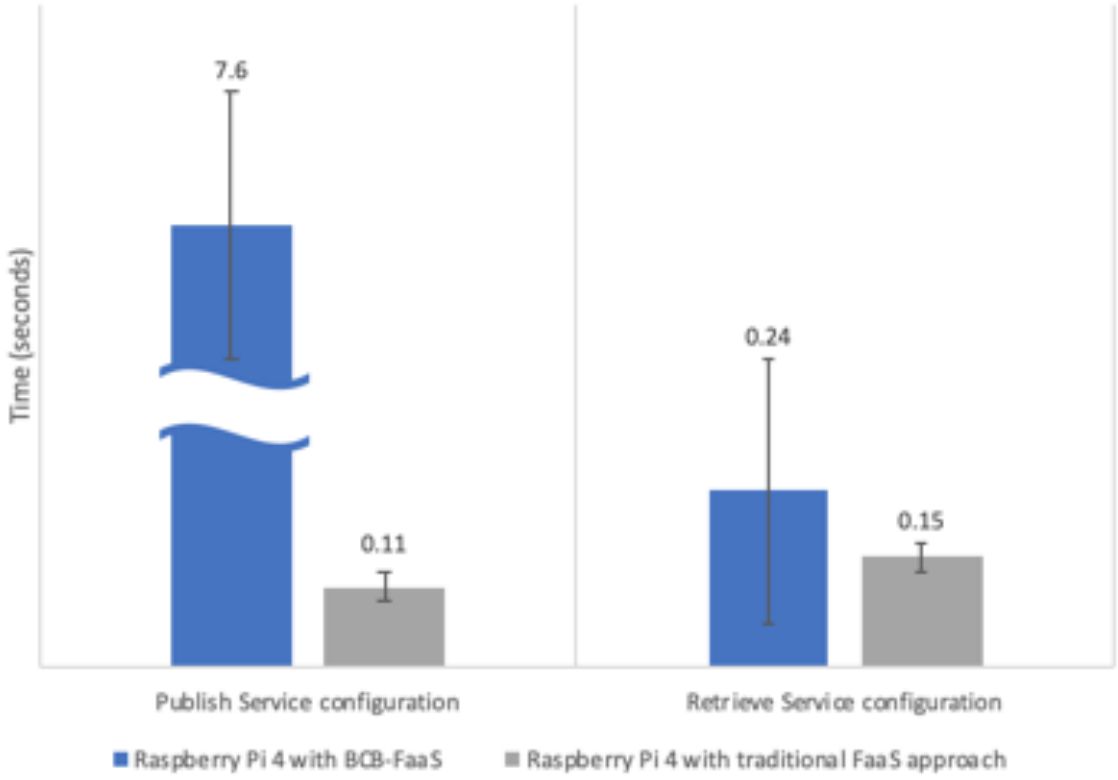


Figure 3.24: Execution time comparison on a Raspberry Pi mod 4 between BCB-FaaS and traditional Faas approach implementations to publish and retrieve service configuration.

whereas on the y-axis it is reported the processing time expressed in seconds. From the graph, it is possible to observe that the time required to publish and update the service configuration heavily deviates in the two implementations, changing from 0.13 seconds up to 6 seconds. This is due to the mining time required to store the data in the Blockchain transaction. The time necessary to retrieve the service configuration presents a similar behaviour as compared to the first step. Downloading data moves from 0.13 seconds for the traditional FaaS approach to 0.2 seconds to retrieve data from Blockchain.

Figure 3.24 shows the results of the same test performed on a Raspberry Pi mod 4, to demonstrate how an IoT device can transparently adopt the proposed framework. From the graph, it is possible to observe that the time required to publish and update the service configuration is considerably different in the two implementations, changing from 0.11 seconds up to 7 seconds, again, due to the mining time required to store the data in the Blockchain transaction. The time required to retrieve the service configuration is similar to the first step, moving from 0.15 seconds for the traditional FaaS approach to 0.2 seconds to retrieve data from Blockchain.

Figure 3.25 shows the CPU usage % difference between the two system implementations

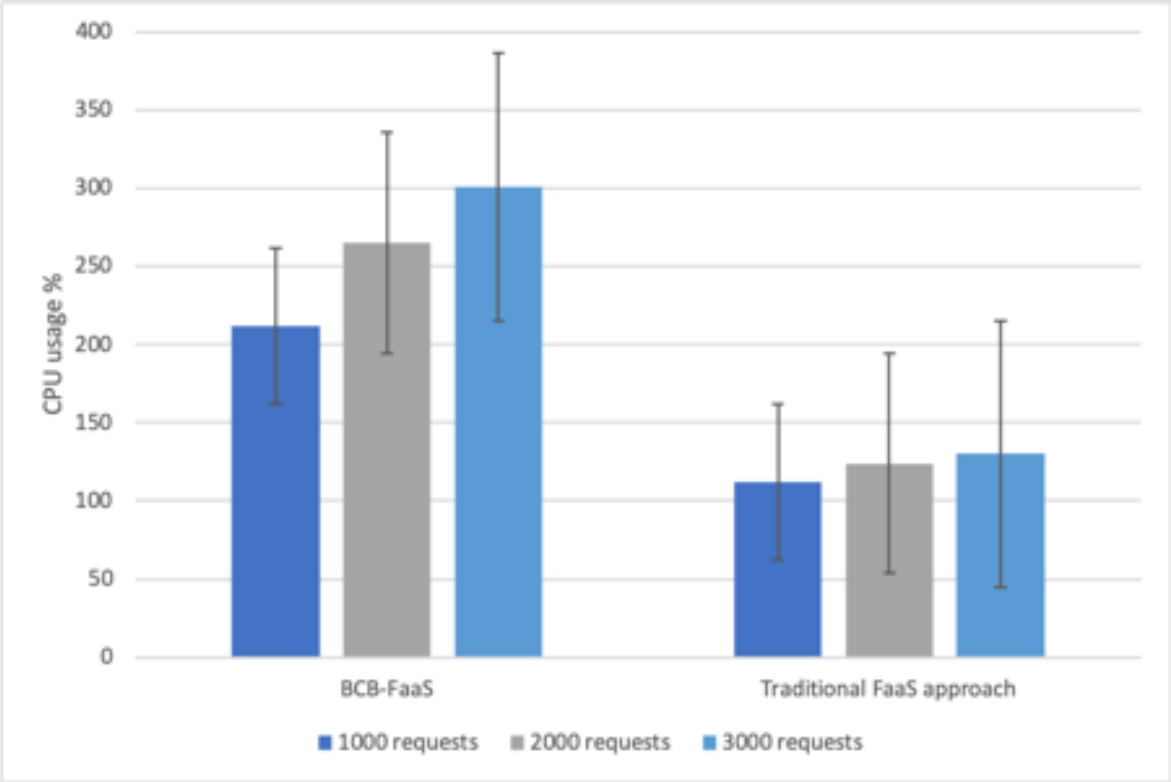


Figure 3.25: CPU usage % comparison between BCB-FaaS and traditional Faas approach implementations to publish service configuration.

to publish the service configuration. On the x-axis, it is reported the differences to perform 1000, 2000 and 3000 requests, whereas on the y-axis it is reported the CPU usage %. From the graph, it is possible to observe that the CPU usage required to publish the service configuration is double in comparison with the Traditional FaaS implementation, but it does not saturate the available system resource. On the other hand, it can be noted in Figure 3.26 how the CPU usage % of the Raspberry Pi mod 4 for the same requests is less for the BCB-FaaS as compared to the traditional FaaS approach. This is due to the fact that more time is required to complete the requests for the BCB-FaaS approach, hence the system is less overloaded, thus resulting in efficient energy consumption and longer duration in case of battery-powered.

Figure 3.27 shows the Inbound network difference between the two system implementations to publish the service configuration. On the x-axis, it is reported the differences to perform 1000, 2000 and 3000 requests, whereas on the y-axis it is reported the Inbound network consumption expressed in KB/s. From the graph, it is possible to observe that the two implementations are comparable with similar values shown. The same can be observed in Figure 3.28 for the Inbound network consumption on the Raspberry Pi mod 4, indicating that there is no significant difference for the IoT device.

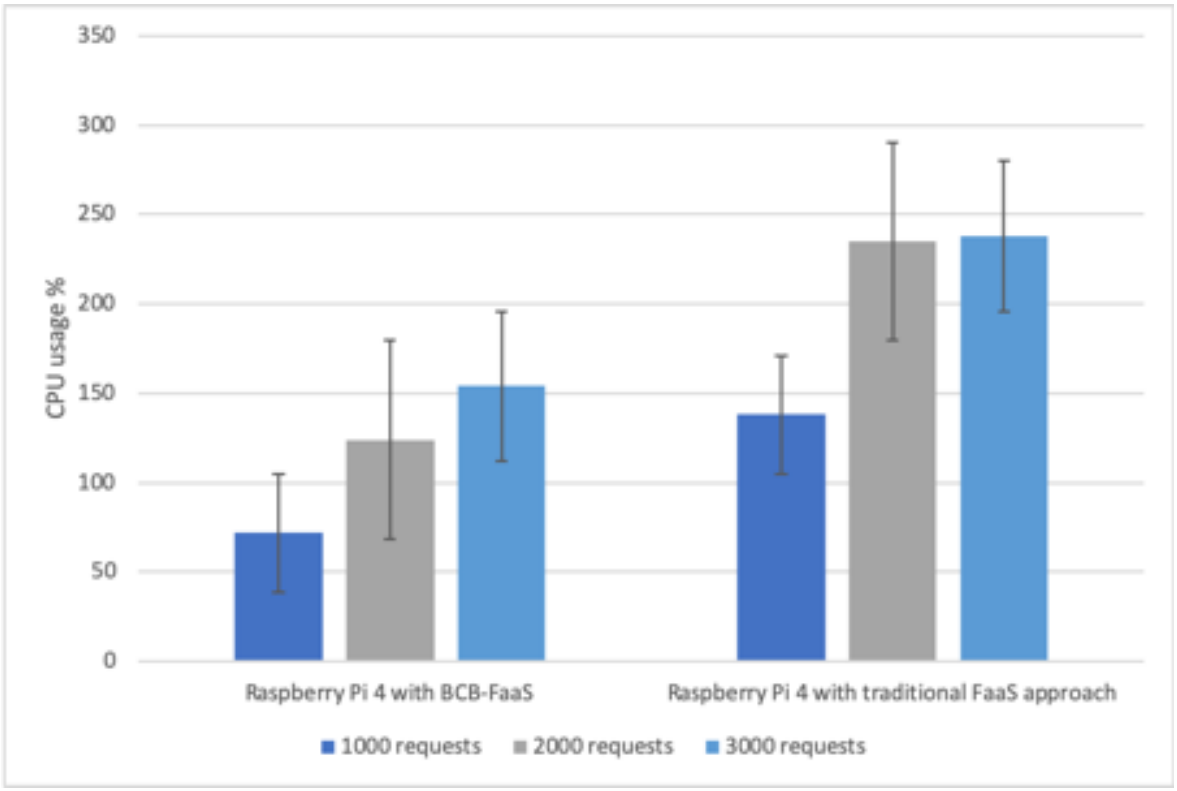


Figure 3.26: CPU usage % comparison on a Raspberry Pi mod 4 between BCB-FaaS and traditional Faas approach implementations to publish service configuration.

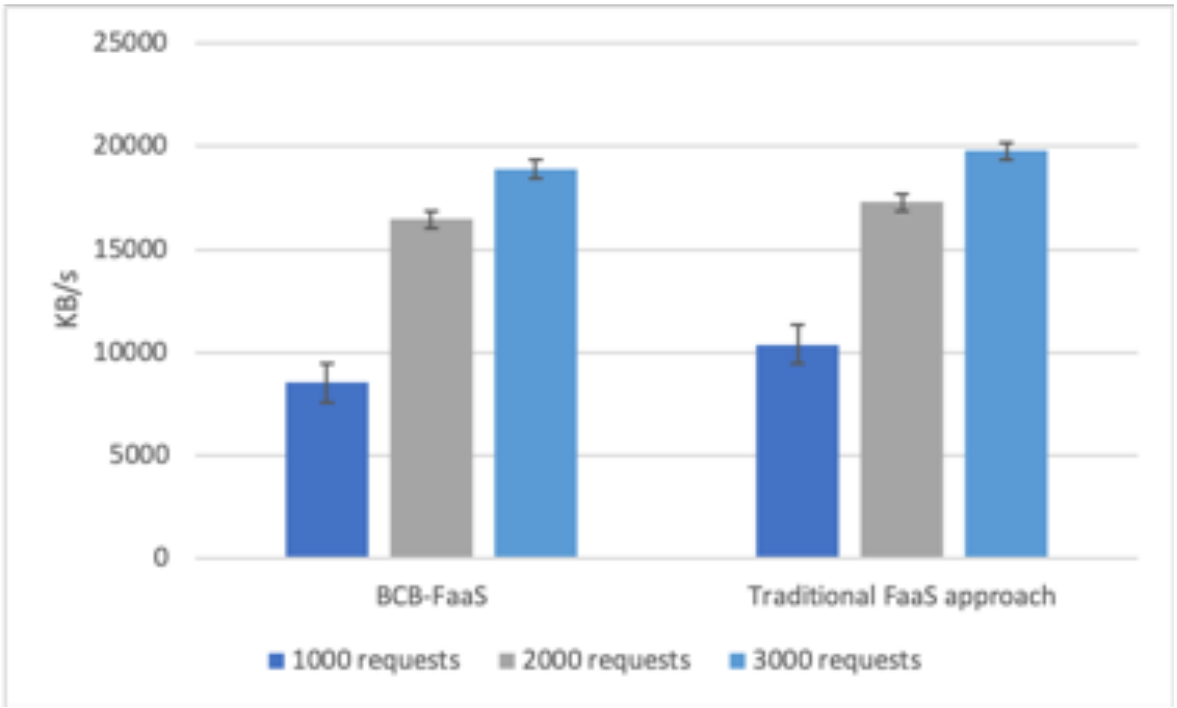


Figure 3.27: Inbound network comparison between BCB-FaaS and traditional Faas approach implementations to publish service configuration.

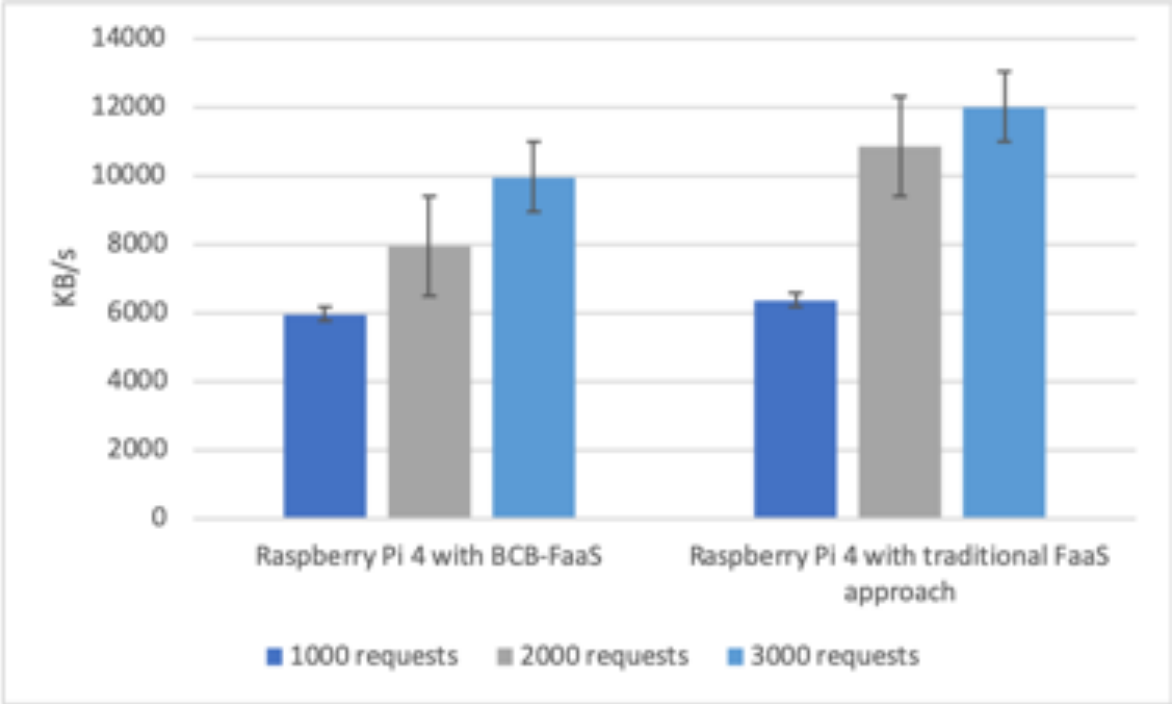


Figure 3.28: Inbound network comparison on a Raspberry Pi mod 4 between BCB-FaaS and traditional Faas approach implementations to publish service configuration.

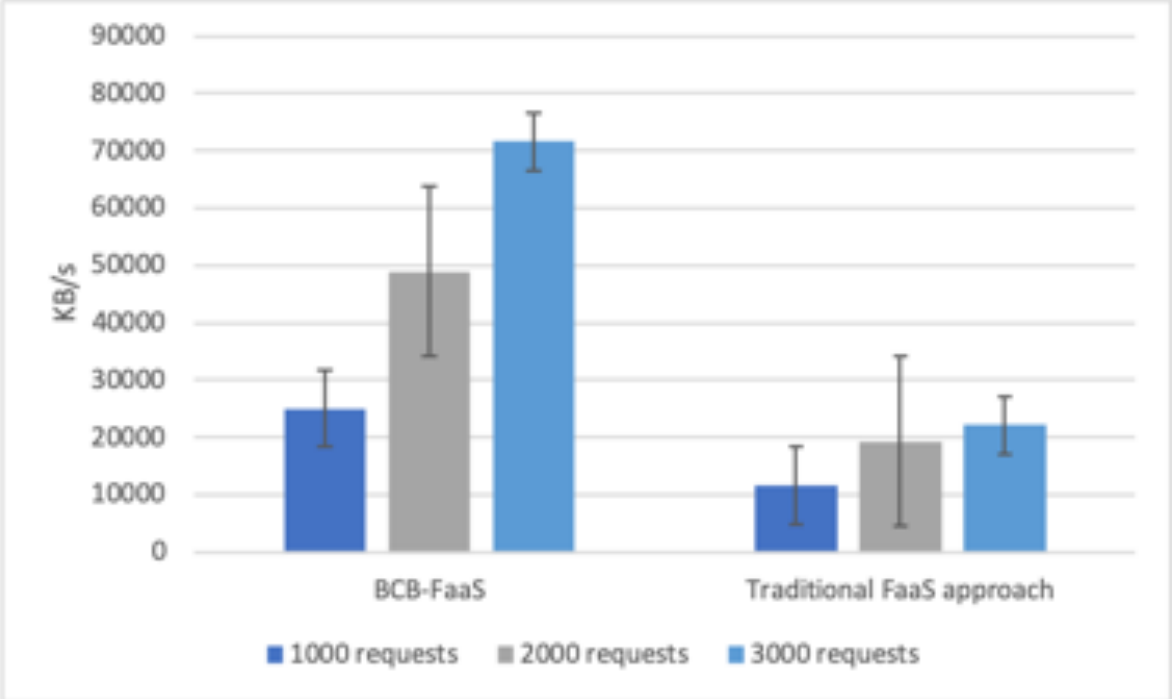


Figure 3.29: Outbound network comparison between BCB-FaaS and traditional Faas approach implementations to publish service configuration.

Figure 3.29 shows the Outbound network difference between the two system implementations to publish the service configuration. On the x-axis, it is reported the differences to perform 1000, 2000 and 3000 requests, whereas on the y-axis it is reported the Outbound

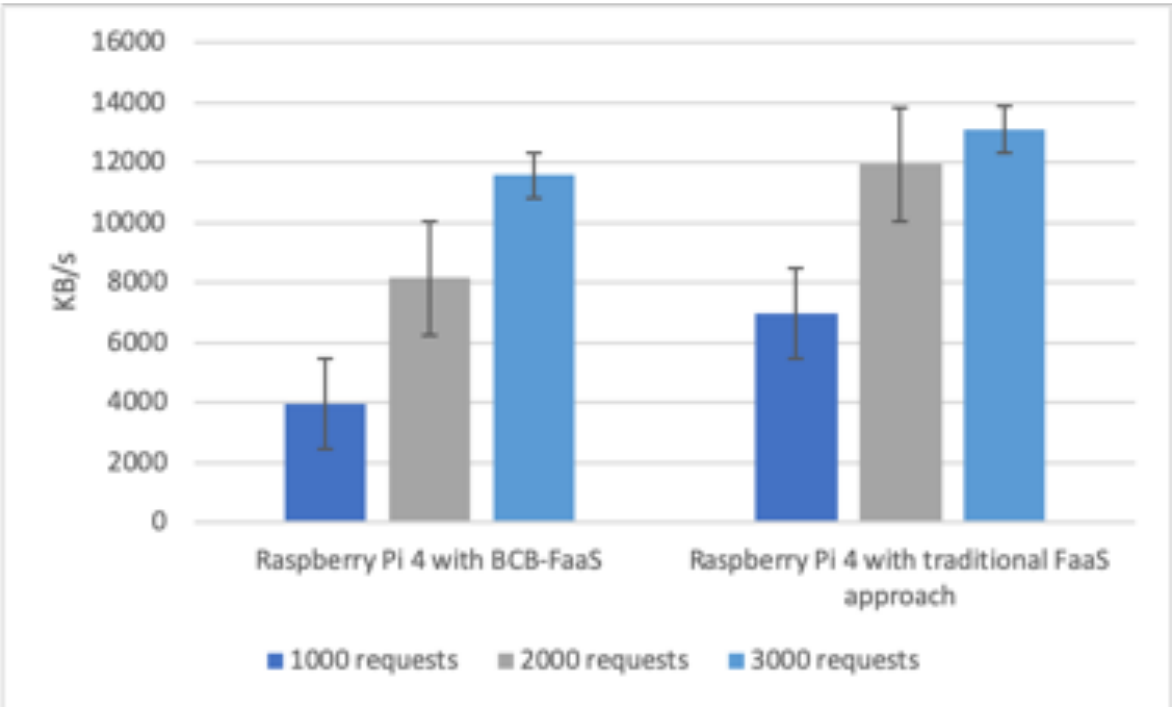


Figure 3.30: Outbound network comparison on a Raspberry Pi mod 4 between BCB-FaaS and traditional Faas approach implementations to publish service configuration.

network consumption expressed in KB/s. From the graph, it can be observed that the BCB-FaaS implementation has a higher Outbound network consumption as compared to the Traditional FaaS implementation. This is expected because the server has to communicate with the Blockchain node to publish the new service configuration.

The Outbound network consumption in Figure 3.30 describes the behaviour on the Raspberry Pi mod 4, indicating that there is no significant difference for the IoT device.

3.3.8 Some remarks

Any protocol based on a central server architecture is limited in terms of security because it is a Single Point of Failure (SPoF) and it is susceptible to many cyber attacks such as DDoS and MITM. For FaaS protocol, the attacked service can be the server that manages the service configuration, altering or destroying information, making the entire service configuration unusable. To address such a limit, several solutions have been proposed so far that are often cost expensive and difficult to be maintained.

This Section describes an improved FaaS framework aimed at taking the advantage of inherit features of security and trustiness of a Blockchain network: the *BlockChain-Based Function-as-a-Service (BCB-FaaS)*.

The BCB-FaaS framework, thanks to the flexibility of the Blockchain technology, can

be implemented according to different strategies to establish human-to-human, human-to-machine and machine-to-machine secure communications in both Cloud/Edge Computing and IoT scenarios. In particular, Ethereum public Blockchain network is used to consider a large number of nodes (a greater number of nodes means a greater degree of security) but, in enterprise environments, private Blockchain technologies such as Hyperledger Fabric can be also adopted if the number of involved nodes is reasonably large to resist to a distributed attack.

Experimental results highlighted that the overhead introduced by Blockchain in the BCB-FaaS implementation in terms of execution time is acceptable, considering the obvious security and robustness advantages obtained by the Blockchain technology. Furthermore, the solution has been validated with a Raspberry Pi mod 4 to analyze the CPU usage and the Inbound and Outbound network. From the results obtained, the proposed BCB-FaaS approach is efficient in terms of energy consumption, resulting in a longer duration for the battery-powered IoT device.

In future developments, the BCB-FaaS framework will be tested in private Blockchain networks such as Hyperledger Fabric and Sawtooth, to find the best trade-off between data privacy and system efficiency.

3.4 Overall consideration

This Chapter focuses on the urban mobility application of IoT devices in Smart City contexts.

Specifically, Section 3.1 compares the results obtained with a closed source traffic camera and a Raspberry Pi equipped with an entry level camera and a machine learning algorithm built for this purpose. Evaluation performed of a full day of traffic demonstrates that the open source solution offers significant improvements in counting the vehicle in a congested road and correctly identifying the license plates even in low light conditions.

Section 3.2 put the basis to develop further protocols based on Artificial Intelligence techniques to validate the self-configuration property. The final goal is to unify the mesh networks discussed, including devices of a different kind, such as MPUs and MCUs, in just one. Moreover, different topology networks (star, ring, Von Neumann, etc.) could identify the best social structure the mesh network should take on to minimize the routing time.

Finally, Section 3.3 describes how the BCB-FaaS framework, thanks to the flexibility of the Blockchain technology, can be implemented according to different strategies to establish

human-to-human, human-to-machine and machine-to-machine secure communications in both Cloud/Edge Computing and IoT scenarios.

CHAPTER 4

Increase security in public connections through the Blockchain

The recent advancements in miniaturized smart data collecting devices pushed the need of securing communications between people and devices. Traditional approaches based on key exchange protocol can't be performed by resource-constrained embedded devices, and a novel approach, based on a robust decentralization of the eXtended Triple Diffie-Hellman (X3DH) protocol, has been proposed, namely the BlockChain-Based X3DH (BCB-X3DH) protocol. This Chapter deals with new technologies applied in the Smart City scenario with Edge and IoT nodes, performing intensive analysis on Raspberry Pi 3 model B+ and Raspberry Pi 4 to validate that the new protocol is not only resistant from well-known distributed attacks but can also be executed by miniaturized hardware with benefits in terms of resources, energy consumption and battery life-cycle.

4.1 Securing public connections through Blockchain

Nowadays, the increasing complexity of digital applications for social and business activities has required more and more advanced mechanisms to prove the identity of subjects like those based on the Two-Factor Authentication (2FA). Such an approach improves the typical authentication paradigm but it has still some weaknesses. Specifically, it has to deal with the disadvantages of a centralized architecture causing several security threats like denial of service (DoS) and man-in-the-middle (MITM). In fact, an attacker who succeeds in violating the central authentication server could be able to impersonate an authorized user or

block the whole service. This work advances the state of art of 2FA solutions by proposing a decentralized Microservices and Blockchain Based One Time Password (MBB-OTP) protocol for security-enhanced authentication able to mitigate the aforementioned threats and to fit different application scenarios. Experiments prove the goodness of our MBB-OTP protocol considering both private and public Blockchain configurations.

Nowadays, we live in an era where the digital identity (DI) plays a fundamental role to securely access various kinds of services in our social and working life also enabling non-repudiation [86]. A consolidated approach to manage the DI legitimating applicaiton user's access is the Two-Factor Authentication (2FA) [87]. It requires two distinct forms of identification to authenticate the end user requesting to access an application and it typically involves the generation of a One-Time Password (OTP) in addition to the end user's credentials usually used to access the system. However, the OTP generation is a thorny task that could compromise the security of the whole system. One of the most notorious examples of OTP generation algorithm is the Time-based OTP (TOTP) [88], which is based on the Timestamp value of the request and on a shared secret that is generally stored in a centralized server managed by a trusted authority. The latter is limited in terms of security, as it can be vulnerable, among others, to man-in-the-middle (MITM) and denial of service (DoS) attacks. In fact, an attacker who succeeds in violating the central server could be able to generate an OTP and impersonate an authenticated user.

The main purpose of this research is to approach the generation and distribution of OTPs in a decentralized fashion to reduce the dependence from any Single Point of Failure (SPoF) and make the DI management system more robust against possible attacks. Furthermore, the advantages offered by the Blockchain technology to develop a distributed 2FA protocol is considered. Indeed, Blockchain is a consolidated solution for ensuring the integrity of data in a distributed network. In our view, Blockchain can be successfully adopted to replace the traditional central trusted authority, that is legally responsible for the OTP distribution, with a cooperative trusted environment that implements the same functionalities.

Following the above key concepts, it is proposed a decentralized Microservices and Blockchain based One Time Password (MBB-OTP) protocol for security enhanced authentication. In particular, it is presented a decentralized 2FA system that generates OTPs manipulating different pieces of information (i.e., user identity, request timestamp, and a shared secret). By adopting the microservices architectural concept each microservice hosts only a portion of the "secret" and execute sub-tasks of the whole OTP generation process.

The advantage of MBB-OTP is twofold: i) it is not exposed to MITM attacks: by splitting

the 2FA service into different and independent microservices, in case of violation of one of them, the attacker would get totally decontextualized information without affecting the end-user security; ii) it can potentially mitigate DoS attacks using the microservices architecture principles: a compromised microservice can be easily replaced by a non-corrupted one.

The major contributions of this research are the following:

- an analysis of existing OTP solutions is presented;
- the vulnerabilities of the classic authentication and OTP generation systems is discuss;
- the MBB-OTP protocol is proposed, that is designed as a decentralized solution that makes use of Blockchain.
- a preliminary assessment of the MBB-OPT protocol is provided, analyzing the impact of Blockchain and highlighting the required trade-off between security, efficiency and costs.

4.1.1 Related Work

In recent years, the evolution of digital services has brought to the development of more and more advanced mechanisms to prove the identity of subjects. Currently, there is no one-fit-for-all authentication protocol and its choice depends on the requirements of each specific application. This section highlights the state of the art, limitations and drawbacks of the most used approaches.

An important research that discovers all main solutions for OTP management including a secure web authentication with mobile phones, a piece of framework for securing sensitive input and a user authentication protocol resistant to password stealing and password reuse attack is described in [89].

Some researches have proven some weaknesses of 2FA centralized management, such as MITM, DoS and consequently masquerade attacks. Thus, several works have tried to improve the security in the 2FA implementations. Mainly, they tried to solve the main weakness present in the generation and transmission of the OTP.

To achieve a faster and secure OTP generation for Internet of Things (IoT) devices (including smartphones) authentication, different cryptographic algorithms have been proposed. In particular, AES cryptography has been adopted to encrypt the seed [90], formed by username and timestamp, and Elliptic Curve Cryptography and the isogeny property have been analyzed [91] to improve the generation of OTP and IoT device authentication avoiding the

use of the same keys for different communications. However, performance evaluations are not considered so the value of the research can not be fully appreciated.

The evolution of 2FA is the Multi-Factor Authentication (MFA) in which researchers enriched the typical 2FA with an external IoT device to complete the process [92]. Although this approach is valuable, because the transmission of double OTP is carried out through two different protocols and different technologies increasing security, it is unpractical in many cases as each user has to bring an external IoT device to complete the authentication. For the reasons above described, some researchers have focused their attention on the abstraction of the OTP lifecycle in order to provide multi-tenant OTP Cloud services with the same architecture. For example, in [93] a One-Time Password-as-a-Service (OTPaaS) solution is proposed. Although the idea is innovative, it is not verified how security is improved.

In [94] a 2FA violation is described. In particular, a network violation is analysed, highlighting how the 2FA violation is possible as a result of the hijacking of software installed on the client machine for OTP generation. This would not be possible with our implementation because the request is managed in a decentralized manner, protected by an Identity Provider. As mentioned above, the request of a new OTP is an authenticated action and, thanks to the distributed nature of the architecture, it is generated on the server in response to a user autentication request. The weakness of the transmission of OTP through Short Message Service (SMS) has also been analyzed and it was proved to be susceptible to MITM attacks. In this regard, solutions to reduce the risk of MITM attack through a micro-services architecture is discussed in [95] and [96].

This approach helps to drastically improve the security of 2FA protocols, and the research progresses further by implementing a decentralized architecture, the micro-services paradigm and Blockchain technology.

Several solutions have been proposed in recent years to integrate the intrinsic features of Blockchain of decentralization, data non-repudiability and certification along with 2FA, but all comes with some limitation or compromise.

A 2FA-based system leveraging Ethereum to validate the user according to the possession of a Private Key, whilst the Public part is stored on the server, has been analyzed in [97]. This solution can be impractical in most the scenarios when a user wants to access a system from a mobile device and he/she has to carry the Private key in an insecure manner. Moreover, this solution does not involve the actual generation of the OTP in the Blockchain, but it simply stores a secret key. A 2FA solution implemented using the Message Queuing Telemetry Transport (MQTT) protocol and Blockchain is discussed in [98]. In particular, the OTP is

verified thanks to the Ethereum Blockchain and the Public key. However, such a solution is valuable, it is pretty theoretical and it is not verified experimentally. Furthermore, it implies that each user is bound with an Ethereum address and involved in almost a transaction. Another implementation of the 2FA implementing Blockchain technology is proposed [99]. Here, the user sends a key as a transaction to the server using a public Blockchain such as Ethereum, whereas, the server, having verified the key, generates an OTP encrypted with the user's public key. Only the user will be able to decrypt the message to obtain the OTP and can access the system. Although such a solution increases security in OTP generation, it forces each user to interact directly with the Blockchain. In [100], authors proposed OTP generation mechanisms without a trusted entity with the use of private Blockchain. This approach is interesting but the Blockchain utilization is limited to the OTP verification and it is not involved in OTP generation.

Differently from the aforementioned solutions, our MBB-OTP protocol is user-friendly because it is transparent to the end user which has all the technical details abstracted from the application.

4.1.2 The MBB-OTP protocol

The typical centralized OTP generation approach presents several weaknesses. As shown in Figure 4.1a, it groups all the steps in a single service exposing applications to MITM attacks. Moreover, if a hypothetical attacker violates the system, he/she could easily simulate the login of any user and his/her OTP generation.

Figure 4.1a identifies three conceptual phases and related processes in the OTP management:

- The Authentication process receives an access request from the end user and it manages all the authentication, authorization and accounting functionalities. It provides the input to the generation of the new OTP.
- The OTP Generator process runs the actual OTP generation algorithm. Starting from a shared secret, it generates the temporary password to access the application.
- The Sender process delivers the generated OTP to the end user.

Our MBB-OTP protocol applies the microservices architecture paradigm [101] in the whole OTP lifecycle using three loosely coupled fine-grained and lightweight services, each one responsible for one of the three main phases identified in the aforementioned typical

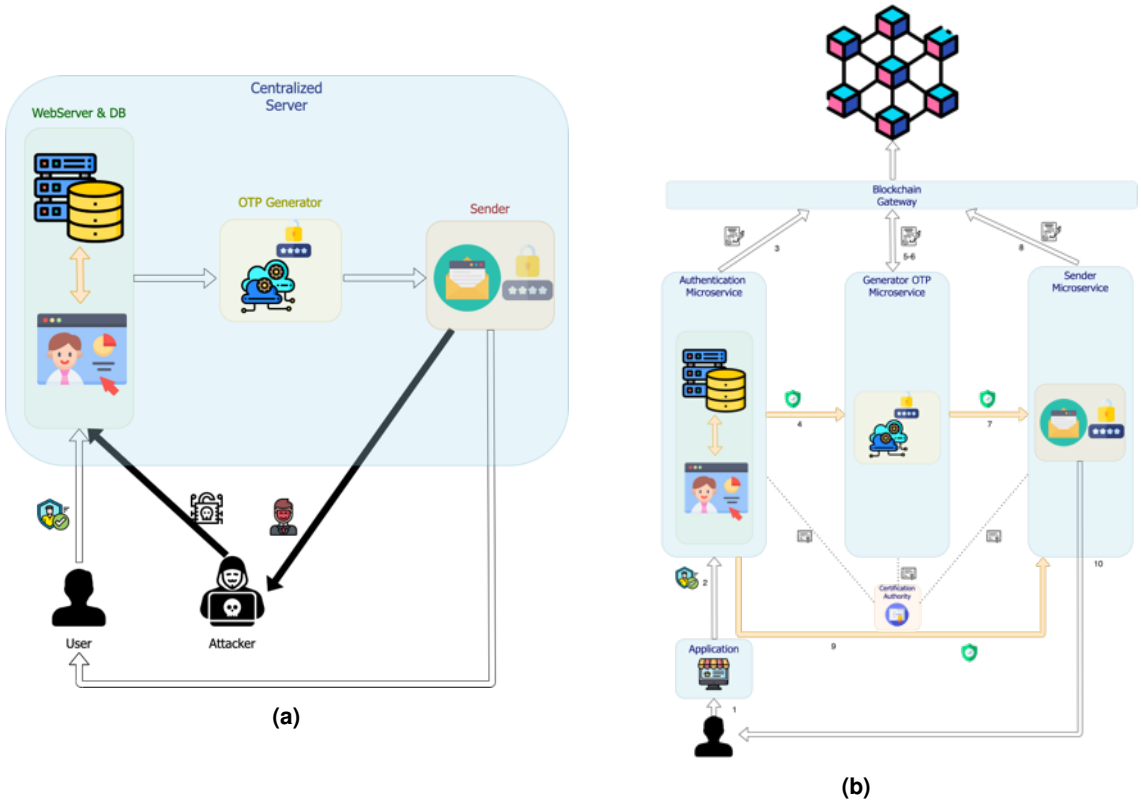


Figure 4.1: Centralized OTP management (left); MBB-OTP protocol architecture (right).

2FA approach. Moreover, the MBB-OTP protocol involves the use of Blockchain to ensure the integrity of tasks performed by each microservice. Therefore, MBB-OTP abstracts any specific application which uses it: different applications could potentially use the same MBB-OTP implementation. In fact, as it can observed in Figure 4.1b, the Application block is independent with respect to the other component blocks.

The MBB-OTP architecture mainly identifies the following component blocks:

- The Microservices: all the conceptual phases identified in the classical approach are implemented through different and independent microservices.
- Blockchain Gateway: it allows all microservices to write data about the tasks they perform in the Blockchain.
- Certification Authority: it manages the certificates that ensure secure communication between all the microservices.

Moreover, the protocol requires that the communication between the microservices has to be secured. Indeed, a Public Key Infrastructure (PKI), through a Certification Authority manages all the certificates of the microservices in order to sign and encrypt the information exchanged between microservices. The Certification Authority releases, after the deployment of each

microservices and during the configuration of the whole MBB-OTP system, a certificate ensuring that all communications are trusted. The integrity of each completed microservice task is ensured by the Blockchain network. In fact, the Blockchain plays an important role because it stores a shared secret that acts as a "seed" for the OTP generation. All the steps required by the MBB-OTP protocol to securely generate and deliver the OTP are shown in Figure 4.2:

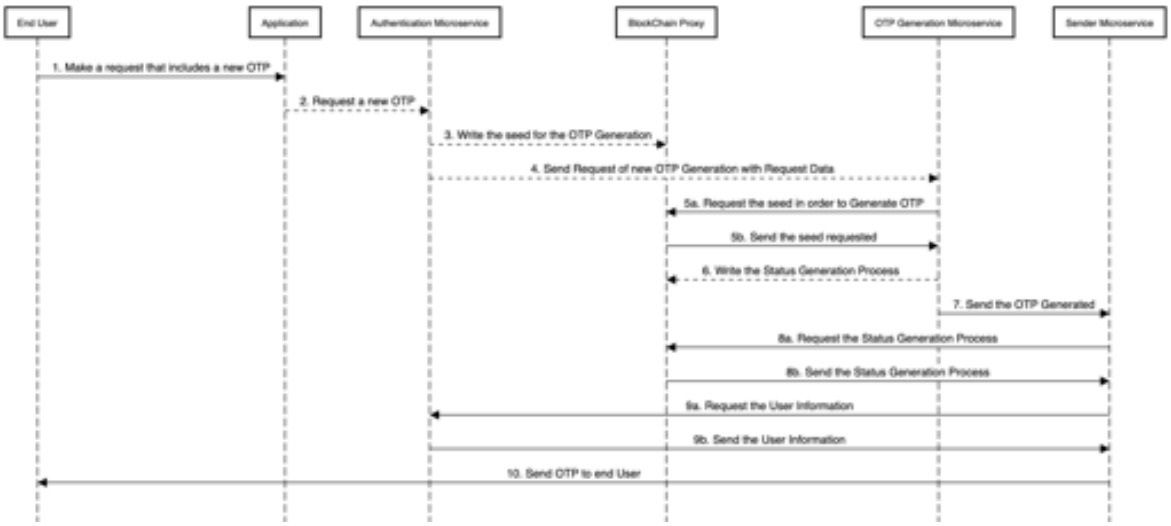


Figure 4.2: MBB-OTP protocol sequence diagram.

1. The end user sends an access request to the application that requires an authentication with OTP.

2. The application asks for the generation of a new OTP.

3. The Authentication Microservice (AMS) communicates with the Blockchain Proxy to write the seed for the OTP generation in the Blockchain. The information written in the Blockchain is signed with the Private Key of the AMS. The seed can be publicly available in the Blockchain because it is not a password on its own, and it needs to be combined with other elements to become an OTP. Furthermore, the seed is stored using the hash digest of a combination of user id and timestamp, so it is not possible to identify the seed of a requesting user without having the full set of attributes associated.

4. The AMS notifies the OTP Generation Microservice (OTP-GMS), providing the request timestamp and the key to retrieve the seed in the Blockchain. The information is encrypted and the communication is secure thanks to the PKI infrastructure.

5. The OTP-GMS retrieves the seed from the Blockchain through the information previously received from the AMS. After that, the OTP-GMS generates, using the received timestamp, the OTP with the TOTP algorithm, initializing it with the retrieved seed.
6. The OTP-GMS writes in the Blockchain the OTP generation process status to ensure the integrity of the flow.
7. The OTP-GMS sends the generated OTP to the Sender Microservice (SMS), encrypting the messing through the PKI infrastructure.
8. The SMS verifies the OTP Generation process status by retrieving the information from the Blockchain. This ensures that the OTP request is legitimately coming from the AMS and it followed the entire flow as expected. In case the SMS is invoked without a valid Generation process status present in the Blockchain, the service would be refused.
9. The SMS requires to the AMS the User Information, such as email address or mobile phone number to deliver the newly generated OTP.
10. The SMS sends the OTP to the end user who can complete his/her authentication to access the application.

Once the end user submits the OTP to access the application, the latter verifies it. The verification process is analogous to the initial OTP generation steps. The AMS retrieves the OTP generation timestamp and the Blockchain access key value from the Database used for authentication. Then, it sends them to the OTP-GMS that gets the seed and along with the timestamp generates an OTP. In the end, it sends back the OTP to AMS that can verify the validity with the submitted one.

The MBB-OTP protocol drastically reduces the risk of identity violation. Unlike the centralized 2FA approach, it separates all phases, making each component independent. For example, the OTP-GMS simply takes the seed and the timestamp to generate the OTP, sending that to the SMS. It does not store and it does not know any information about the end user requesting the OTP.

4.1.3 Implementation

In this Section, a prototype implementation of our MBB-OTP protocol is discussed. Each microservice is implemented through the programming language Python 3, the Flask framework and the Gunicorn Python Web Server Gateway Interface (WSGI) HTTP server. Furthermore, each microservice is deployed within a Docker container to take the advantage of the

virtualization technology allowing service portability, resiliency, and automatic updates that are typical of a distributed system acting either at the Cloud or the Edge [102]. The secure communication between the microservices is accomplished through the HTTPS protocol based on a certification authority created and managed through the OpenSSL tool. The implementation details of our prototype components are described below. The *AMS* component accomplishes Authentication, Authorization and Accounting (i.e., AAA functionalities). This microservice communicates with a DataBase Management System (DBMS) to retrieve registered user information. This is required to ensure that the entire process flow runs only if the requesting user is authenticated and recognized. The database implementation is out of the scope of this paper and it can be carried out either with SQL, NoSQL, or NewSQL solutions. Moreover, the AMS communicates with the Blockchain Gateway through REST calls and writes a random seed on Blockchain as a key-value pair.The key of the Blockchain entry is the hash digest of concatenation of the user id and the timestamp. The value is the seed. Once the Blockchain transaction is committed, the AMS sends to the OTP-GMS, in the form of a REST call, the request timestamp and the hash digest of the Blockchain transaction by which the seed is stored. The pseudo-code about how this component works is provided in Algorithm 2. The *OTP-GMS* component leverages the TOTP algorithm using the seed stored in the Blockchain that is retrieved through a REST call to the Blockchain Gateway. After that, the component sends the OTP through a REST call to the SMS component. The pseudo-code about how this component works is provided in Algorithm 3. The *SMS* component is responsible to deliver the OTP to the end user requesting to access an application. This component can be configured with multiple communication protocols (e.g., SMTP, MQTT, mobile push notifications, and so on). The *Blockchain Gateway* component can be implemented either with Hyperledger Fabric Private Blockchain or Ethereum Public Blockchain according to the specific application scenario and security policy. In fact, some application might require the adoption of a private authentication service, others might tolerate the use of a public one. The first one uses Fabric 2.2 with an architecture based on four peer nodes distributed in four remote servers and four orderer nodes. The peer node receives updates and broadcasts them to the other peers in the network. The orderer node is responsible for consistent Ledger state across the network, it creates the block and delivers that to all the peers. The Public Blockchain implementation leverages the Infura web-based Application Program Interface (API) to interact with Ethereum. The Smart Contract is written in JavaScript (in the case of Hyperledger Fabric) and Solidity (in the case of Ethereum) programming languages, and it is designed to store the hash digest of the user id and the seed in the form of free text. The

Smart Contract data structure includes two attributes: *Hash User Id* (String type), representing the anonymized user identification code; and *Seed* (String type), representing the value used as input for the TOTP generation.

Algorithm 2 OTP Request

Class *Authenticator*
method *RequestOTP*(*username*)
  user_id ← *getAuthenticatedUser*(*username*)
  seed ← *generateSeed*()
  hashForSeed ← *generateHashForSeed*(*user_id*, *current_timestamp*())
  hashForLog ← *generateHashForLog*(*current_timestamp*(), *user_id*)
  saveSeedToBlockchain(*hashForSeed*, *seed*)
  saveLogToBlockchain(*hashForLog*, *msg*)
  sendToOtpGenerator(*hashForSeed*)

Algorithm 3 OTP Generator

Class *Generator*
method *GenerateOTP*(*hashForSeed*)
  seed ← *getSeedFromBlockchain*(*hashForSeed*)
  OTP ← *generateTOTP*(*seed*, *current_timestamp*())
  hashForLog ← *generateHashForLog*(*current_timestamp*(), *hashForSeed*)
  saveLogToBlockchain(*hashForLog*, *msg*)
  hashForLog ← *generateHashForLog*(*current_timestamp*(), *user_id*)
  sendOtpToSender(*hashForLog*, *OTP*)

4.1.4 Experiments

Blockchain represents an additional element in our MBB-OTP protocol with respect to a traditional 2FA one. For this reason, this research is specifically focused on preliminary experiments aimed at assessing the overhead introduced by the private and public Blockchain in our protocol implementation. Specifically, the prototype is tested and compared the OTP generation time and resources usage in terms of CPU and inbound/outbound network traffic. With the private Blockchain approach, according to our MBB-OTP protocol the seed is stored in a private Blockchain by the AMS and retrieved by the OTP-GMS to generate the OTP. Such an approach obviously does not include any cost in terms of money. Whereas, with the public Blockchain approach, the seed is sent to a public Blockchain node to be added to the queue for immutable storage and mining. Time-to-mine and Ethereum (ETH) cryptocoin currency transaction cost are subject to Ethereum network traffic. In this case, the longer is the time required to mine, the higher is the cost to be paid for the transaction.

The testbed was arranged using a remote server with the following features: Intel® Xeon®

E3-12xx v2 @ 2.7GHz, 2 core CPU, 4 GB RAM running Ubuntu Server 18.04. For private Blockchain, tests have been performed with 2 CLI, 4 peers and 4 orderers distributed across 4 different hosts. For public Blockchain, all tests have been performed using Ropsten Ethereum public Blockchain test network, leveraging 300+ available nodes with a real server load status. Tests have been executed for 100, 200 and 300 read and write operations, considering 95% confidence intervals and the average values. A summary of experiments setups is reported in Table 4.1.

Table 4.1: Summary of experiments performed.

Parameter	Value
Contract address	0xa1032b39180538D7e17 23AedC39154C89A113BE5
Test executed for each scenario	[100, 200, 300]
Confidence interval	95%
Gas used by transaction	46,000
Gas price (Gwei)	200
Average cost per transaction (ETH)	0.0092326

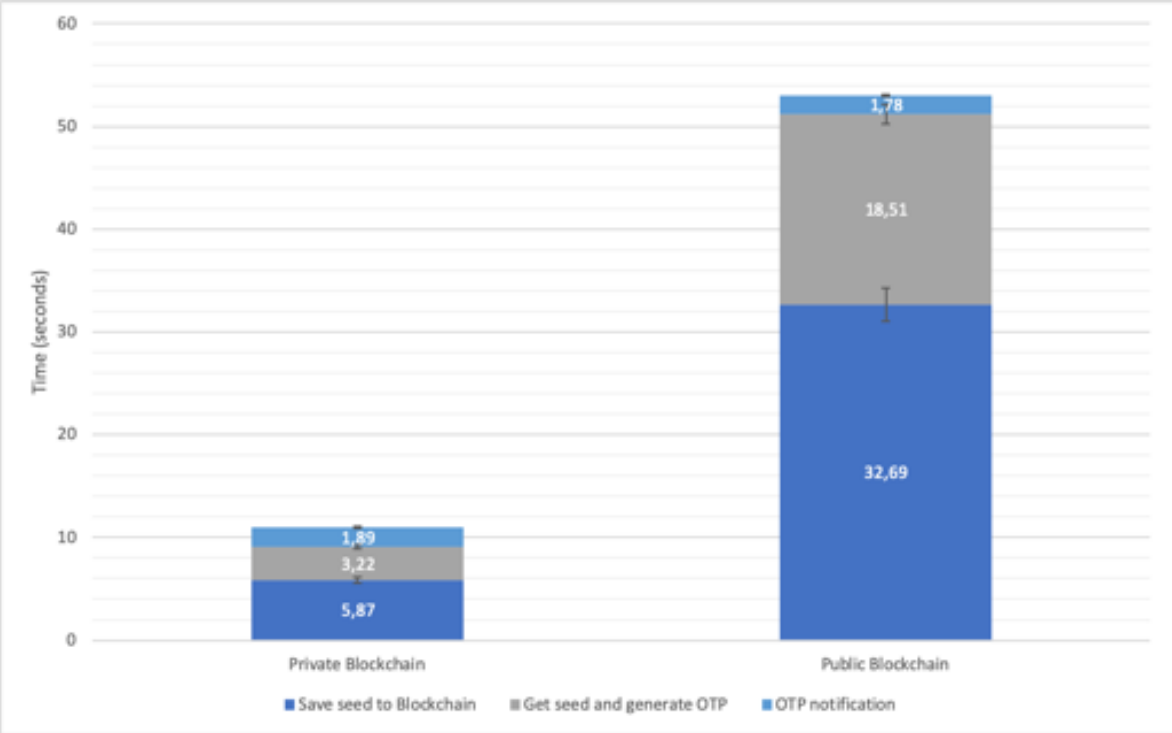


Figure 4.3: Execution time comparison for a single OTP generation (average) between private and public Blockchain.

Figure 4.3 shows the execution time difference expressed in seconds between the two system implementations to generate the OTP. On the x-axis, it is reported the two approaches for OTP generation. On the y-axis it is reported the processing time expressed in seconds. It

is possible to observe that the time required to generate the OTP is five times greater with the Public Blockchain approach, due to the mining time required to store and retrieve the seed in the Public Blockchain. Instead, the average time required to notify the newly generated OTP is similar in both implementations.

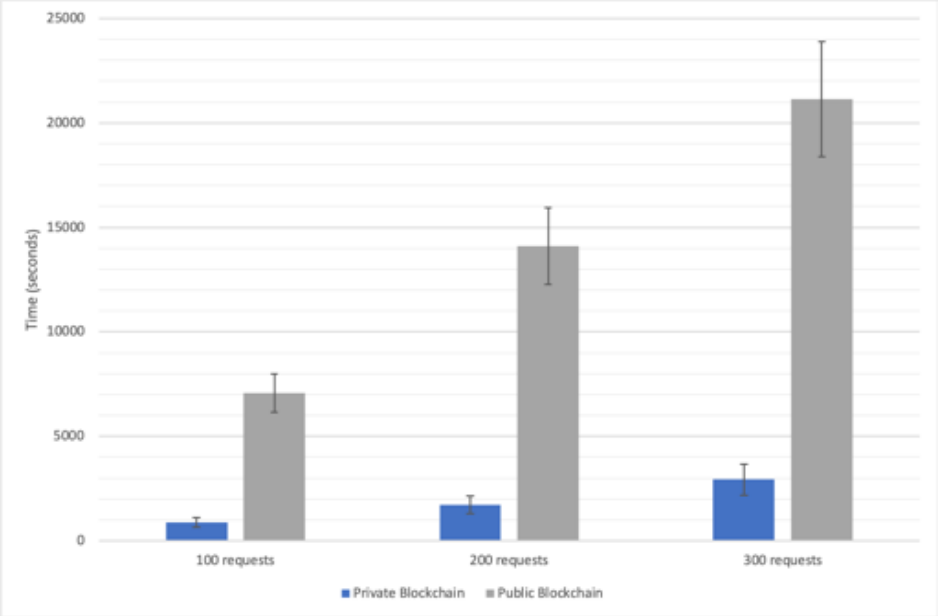


Figure 4.4

Figure 4.5: Execution time comparison between Private and Public Blockchain approach implementations for OTP generation.

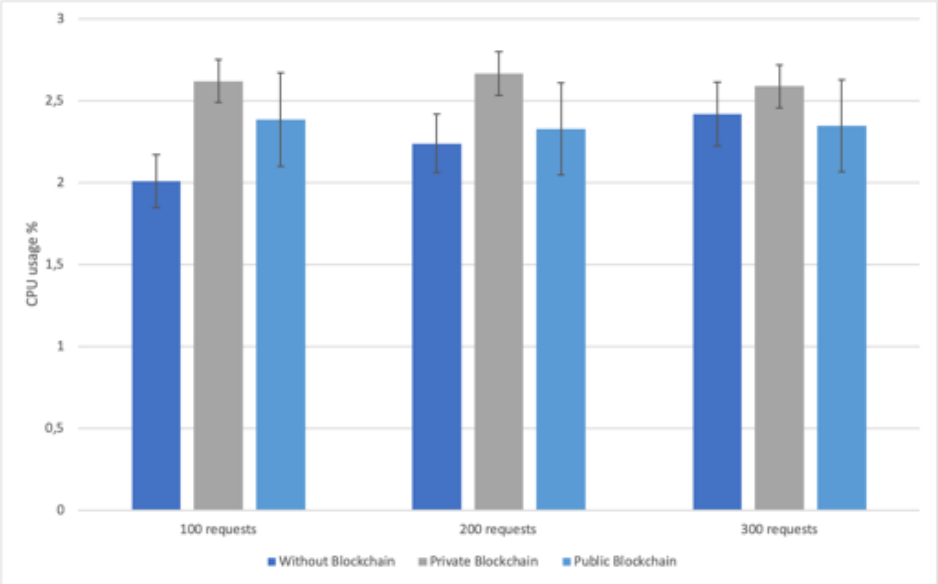


Figure 4.6

Figure 4.7: CPU usage % comparison between Private and Public Blockchain approach implementations for OTP generation.

Figure 4.4 shows the execution time difference expressed in seconds between the two system implementations to generate the OTP. On the x-axis, it is reported the differences to perform 100, 200 and 300 requests for OTP generation. On the y-axis it is reported the processing time expressed in seconds. From the graph, it is possible to observe that the time required to generate the OTP is largely different in the two implementations, and it shows a linear behavior. This is due to the mining time required to store the data in the public Blockchain that is considerably different from the private Blockchain approach.

Figure 4.6 shows the CPU usage % difference between the two system implementations for the OTP generation. On the x-axis, it is reported the differences to perform 100, 200 and 300 requests. On the y-axis it is reported the CPU usage %. It is possible to observe three different scenarios: a server that does not use the Blockchain, but it stores the seed in a local MySQL DBMS instance that is compared with both private and public Blockchain implementations. It can be appreciated that the CPU usage % is kept in low range values and it is comparable in the three scenarios.

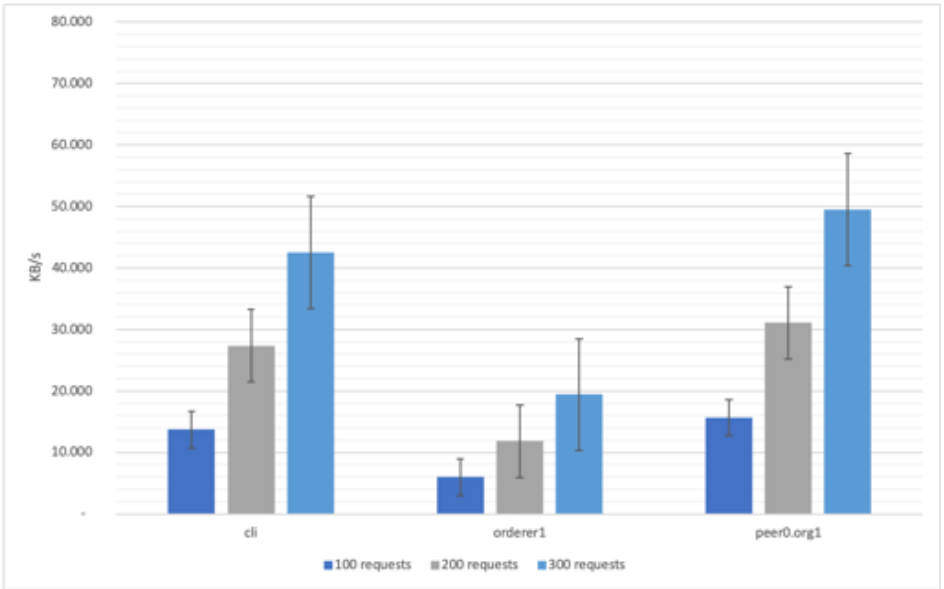


Figure 4.8

Figure 4.9: Inbound network comparison between Private Blockchain components required for the OTP generation.

Figure 4.8 shows the inbound network difference for the three main Hyperledger Fabric components deployed in host 1. On the x-axis, it is reported the differences to perform 100, 200 and 300 requests, whereas on the y-axis it is reported the Inbound network consumption expressed in KB/s. Hereby, it is possible to observe that the network traffic grows up as the number of requests increases with a linear behavior.

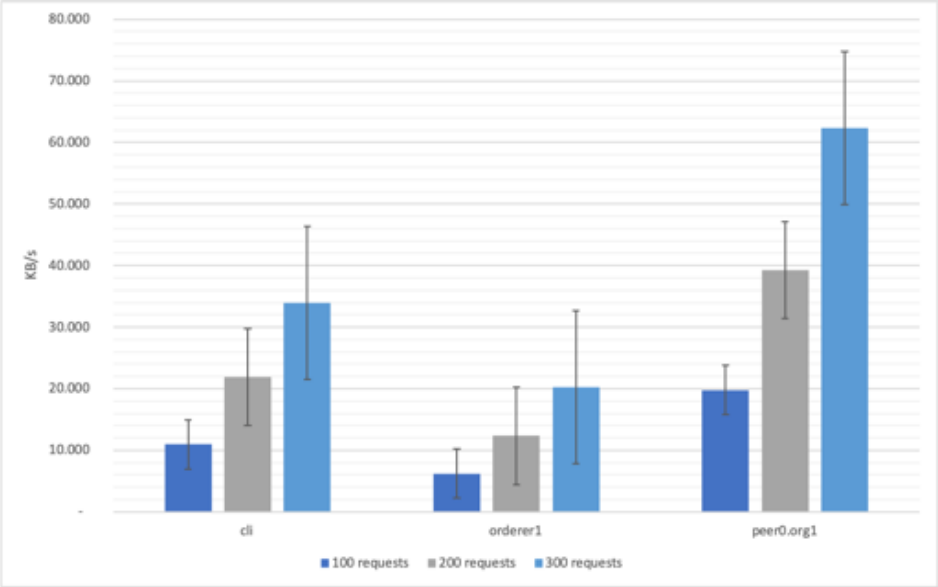


Figure 4.10

Figure 4.11: Outbound network comparison between Private Blockchain components required for the OTP generation.

Figure 4.10 shows the outbound network difference for the three main Hyperledger Fabric components deployed in host 1. On the x-axis, it is reported the differences to perform 100, 200 and 300 requests, whereas on the y-axis it is reported the Inbound network consumption expressed in KB/s. Hereby, it is highlighted that peer0.org1 has a considerable amount of outgoing network traffic. This is expected and is due to the block propagation required to share data with the other peers participating in the Private Blockchain network.

4.1.5 Some remarks

The increasing complexity of digital applications for social and business activities has required ever more advanced security mechanisms to prove the identity of subjects. T-OTP is one of the most common 2FA systems designed to achieve such an objective, but it based on a centralized architecture that represents a SPoF being consequently exposed to MITM attacks. To address such concerns, the MBB-OTP protocol is proposed that allows to combine the security advantages of a decentralized 2FA system with the features of data non-repudiation and immutability that are typical of Smart Contracts. This protocol can be seamlessly implemented in any human-to-machine and machine-to-machine communication well-fitting the recent Cloud, Edge and IoT scenarios.

The solution has been prototyped and tested with both private and public Blockchain approaches. In the first case, although the trusted network needs to be built with a reasonable

number of nodes to guarantee sufficient level availability and reliability, low response times is observed. Instead, in the second case, considering a high level of availability and reliability thanks to the larger number of nodes, it is observed an acceptable degradation of response times.

Experimental results highlighted that the CPU usage % remains roughly unchanged if using a DBMS such as MySQL to store the information, a private or public Blockchain, demonstrating that the MBB-OTP protocol can be implemented in any scenario without any performance degradation in terms of CPU usage. Obviously and as expected, the time required to store and retrieve any data to/from the public Blockchain heavily affects the overall performance, but waiting one minute to obtain a strongly secured generated OTP can be considered acceptable, considering the obvious security and robustness advantages.

The current MBB-OTP protocol, compared to traditional 2FA ones, mitigates MITM attacks and it can also potentially mitigate also DOS attacks by considering replicated microservices that can replace corrupted ones.

4.2 A Blockchain based improved version of the Extended Triple Diffie-Hellman protocol

The Extended Triple Diffie-Hellman (X3DH) protocol has been used for years as the basis of secure communication establishment among parties (i.e, humans and devices) over the Internet. However, such a protocol has several limits. It is typically based on a single trust third-party server that represents a single point of failure (SPoF) being consequently exposed to well-known Distributed Denial of Service (DDOS) attacks. In order to address such a limit, several solutions have been proposed so far that are often cost expensive and difficult to be maintained. The objective of this research is to propose a *BlockChain-Based X3DH (BCB-X3DH)* protocol that allows eliminating such a SPoF, also simplifying its maintenance. Specifically, it combines the well-known X3DH security mechanisms with the intrinsic features of data non-repudiation and immutability that are typical of Smart Contracts. Furthermore, different implementation approaches are discussed to suits both human-to-human and device-to-device scenarios. Experiments compared the performance of both X3DH and BCB-X3DH.

Nowadays, with the exponential growth of connected devices over the Internet, security is one of the major concerns for network communications between heterogeneous parties (i.e., people, devices, etc.). Security is not only about protecting the $confidentiality$ of messages exchanged between parties, but as stressed by the Central Intelligence Agency (CIA) of the

US Government in the "triad security policy development model" [78], but it involves also *integrity*, and *availability*.

Since 1976, secure communications rely, among others, on the Diffie-Hellman key exchange protocol [103]. However, at the beginning it presented an important limitation for parties because they had to stay online during the agreement process, so that, in recent years, a novel approach known as eXtended Triple Diffie-Hellman (X3DH) [104] was developed. The latter relies on the use of a centralized trusted server which is responsible to hold the public keys and the ephemeral one-time password of all involved parties. Furthermore, it is also available in case of offline parties, asynchronously enabling the agreement process. Although this protocol solves the connection limits, it carries an important security bottle-neck: the server used to perform the agreement offline represents a single point of failure (SPoF) being exposed to well-known Distributed Denial of Service (DDOS) attacks [76]. To mitigate such a risk, multiple strategies have been proposed, but these cause an increase in costs for maintaining a security infrastructure [77].

The objective of this research is to propose an alternative X3DH protocol based on Blockchain technology to eliminate such a SPoF. Specifically, it is proposed the *BlockChain-Based X3DH (BCB-X3DH)* protocol that allows us to combine the well-known X3DH security mechanisms with the intrinsic features of data non-repudiation and immutability that are typical of Smart Contracts. In simple words, in BCB-X3DH, the centralized server used to perform agreements among parties is replaced by a distributed Blockchain network.

Blockchain has been increasingly recognized as a technology able to address existing information access problems in different applications domains. Born in 2009 as the technology behind Bitcoin [15], it has completely revolutionized traditional encryption-based security systems, introducing a new approach applying hash-based encryption to link blocks of various content. Smart Contract is one of the major applications of Blockchain: it is a protocol aimed at digitally facilitating, verifying, and enforcing the negotiation of an agreement between parties without the need of a centralizes certification third party.

4.2.1 Background and Related Work

Recently, continuous research for the most secure communication method between parties has brought to the development of several scientific works.

The original Diffie-Hellman Key Exchange protocol over an untrusted network [103] has been revised in recent years many times up to the current version called eXtended Triple Diffie-Hellman, X3DH [104]. It enables key exchange even when one party is not available by

means of a third party that can be a single server or distributed system.

The Double Ratchet protocol [105] is an alternative solution that combines a cryptographic so-called "ratchet" based on the Diffie–Hellman key exchange (DH) and a ratchet based on a Key Derivation Function (KDF), e.g. a hash function. It features security properties such as encryption, integrity and authentication as well as plausible deniability and forward secrecy. This protocol has been firstly proposed and implemented in the signal message application [106] and has already been widely accepted by the community. In fact, it is currently implemented in most of the modern Instant Messaging systems such as WhatsApp and Facebook Messenger [107, 108].

Blockchain can be used as a piece of decentralized security framework that can be used to implement the third party of the X3DH protocol in a distributed fashion by means of the trustability and transparency offered by Smart Contracts [109]. In fact, it has been recently demonstrated how Smart Contracts can be trusted and secured through several tools [75]. In this context, several scientific works have been recently published demonstrating how Blockchain can be used as an alternative to classic server architecture to secure communications of devices context [110]. Furthermore, it was discussed how Blockchain and Smart Contract technologies can be adopted to achieve trustiness and transparency in different application domains [1]. Regarding the establishment of secure communications among parties, an implementation of Smart Contract for secured key exchange mechanism has been proposed in [111] implementing the basic Key Exchange algorithm on-chain as a Smart Contract in Ethereum.

Differently from the above mentioned recent scientific initiatives, this paper proposes a practical implementation of how Blockchain and Smart Contracts can be adopted to implement a trusted distributed system as an alternative to a single third party server to securely perform the X3DH protocol on-chain enhancing processing time and eliminating both SPoF and the risk of DDOS attacks.

4.2.2 Motivation

The X3DH protocol relies on a centralized server (or a cluster of servers) which holds the public keys of the participants to a secure communication channel as previously described. Central server architecture is limited in terms of security because it is susceptible to Man-in-the-middle (MITM) attacks [66]. This type of attack can take place in different forms: a) as a malevolent user who takes the control over the communication channel between legitimate parties sending altered messages to them; b) as a service malfunction refusing to deliver data

to one of the parties, causing the incomplete exchange of messages which can drastically alter the meaning of the communication. Moreover, this configuration is a SPoF because it is sufficient to attack the centralized server (or the cluster of servers) through a DDOS attack [67], making the entire service unusable. Although this is a well-known attack and it has origin at the beginning of the IT era, it is still one of the most used and crucial types of attack. DDOS consists in obtaining the disruption of services by attempting to limit access to a machine, making a network incapable of providing normal service by targeting either the network bandwidth or its connectivity. These attacks achieve their goal by sending to the victim host a stream of packets that saturate his network or processing capacity, denying access to his consumers. Nowadays, using simple and light-weight tools, malevolent users can malicious code into unaware victims to leverage a huge number of machines ready to run a distributed version of such attack. These attacks consume some critical resource at the target and deny the service to legitimate clients as the attack volume can be larger than what the system can handle. There are multiple strategies to prevent such attacks [76], but these have an immediate correlation with the increase of cost to maintain the secure infrastructure [77]. In case of X3DH implementation, the attacked services can be the server which handles all requests and the database which contains the public keys of all participants. During a DDOS attack, the malevolent user can gain root permission to the system or database and add or remove data from the system making it difficult to identify what is legitimate and what is altered. DDOS is just an example of possible distributed attacks, but the same principle applies for almost all of them.

4.2.3 System Design

In the X3DH system, it is fundamental to guarantee the trustiness of the third party server on which a secure message exchange is based on. In fact, if an attacker takes control of the server he/she can alter the communication between parties or can arbitrarily refuse to delivery one or more messages.

In the original Diffie-Hellman key exchange protocol, schematized in Figure 4.12, at the beginning two users, i.e., Alice and Bob respectively define a and b values. Afterwards, they agree upon two additional values: a prime number g and a primitive root modulus p, preferably using a different channel than the one they want to secure to avoid packet sniffing by malevolent users.

The key calculation uses the difficulty of computing logarithms over a finite field $GF(g)$ with a prime number g of elements, and its security depends on the difficulty of computing

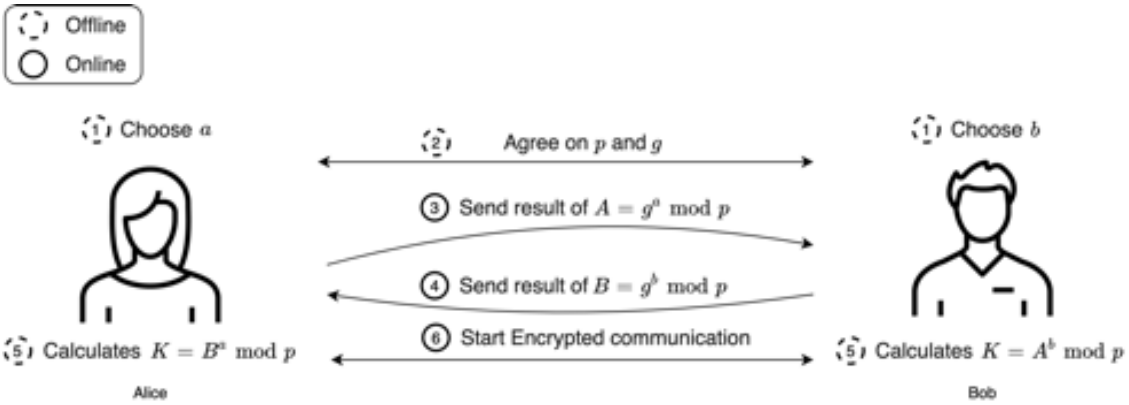


Figure 4.12: Original Diffie-Hellman scheme.

logarithms mod g. This approach has been used for decades, but the recent security requirements have required an innovative mechanism for agreeing upon a shared secret key.™ To overcome such a limitation, the X3DH protocol based on Elliptic Curve cryptography has been proposed to add the capability to exchange the secret key even when one of the participants is offline (further details are available in Chapter 3 of X3DH protocol [104]). The

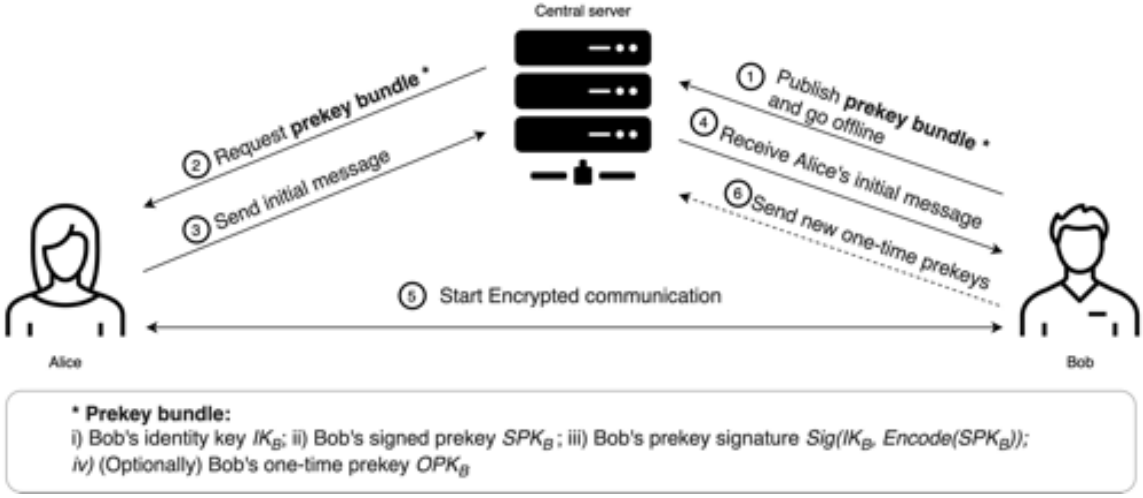


Figure 4.13: Extended Triple Diffie-Hellman scheme.

system described in Figure 4.13 highlights the following phases:

1. **Publish keys:** Bob publishes a set of public keys to the server, known as "prekey bundle", containing Bob's identity key IK_B, Bob's signed prekey SPK_B, Bob's prekey signature $Sig(IK_B, Encode(SPK_B))$, (Optionally) Bob's one-time prekey OPK_B;

2. **Request prekey bundle:** Alice, who wants to secure the communication with Bob, contacts the server and fetches Bob's prekey bundle. The server sends the bundle to Alice including one of Bob's one-time prekeys if it exists, and then delete it. If all of

Bob's one-time prekeys on the server have been deleted, the bundle will not contain a one-time prekey;

3. **Send initial message:** Alice calculates her secret key (SK) and sends an initial message to Bob containing her Identity Key IK_A and Ephemeral Key EK_A;

4. **Receive Alice's initial message:** Bob extracts Alice's keys and calculates the secret key (SK);

5. **Start encrypted communication:** Upon successful calculation of SK for both Alice and Bob a secure communication can begin between participants;

6. **Send new one-time prekeys:** At any time Bob may upload new one-time prekeys as these keys are deleted every time a participant requests a prekey bundle.

Starting from the X3DH scheme, in this scientific work it is proposed an improved version of X3DH aimed at eliminating the drawbacks of a central server and taking the advantage of inherit features of security and trustiness of a Blockchain network. This new alternative protocol is named *BlockChain-Based X3DH (BCB-X3DH)*. Figure 4.14 schematizes it.

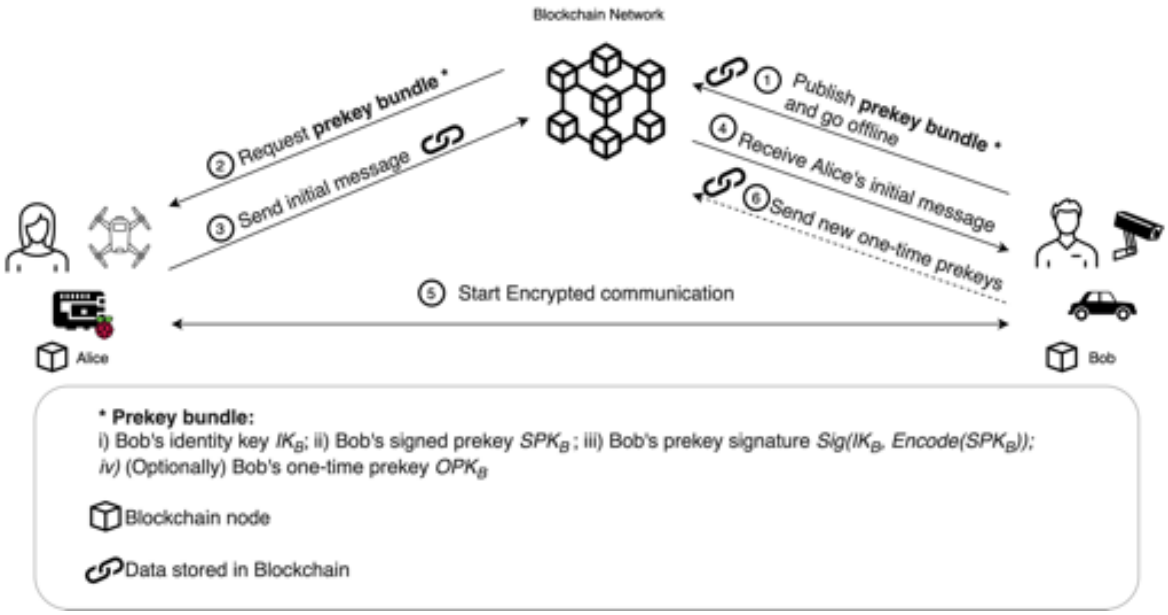


Figure 4.14: Blockchain-based X3DH Scheme.

The central server is replaced by a Blockchain network which collects data in an immutable way and append-only mechanism so that Alice and Bob act as Blockchain nodes. With this approach, the vulnerability of a central server architecture is eliminated as previously motivated. Every important information (e.g., related to Public keys) that requires to be

immutably stored in the Blockchain is sent to the local node which delivers it to the public network to be added in the mining queue. Depending on the system's resource of the client, it can participate in the mining pool or not without altering the whole architecture, assuming the system has enough full Blockchain nodes with mining capability.

The BCB-X3DH protocol can be seamlessly adopted in device-to-device communication scenarios including IoT where low-resource smart devices (e.g. drones, Raspberry Pi, cars, CCTV cameras, etc.) can act as light Blockchain nodes without significantly altering their system configurations.

The implementation details of the proposed architecture using the Ethereum Blockchain network is described in the next section.

4.2.4 Implementation

In this section, a prototype implementation of the BCB-X3DH protocol is discussed. Ethereum [79] was chosen as Blockchain network since it is currently the most widely used Blockchain platform thanks to its flexibility in developing Distributed Apps (DApps) through Smart Contracts. Moreover, this has been chosen against private Blockchain implementation such as Hyperledger Fabric [80] or Sawtooth [81] due to the fact that the greater number of nodes on public Blockchain network improves the trustiness of the system.

Three approaches were picked out to implement the system prototype: i) using a full Ethereum node for each user, which stores the whole Blockchain data and participates in the mining process for the consensus mechanism; ii) using a light Ethereum node running in Light Sync Mode [112] which downloads only a minimal part of the Ethereum Blockchain (i.e., 2,300 blocks in the past) and does not participate in the mining process; and iii) using a web-based Application Program Interface (API) such as Infura [83] to interact with Ethereum to be more efficient in terms of resource because the only requirement for the user is to be able to perform secure HTTP requests. In this research, it is explored the third approach because it well suits, apart from human-to-human communication scenarios, also human-to-machine and machine-to-machine ones including low-resource IoT devices. However, in all the aforementioned approaches, resources and time required to submit a transaction to the Ethereum Blockchain network is comparable and it does not depend on the type of considered client (either human or device).

Furthermore, two Implementation Strategies (IS) were analyzed considering both pros and cons:

- **IS1**: random seed and the key pairs are generated locally at the client-side and rely on the Blockchain network only for offline key exchange
  - **Pros**. It well suits the requirements of a human-to-human secure communication channel over the Cloud. Keys are generated locally and only the public part is sent over the untrusted HyperText Transfer Protocol (HTTP) channel (as the public key is meant to be used);
  - **Cons**: generate the random seed locally can be computationally challenging for low-performance devices (in case of machine-to-machine secure communications typical of Edge computing and IoT scenarios).
- **IS2**: keys generation is processed on the Smart Contract in the Blockchain network; the Public Key is stored in the Blockchain and the Private key is sent to the requester and never stored on the network
  - **Pros**. It well suits the requirements of a human-to-human, human-to-machine and machine-to-machine secure communication channels over the Cloud, Edge and IoT. In fact, every type of client, including also the low-resource device can gain benefit from the BCB-X3DH protocol as the crypto-computation is not required;
  - **Cons**. communication with the node processing the Smart Contract needs to be end-to-end secured otherwise it is vulnerable to Man-in-the-Middle attack.

In this research, the experimentation begins with a human-to-human secure communication scenario, focusing the study on the IS1. The Smart Contract, written in Solidity programming language, is the core of the proposed implementation. The main *User* data structure of the Smart Contract is shown in Table 4.2.

Table 4.2: User's data structure of the implemented Smart Contract.

Attribute	Data Type	Description
IK	string	User's Identity Key
SK	string	User's Signed Prekey
PreSign	string	User's Prekey Signature
OPK	string	User's One-time Prekey (Optional)
EK	string	User's Ephemeral Key (Optional)

Before calculating the Secret Key (SK) to be used for secure communication with our BCB-X3DH protocol, considering scheme depicted in Figure 4.14, Bob needs to store his Identity and Public keys in the Ethereum Blockchain network by calling an $external public function$. Similarly, Alice, and in general all users who want to start a secure communication with Bob,

need to retrieve Bob's Public keys from the Blockchain network to calculate the SK. To do so, Alice only requires to know Bob's ID, which can be a simple alphanumeric string. After receiving Bob's Prekey bundle, Alice calculates Bob's Prekey Signature which has to match with what is stored in the Blockchain. Upon successful verification of Bob's Prekey Signature, Alice sends to the Blockchain network some data known as Initial Message, containing Identity Key (IK) and Ephemeral Key (EK), required by Bob to calculate the same SK. It is important to note that Users pay the Blockchain transactions only when it is required to store data (i.e. storing keys), and do not pay any requests for data retrieval.

Once the Secret Key is calculated by both participants, it can be used to secure any communication over untrusted channels.

4.2.5 Experiments

The objective of experiments discussed in this Section is to verify if the overhead introduced by Blockchain in our BCB-X3DH prototype implementation, is acceptable in terms of both read and write access times, compared to the classic X3DH implementation based on a single third-party server. Specifically, the following implementations is considered:

- **Traditional X3DH**: Bob's key pairs are generated locally and the public prekey bundle, containing identity key IK_B, signed prekey SPK_B and one-time prekey OPK_B, is sent to a remote single third party server along with Bob's identification key (a simple alphanumeric string) and stored in a MySQL database. Alice requests this information to the server and then generates the Secret Key.

- **BCB-X3DH**: Bob's key pairs are generated locally, in the same manner as the first approach, and the prekey bundle is sent to a Blockchain node to be added in the queue for mining and immutable storage (time-to-mine and Ethereum (ETH) cryptocoin currency transaction cost are subject to Ethereum network traffic, and the more time is required to mine the higher is the cost to be paid to mine a transaction). Alice interrogates a Blockchain node to retrieve this information and then generates the Secret Key.

The system prototype assessment was conducted analyzing the total execution time required to complete both X3DH and BCB-X3DH processes. Each User was simulated considering a node with following hardware/software configuration: Intel® Xeon® E3-12xx v2 @ 2.7GHz, 4 core CPU, 4 GB RAM running Ubuntu Server 18.04. Each test has been repeated 30

times considering 95% confidence intervals and the average values. All tests have been performed using Ropsten Ethereum public Blockchain test network, leveraging 300+ available nodes with a real server load status. A summary of experiments setups are reported in Table 4.3.

Table 4.3: Summary of experiments performed.

Parameter	Value
Test executed for each scenario	30
Confidence interval	95%
Gas used by transaction	72,059
Gas price (Gwei)	20
Average cost per transaction (ETH)	0.0014411

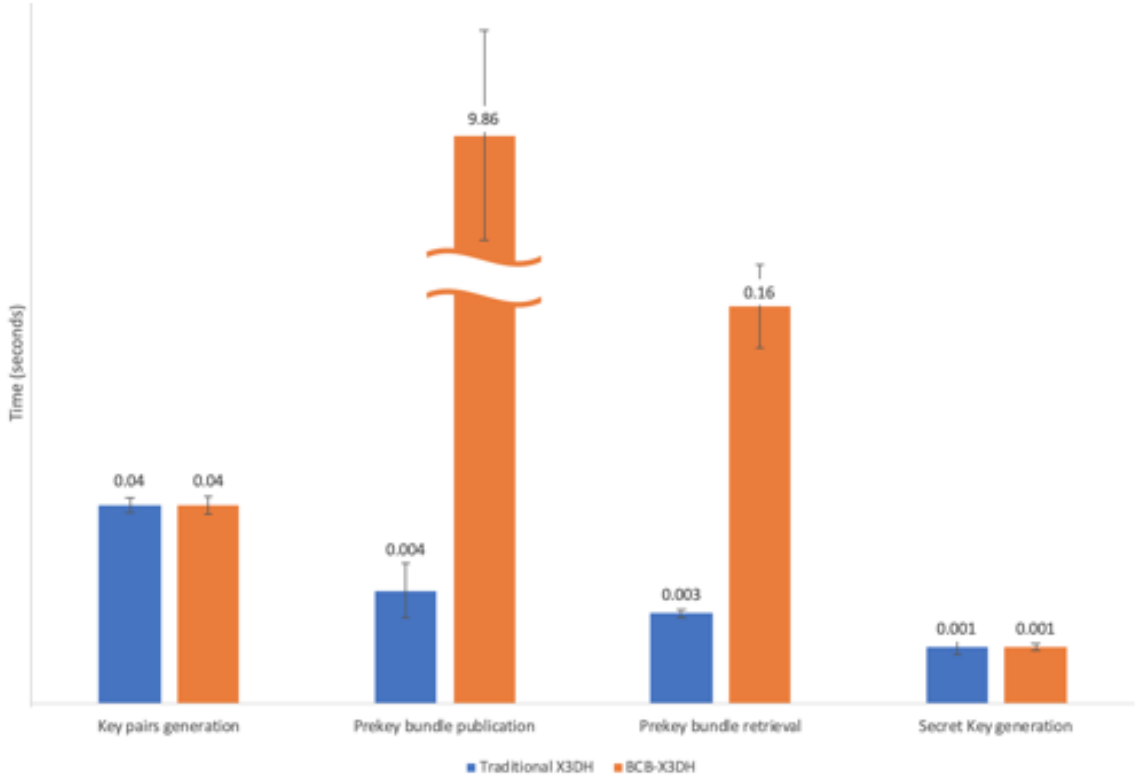


Figure 4.15: Response time comparison between Traditional X3DH and BCB-X3DH implementations for a complete key exchange process.

Figure 4.15 shows the execution time difference expressed in seconds between the two system implementations considering a complete key exchange process. On the x-axis, it is reported the different steps required to perform the key exchange workflow, whereas on the y-axis it is reported the processing time expressed in seconds. From the graph, it is possible to observe that the time required to perform Key pairs generation and Secret Key generation is the same in both implementations, as these are not dependent neither on the

single third-party server nor on the Ethereum Blockchain network. Instead, the time required to publish the prekey bundle is considerably different in the two system implementations, ranging from less than a millisecond up to 10 seconds. This is due to the mining time required to immutably store the data collected in the Blockchain transaction. Similarly and as expected, the time required to retrieve the Prekey bundle increases in the proposed approach as the access to the information stored in the Blockchain is slower than SQL access, but this time difference is less considerable. It is worth to consider that the time difference for Prekey bundle publication and retrieval is acceptable given the security benefit introduced by the Blockchain. In this implementation, Ethereum has been chosen as the underlying technology that leverages 1.5 millions of nodes, at the time of writing, thus making practically unfeasible SPoF or MITM attacks.

Apart from processing time, an additional analysis has been performed considering the ETH cryptocoin currency. Although Blockchain transaction writing comes with a cost, this is considered negligible as compared to the cost of preparing, configuring and maintaining a secure network infrastructure. In fact, with this solution, it is possible to perform over 50,000 transactions spending as much as a basic server configuration.

4.2.6 Some remarks

The Extended Triple Diffie-Hellman (X3DH) protocol has been used for years based on secure communication establishment among parties (i.e, humans and devices) over the Internet. However, such a protocol has several limits. It is typically based on a single trusted third-party server that represents a single point of failure (SPoF) being consequently exposed to well-known Distributed Denial of Service (DDOS) attacks. In order to address such a limit, several solutions have been proposed so far that are often cost expensive and difficult to be maintained.

To address such concerns, the *BlockChain-Based X3DH (BCB-X3DH)* protocol is proposed that allows us to combine the well-known X3DH security mechanisms with the intrinsic features of data non-repudiation and immutability that are typical of Smart Contracts.

The BCB-X3DH protocol, thanks to the flexibility of the Blockchain technology, can be implemented according to different strategies to establish human-to-human, human-to-machine and machine-to-machine secure communications in both Cloud/Edge computing and IoT scenarios.

This research is specifically focused on the implementation of the BCB-X3DH protocol aimed at a human-to-human secure communication in a Cloud scenario. In particular, the

Ethereum public Blockchain network is adopted to consider a large number of nodes (a greater number of nodes means a greater degree of security) but, in enterprise environments, private Blockchain technologies such as Hyperledger Fabric can be also adopted if the number of involved nodes is reasonably large in order to resist to any distributed attack.

Experimental results highlighted that the overhead introduced by Blockchain in the BCB-X3DH implementation in terms of execution time is acceptable, considering the obvious security and robustness advantages obtained by the Blockchain technology.

However, the BCB-X3DH protocol, considering IS2 discussed in Section 4.2.4, also well suit the requirements of human-to-machine and machine-to-machine in Cloud/Edge computing and IoT scenarios. In fact, most recent IoT devices lack of significant memory and processing resources to use public key cryptography techniques as they have typically very limited memory, processing, and energy resources. Furthermore, low-energy IoT devices with attached sensors adopted in many application scenarios (such as smart cities [113], eHealth [2], marine monitoring [114] and so on) that are typically battery powered and rechargeable through solar panels, struggle to maintain a secure communication channel for sending the collected data to the Cloud. The X3DH protocol expects parties to calculate their key-pairs sending the public keys to the server which only performs the key exchange part. Although this is the common practice for most of the cases, it is inefficient and sometimes unfeasible for low-resource IoT devices [115]. The BCB-X3DH protocol, decentralizing part of the secure communication channel maintenance to a Blockchain network, can lighten the workload of such IoT devices, hence improving their performance.

In future developments, the IS2 of the BCB-X3DH protocol will be studied in order to fit Edge and IoT scenarios including low-power embedded devices with limited hardware capabilities with the purpose to guarantee complete on-chain secure communication management, even optimizing battery life-cycle at the same time.

4.3 Innovating the X3DH protocol for IoT with Blockchain

The massive use of the Internet with the increase in the number of IoT devices in the last decade is leading to explore new frontiers to ensure secure communication between people and devices. Existing solutions have severe limitations in terms of security and resistance to distributed attacks, and a Blockchain-based solution has not yet been explored. The Extended Triple Diffie-Hellman (X3DH) protocol has been used for years but it is typically based on a single trust third-party server that represents a single point of failure (SPoF) being

consequently exposed to well-known Distributed Denial of Service (DDOS) attacks. The *BlockChain-Based X3DH (BCB-X3DH)* protocol has already been proposed to eliminate such a risk, combining the well-known X3DH security mechanisms with the intrinsic features of data non-repudiation and immutability that are typical of Smart Contracts. In this research, the research progresses further to implement this protocol in order to fit Edge and IoT scenarios including low-power embedded devices with limited hardware capabilities with the purpose to guarantee complete on-chain secure communication management, even optimizing battery life-cycle at the same time.

When talking about distributed devices, Cloud Computing and IoT technologies, securing communications between devices and humans is always a priority. Several encryption mechanisms have been proposed over the years, making brute force attacks [116] practically unfeasible, but all of those rely on a secret key that, as the name clearly suggests, needs to remain secret to any person or device apart from the intended parties.

The key exchange protocol has been the bottleneck of the entire secure systems for several centuries, and one of its first evidence is formalized by the byzantine Generals problem [117].

In any IT system, it must be guaranteed that messages exchanged between parties are confidential, parties are authentic and legit and messages are delivered only to the intended receiver. Message encryption has been proposed and experimented since the beginning of time (e.g. Caesar's code), but an efficient key-exchange protocol based on mathematical operation has been presented in 1976 only, by Whitfield Diffie and Martin Hellman [103]. Based on discrete logarithms, it was practically unfeasible to break with the computational power of that era, and this protocol has been used for almost half a century.

In more recent years, to overcome the limitation that parties need to meet, physically or virtually, to exchange the seeds to generate the secret key, the first improved version of the Diffie-Hellman key exchange protocol has been developed, known as eXtended Triple Diffie-Hellman (X3DH) [104]. This relies on the use of a centralized trusted server which is responsible to hold the public keys and the ephemeral one-time password of all involved parties. Furthermore, it is also available in case of offline parties, asynchronously enabling the agreement process.

Unfortunately, this protocol carries an important security bottle-neck: the server used to perform the key exchange represents a single point of failure (SPoF) and can be exposed to well-known cyberattacks [76].

Several strategies have been proposed to mitigate such a risk, but these have an immediate correlation with the increase in costs for maintaining a security infrastructure [77].

In Section 4.2, an alternative X3DH protocol based on Blockchain technology has already been discussed, with the objective to eliminate the SPoF and obtain a secure infrastructure with a fraction of the cost [74]. Specifically, the BlockChain-Based X3DH (BCB-X3DH) protocol allows to combine the well-known X3DH security mechanisms with the intrinsic features of data non-repudiation and immutability of Smart Contracts, and the centralized server used to perform agreements among parties is replaced by a distributed Blockchain network. However, in Section 4.2, the reference scenario with resources running on the Cloud was considered.

In this research, the research progresses further to identify an alternative approach optimized for IoT and, in general, low-resource devices, aimed at lightening such devices from public-key cryptography tasks guaranteeing energy cost efficiency.

The key generation is processed on the Smart Contract and the Public Key of the participants is stored in the Blockchain. The Private key generated is sent to the requester and never stored on the network. This permits to devices with low computation capabilities to securely encrypt and send data without the need to generate the key-pair.

The major contributions of this research are as follows:

- an analysis of strategies and frameworks to secure IoT communications has been performed, with the focus on Elliptic Curve-based cryptography for devices with limited computational capabilities;
- an analysis of diffused distributed cyber-attacks has been performed, with the focus on communication disruption or service unavailability, analyzing the Confidentiality, Integrity and Availability (CIA) triad.
- the BCB-X3DH protocol, enhanced for IoT and low-resource devices, is proposed to ensure secure and efficient secret key generation and exchange.

The remainder of this research is organized as follows. A brief overview of the most recent techniques to secure network communications is provided in Section 4.3.1. Motivations are discussed in Section 4.3.2. Starting from the traditional X3DH protocol, the design of the BCB-X3DH one is discussed in Section 4.3.3. A new implementation strategy to fit the requirements of machine-to-machine secure communications in Cloud/Edge computing and Internet of Things (IoT) scenarios is discussed in Section 4.3.4. Experiments comparing the traditional X3DH and BCB-X3DH protocol implementations focusing on device-to-device secure communication are discussed in Section 4.3.5. In the end, conclusions and light to the future are discussed in Section 4.3.6.

4.3.1 Background and Related Work

The increasing number of smart home devices requires the use of more efficient [118] and safer [119] communication protocols. There is no one-fit-for-all security protocol, and many scenarios have been proposed, each with some limitation or compromise.

This section highlights the difference between human-to-human and machine-to-machine implementations.

The original Diffie-Hellman Key Exchange protocol over an untrusted network [103] has been revised in recent years many times up to the current version called eXtended Triple Diffie-Hellman, X3DH [104]. It enables key exchange even when one party is not available through a third party that can be a single server or distributed system.

A simple strategy to secure communication can surely be to use a combination of symmetric and asymmetric encryption techniques: the first is used to encrypt a message using a pseudo-random generated key; then the ciphered text and the generated key are concatenated and encrypted using a private key. Only the receiver knows the public key and can then split and decipher the message [120]. Although this simple mechanism can be efficient in terms of cost and performance, a malevolent user can intercept the private-public key exchange and can decipher or alter all communications, making the entire system insecure.

A lightweight identity-based cryptosystem suitable for IoT has been proposed [121], which employs Physically Unclonable Function (PUF) to generate the public key uniquely for each device. However, this solution is vulnerable to Single Point of Failure (SPoF) as a compromised server could deliver messages to a malicious node.

To secure communication between devices, Message Queuing Telemetry Transport, better know as MQTT, is often used, which provides a publish-subscribe mechanism on individual topics. This approach has been tested and compared with Blockchain technology and it has been proved that the latter approach offers better results to resist when the infrastructure is under attack [122].

It has been demonstrated that for a different scenario, such as resource-constrained IoT devices, different algorithms should be used to secure communications. In fact, Elliptic Curve-based cryptography presents a better alternative than RSA for low-resource devices [123].

Furthermore, it was discussed how Blockchain and Smart Contract technologies can be adopted to achieve trustiness and transparency in different application domains [1, 124, 125, 126].

Differently from the above mentioned recent scientific initiatives, this paper proposes a practical implementation of how Blockchain and Smart Contracts can be adopted to implement a trusted distributed system as an alternative to a single third party server to securely perform the X3DH protocol on-chain enhancing processing time and eliminating both SPoF and the risk of DDOS attacks.

4.3.2 Motivation

Section 4.2, focuses on an important security limitation in systems that implement the X3DH protocol: a weak resistance against distributed attacks. We proved how a Blockchain-based system can overcome such limitation and how it can also act as a cost-saving solution as compared to setup a secure infrastructure.

Distributed Denial of Service (DDOS) attacks aim at overloading a server with requests to saturate system resources or network bandwidth, with the objective to make the server unable to provide normal services. Several tools are available online and ready to use to send to the victim host a huge number of data packets, and unaware victims of cyber attacks can participate to run a distributed version of this attack.

Man-in-the-Middle (MITM) attacks target sniffing data packets of a conversation between two parties and impersonate one or both the victims establishing a connection that deviates legit traffic. This causes a compromised messages exchange sending fake messages to one or both the victims, altering the complete communication.

In this research, considering an IoT scenario composed of constrained devices (such as a smart cities), aiming at bringing the analysis forward, testing the applicability of Blockchain-based X3DH solutions for low-resources devices.

Furthermore, low-energy remote sensing devices, as the case of Sea Monitoring IoT [114], powered by solar energy suffer from the impossibility to spend resources to ensure a secure channel to send the collected data. The result of this can be sending of raw unprocessed data to the Cloud computing engine, which increases the bandwidth required for each communication. In the disconnection on-the-edge scenario a low-cost sensing device is placed in an isolated area and collects data storing it offline in a local memory drive. This device does not require a continuous network connection, and it receives it once a day through the proximity of a drone. These sensing devices are battery-powered, rechargeable through solar panel, hence are resource-constrained when it comes to encrypting data before sending it to the server. These are just a few among several examples of low-resources devices constraints.

4.3.3 System Design

The X3DH protocol permits to exchange a secret key without the need of both participants to be simultaneously online. This is possible thanks to a central server (or a cluster of servers) that holds part of the seeds to generate the shared secret key.

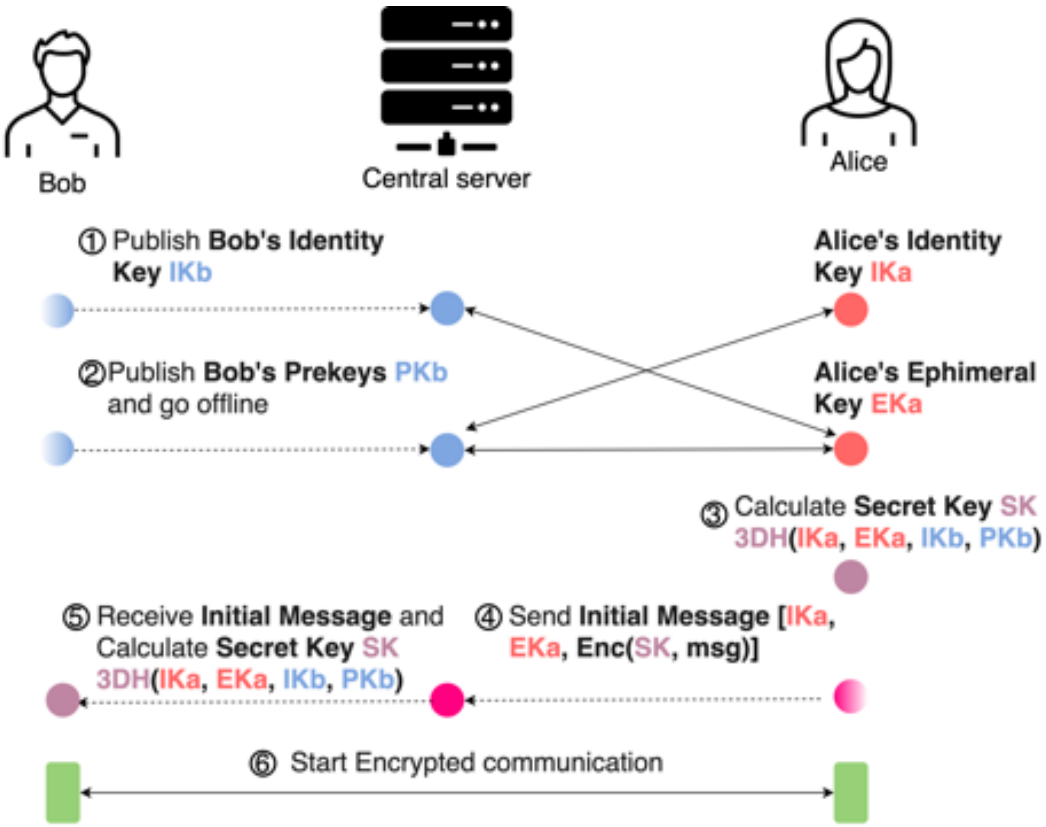


Figure 4.16: Extended Triple Diffie-Hellman scheme.

The system described in Figure 4.16 highlights the following phases:

1. **Publish Identity key:** Bob registers on the server publishing his identity key IK_B
2. **Publish keys:** To start the key-exchange protocol, Bob publishes a set of public keys to the server, known as "Prekeys", containing Bob's signed prekey SPK_B, Bob's prekey signature $Sig(IK_B, Encode(SPK_B))$ and, optionally, Bob's one-time prekey OPK_B;
3. **Secret key generation:** Alice, who wants to secure the communication with Bob, contacts the server requesting Bob's Prekeys. To generate the Secret Key (SK), Alice needs her Identity Key IK_A, her Ephemeral Key EK_A and Bob's public keys downloaded from the server;
4. **Send initial message:** after calculating SK, Alice sends an initial message to Bob containing her Identity Key IK_A, Ephemeral Key EK_A and a ciphered message encrypted

with the *SK*;

5. **Receive Alice's initial message:** Bob extracts Alice's keys and calculates *SK*;

6. **Start encrypted communication:** Upon successful calculation of *SK* from both Alice and Bob, secure communication can begin between participants.

Starting from the X3DH scheme, it is proposed an efficient application of the improved version of the *BlockChain-Based X3DH (BCB-X3DH)* dedicated for low-resource devices that have not enough computational power to generate secret keys ensuring secure communication. Figure 4.17 schematizes it:

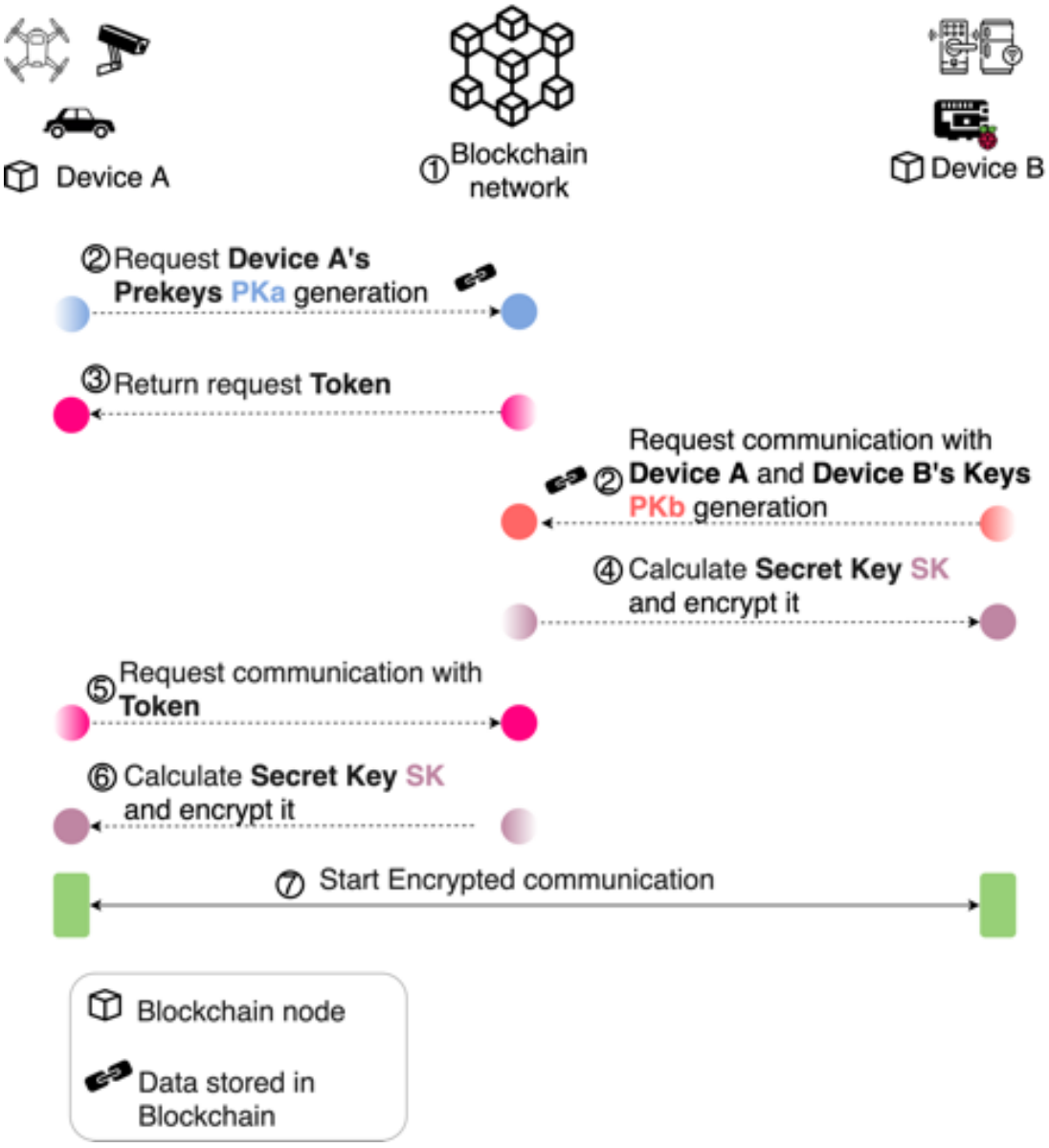


Figure 4.17: Blockchain-based X3DH Scheme.

1. The central server is replaced by a Blockchain network that, thanks to a Smart Contract, generates the secret keys upon request (this communication needs to be secured via a Virtual Private Network (VPN) or any End-to-End encryption mechanism);
2. A generic Device requests a Secret Key (SK) exchange with another Device providing a generated password (GP);
3. If Device A's Prekeys are not available or present, the Smart Contract returns a unique identifier for the request;
4. If Device A's Prekeys are available, the Smart Contract generates the Secret Key, encrypts it with GP and the result is returned to the requester;
5. A Device requests a SK passing the unique identifier received and a generated password GP;
6. If Device B's Prekeys are available, the Smart Contract generates the Secret Key, encrypts it with GP and the result is returned to the requester;
7. When both Devices have received the SK, a secure encrypted communication can start.

Key pairs are generated by the Smart Contract, but only Public keys are stored in the Blockchain. The Private Key is encrypted and sent back to the requester via the same channel that has to be secured.

The IoT-oriented BCB-X3DH protocol is meant to be adopted in device-to-device communication scenario for low-resource smart devices (e.g. drones, Raspberry Pi, cars, CCTV cameras, etc.). The implementation details of the proposed architecture using the Ethereum Blockchain network is described in the next section.

4.3.4 Implementation

In this section, we discuss the IoT-oriented implementation of the BCB-X3DH protocol. Similar to the first implementation [74], the development is based on Ethereum, but the same design can be implemented with private Blockchain networks if the number of participating nodes is sufficient to guarantee trustiness of the system, data immutability and resistance to distributed attacks.

For this implementation, the only requirement for the IoT device is to be able to perform secure HTTP requests.

The keys generation is processed on the Smart Contract in the Blockchain network; the Public Key is stored in the Blockchain and the Private key is sent to the requester and never stored on the network. This approach well suits the requirements of human-to-human, human-to-machine and machine-to-machine secure communication channels over the Cloud, Edge and IoT. In fact, every type of client, including the low-resource device can gain benefit from the BCB-X3DH protocol as the crypto-computation is not required. On the other hand, communication with the node processing the Smart Contract needs to be end-to-end secured otherwise it is vulnerable to Man-in-the-Middle attack.

The Smart Contract is written in Solidity programming language and makes use of Elliptic Curve library [127] for mathematical calculations.

The main $X3DH$ data structure of the Smart Contract is shown in Table 4.4.

Table 4.4: X3DH's data structure of the implemented Smart Contract.

Attribute	Data Type	Description
IK	bytes32	Device's Identity Key
PK	bytes32	Device's Prekey
EK	bytes32	Device's Ephemeral Key (Optional)
OPK	bytes32	Device's One-time Prekey (Optional)

Like every programming language, when developing a Smart Contract with Solidity it is important to consider the computational complexity because it has an immediate effect on the cost to pay to the Blockchain network.

To design the Smart Contract presented in this paper, the following considerations have been taken:

- **Contract halt**: the contract must not contain any deadlock for all entities waiting for some status change.
- **Infinite loops**: a Smart Contract must provide an upper bound value for a transaction calculation to be set as the Gas limit to avoid infinite loops and contract halt.
- **Data structure**: the Data structure used to store the attributes of the contract must use a proper data type, because waste in memory space has a direct result in the transaction cost.
- **Iterations**: one of the most expensive operations in a Smart Contract is looping through an array. This should be avoided when possible and this computation should be performed on the client-side.

Due to the intrinsic complexity to perform elliptic curve cryptographic operations, the Smart Contract proposed in this paper has an average execution cost of 1,100,143 Gas.

To calculate the Secret Key (SK) to be used for secure communication with BCB-X3DH protocol, based on the scheme in Figure 4.17, a Device A requests its Identity and Prekeys generation by invoking an $external public function$ in the Ethereum Blockchain network. Any Device B that wants to start a secure communication with Device A, requires to know Device A's ID and have to provide its own keys. This can be done by generating these via the Smart Contract (as the same approach of Device A) or, if Device B has enough computational power, can generate its own key and send these to the Smart Contract only for storing them, saving a considerable amount of Ethereum as in this process no calculation is required by the Smart Contract.

It is important to note that each Device pays the Blockchain transactions only when it is required to store data (i.e. storing Public keys), and does not pay any fee for data retrieval.

Once the Secret Key is received by both participants, it can be used to secure any communication over untrusted channels.

The pseudo-code about the implementation of the Smart Contract is provided in Algorithm 4.

Algorithm 4 BCB-X3DH algorithm

Class $X3DH$
method $generateKey(deviceId, entropy)$
  $ec = EllipticCurve$
  if $isSet(entropy)$ **then**
    $random \leftarrow keccak256(entropy)$
    $privKey \leftarrow keccak256(random)$
  else
    $temp \leftarrow keccak256(ec.getRand(), ec.getRand())$
    $privKey \leftarrow keccak256(ec.getRand(), temp), ec.getRand())$
  end if
  $x \leftarrow \emptyset$
  $y \leftarrow \emptyset$
  $x, y \leftarrow ec.publicKey(privKey)$
  $device[deviceId].x \leftarrow x$
  $device[deviceId].y \leftarrow y$
  return $privKey, x, y$

4.3.5 Experiments

In this Section, experiments are presented and discussed to verify if low-resource IoT device can benefit from the BCB-X3DH protocol, with focus on the following analyses:

execution time to perform key generation, locally and via the Smart Contract, finally, CPU usage % for both solutions are compared. Specifically, the following implementations are considered:

- **Traditional X3DH**: Key pairs (identity key IK_B, signed prekey SPK_B and one-time prekey OPK_B) are generated locally for both Device A and Device B and then generates the Secret Key accordingly to X3DH protocol;
- **BCB-X3DH**: Device A's key pairs are generated remotely via the Smart Contract. Time-to-mine and Ethereum (ETH) cryptocoin currency transaction cost are subject to Ethereum network traffic, and the more time is required to mine the higher is the cost to be paid to mine a transaction. Device B interrogates the Blockchain network to retrieve Device A's key, the Smart Contract generates key pairs for Device A too and then calculate the Secret Key remotely.

The system prototype assessment was conducted analyzing the total execution time required to complete both X3DH and BCB-X3DH processes. The device used for testing is a Raspberry Pi model 4 equipped with a Quad-core Cortex-A72 (ARM v8) 64-bit SoC @ 1.5GHz and 4 GB RAM running Raspbian OS. The remote server where a Blockchain node is running is a Intel® Xeon® E3-12xx v2 @ 2.7GHz, 4 core CPU, 8 GB RAM running Ubuntu Server 18.04. Tests have been executed for 100, 200 and 300 read and write operations, considering 95% confidence intervals and the average values. All tests have been performed using Ropsten Ethereum public Blockchain test network, leveraging 300+ available nodes with a real server load status. It must be considered that the Ethereum Blockchain Ropsten environment is based on Proof of Work (PoW) consensus protocol which makes it difficult to obtain scalability and system speed. A summary of testbed characteristics is reported in Table 4.5, experiments setups is reported in Table 4.6.

Table 4.5: Testbed characteristics.

Parameter	Server	Raspberry
CPU	Intel® Xeon® E3-12xx v2 @ 2.7GHz, 4 core CPU	Quad core Cortex-A72 (ARM v8) 64-bit SoC @ 1.5GHz
RAM	8GB	4GB
OS	Ubuntu Server 18.04	Raspbian

Table 4.6: Summary of experiments performed.

Parameter	Value
Test executed for each scenario	[100, 200, 300]
Confidence interval	95%
Gas price (Gwei)	45
Gas used by transaction (generation and storing)	1,092,435
Gas used by transaction (storing only)	121,410
Average cost per transaction (ETH) (generation and storing)	0.049
Average cost per transaction (ETH) (storing only)	0.005

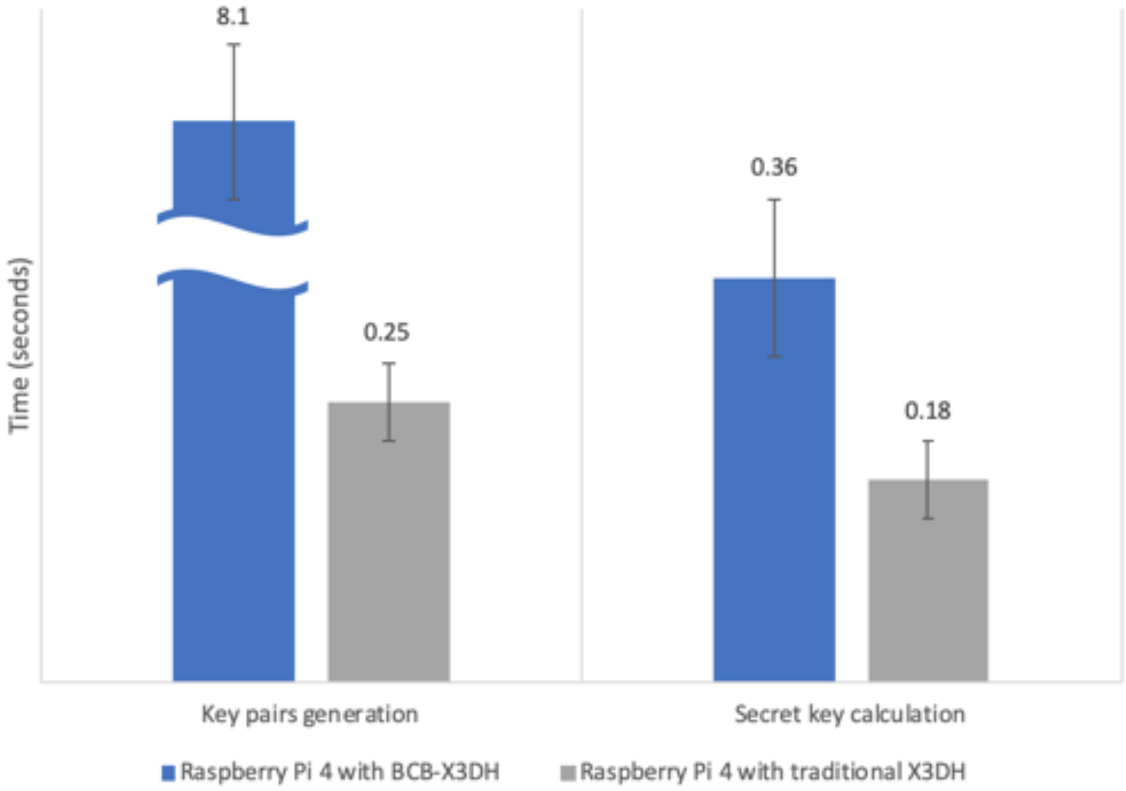


Figure 4.18: Response time comparison on Raspberry Pi 4 between Traditional X3DH and BCB-X3DH implementations for a key exchange process.

Figure 4.18 shows the execution time difference expressed in seconds between the two system implementations considering a complete key exchange process. On the x-axis, it is reported the different steps required to perform the key exchange workflow, whereas on the y-axis it is reported the processing time expressed in seconds. It is possible to observe that the time required to perform Key pairs generation heavily deviates in the two implementations,

changing from 0.25 seconds up to 8 seconds. This is due to the mining time required to store the data in the Blockchain transaction. The time necessary to calculate the Secret Key presents a similar behaviour as compared to the first step. In the traditional X3DH approach, the Public keys of the second device are downloaded from a remote server and the Secret Key is calculated in 0.18 seconds, whereas the same steps performed by the Smart Contract require 0.36 seconds.

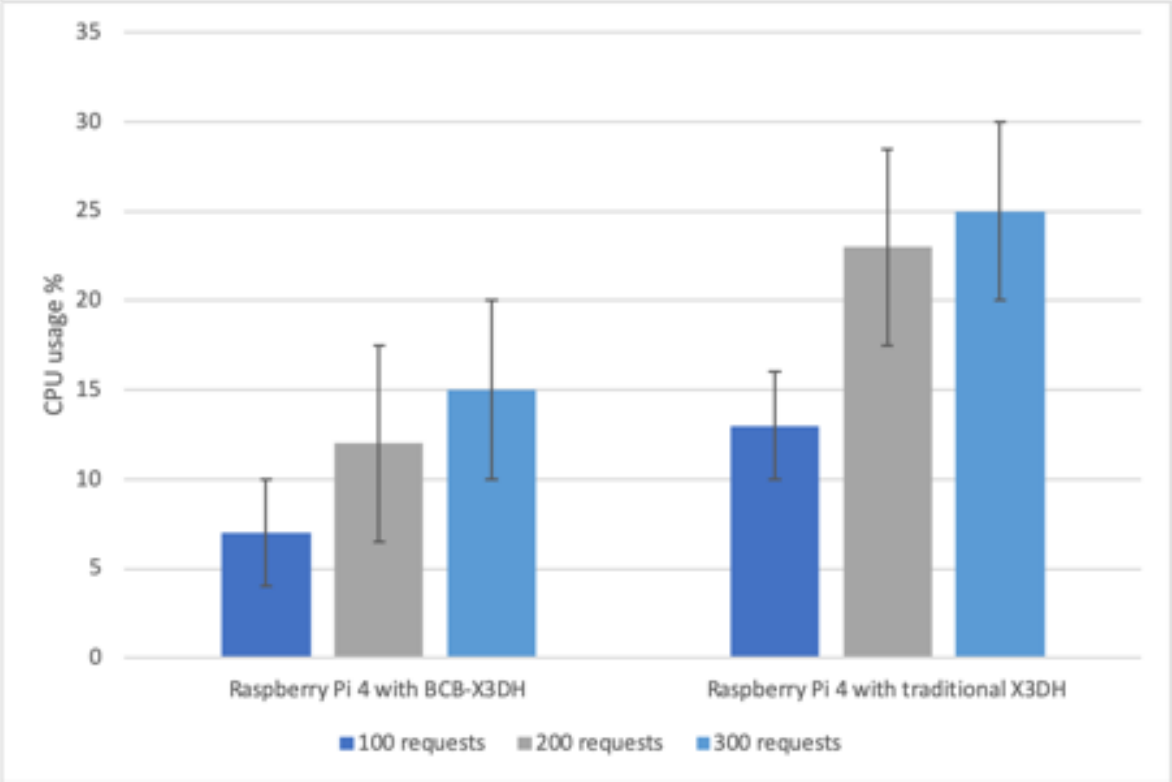


Figure 4.19: CPU usage % comparison on a Raspberry Pi mod 4 between BCB-X3DH and traditional X3DH implementations to generate and calculate the Secret Key.

Figure 4.19 shows the CPU usage % difference between the two system implementations for a key exchange process. On the x-axis, it is reported the differences to perform 100, 200 and 300 requests, whereas on the y-axis it is reported the CPU usage %. It is possible to observe that fewer CPU % is required to perform the key exchange with the BCB-X3DH implementation, resulting in a less overloaded system, thus resulting in efficient energy consumption and longer duration in case of battery-powered systems.

4.3.6 Some remarks

This Section progresses further the study to implement the improved version of the Extended Triple Diffie-Hellman protocol based on Blockchain considering an IoT scenario.

The Smart Contract developed has the capability to generate the Key Pairs required in the X3DH protocol and to store the public keys only in a dedicated data structure that can be retrieved by any device that wants to perform the Secret Key exchange.

This research specifically focuses on the implementation of the BCB-X3DH protocol aimed at securing communication between devices. In particular, the Ethereum public Blockchain network is adopted to consider a large number of nodes (a greater number of nodes means a greater degree of security) but, in enterprise environments, the same design can be implemented with private Blockchain networks if the number of participating nodes is sufficient to guarantee trustiness of the system, data immutability and anti-tampering and to resist to distributed attacks.

Experimental results highlighted that the overhead introduced by Blockchain in the BCB-X3DH implementation in terms of execution time can be acceptable, with the consideration that the computation for the key generation is moved from the IoT to the Blockchain, hence the energy required from IoT is reduced. The solution has been validated with a Raspberry Pi mod 4 and the CPU usage has been monitored. From the results obtained, the proposed BCB-X3DH approach is efficient in terms of energy consumption, resulting in a longer duration for the battery-powered IoT device. The BCB-X3DH protocol, decentralizing part of the secure communication channel maintenance to a Blockchain network, can lighten the workload of IoT devices, hence improving their performance.

In future developments, the BCB-X3DH protocol will be tested in private Blockchain networks such as Hyperledger Fabric and Sawtooth, to find the best trade-off between data privacy and system efficiency, and an energy analysis will be performed to validate if this solution is the best alternative to efficiently secure communications between devices.

4.4 An Energy efficiency analysis of the BCB-X3DH protocol for IoT

The recent advancements in miniaturized smart data collecting devices pushed the need of securing communications between people and devices. Traditional approaches based on key exchange protocol can't be performed by resource-constrained embedded devices, and a novel approach, based on a robust decentralization of the eXtended Triple Diffie-Hellman (X3DH) protocol, has been proposed, namely the BlockChain-Based X3DH (BCB-X3DH) protocol. This work progresses the analysis further to fit a generic Smart City scenario with Edge and IoT nodes, performing intensive analysis on Raspberry Pi 3 model B+ and

Raspberry Pi 4 to validate that the new protocol is not only resistant from well-known distributed attacks but can also be executed by miniaturized hardware with benefits in terms of resources, energy consumption and battery life-cycle.

Today, with the increased development of devices connected over the Internet, securing communications between diverse parties (i.e., individuals, devices, etc.) is a must. Any Information Technology (IT) system must guarantee that communications exchanged between parties are kept secret, that parties are trustworthy and legitimate, and that messages are sent exclusively to the intended recipient. Since Caesar's code, message encryption has been suggested and tested, but Whitfield Diffie and Martin Hellman [103] established a competent key-exchange convention based on arithmetic operations in 1976. It was very impossible to break with the processing capacity of the time using discrete logarithms, therefore this method has been used for over half a century.

Unfortunately, this protocol has security limitations due to the centralized trust model's single point of failure (SPoF), making it vulnerable to Distributed Denial of Service (DDOS) and other cyber-attacks. Various ways exist to guard against this, resulting in increased costs for maintaining a security framework [128].

To transmit messages and data, both Cloud Computing and resource-constrained devices require a secure communications connection. Several encryption methods have been developed over time to render brute force attacks useless [116], however they all rely on a secret seed that must not be revealed to anybody or any device other than the aiming parties.

Previously, an alternative to the X3DH protocol has proposed with its foundation on Blockchain technology. The Blockchain-Based X3DH (BCB-X3DH) protocol aims at completely reducing the vulnerability of the SPoF and getting a secure framework with a negligible cost [74].

In summary, the BCB-X3DH protocol integrates the key benefits of Smart Contract such as immutability and data non-repudiation, and the centralized server utilized to perform the agreement process is supplanted by a distributed Blockchain network.

The Smart Contract developed can generate the public/private key pair, saving the public key on the network and sending the encrypted private key to the requester, without storing it on the Blockchain. This is required for all the instances where a resource-constrained device requires to encrypt and share data but is not able to generate the key pair.

Progressing from the original study, the same protocol has been developed and tested for these domains where low-resource and battery constrained devices were present [129].

Furthermore, in the Internet of Things (IoT) era, including billions of embedded devices

with limited storage and processing resources, the secondary objective of our BCB-X3DH protocol is to lighten such devices from public key cryptography tasks guaranteeing energy cost efficiency.

The following is a summary of the research's contribution:

- a review of processes and systems for securing IoT communications was carried out, with a focus on Elliptic Curve-based encryption for devices with limited computing capabilities;
- a study of distributed cyber-attacks was conducted, with a focus on communication disruption or service inaccessibility;
- the BCB-X3DH protocol is designed to ensure secure and efficient secret key generation and exchange in IoT and low-resource devices;
- an energy study was conducted to show how this approach is the most efficient way to encrypt communications between devices.

4.4.1 Background and Related Work

The Blockchain technology is considered in both academic and industrial fields as a possible solution to resolve most of the security threats and privacy concerns in a heterogeneous application environment [1], and it ought to resolve many open information access issues [130] thanks to the adoption of Smart Contracts to achieve trustiness, transparency and traceability [71, 131]. In this context, there are many scientific initiatives aimed at demonstrating how different approaches of Blockchain can be implemented to replace the single server architecture and how this configuration can be applied to communication security in many domains, including the IoT context [72].

The decentralized nature of Blockchain permits to use this framework to replace the central server configuration and reduce the risk of DoS attacks, still achieving the reliability typical of Smart Contracts [109].

Protecting human-to-human communication is vital to guarantee the privacy and avoid eavesdropping from unwanted third parties, and one of the first examples in history can be related to the Caesar cipher, based on a simple character shift of n position shared between the two parties.

Since the computer era, the need for new encryption techniques has raised the research of more complex mathematical approaches, and these can be classified into two groups:

symmetric and asymmetric encryption.

A simple technique can be used to secure communication through a combination of symmetric and asymmetric encryption techniques. A generated symmetric key can be used to encrypt a text and then concatenate the key itself and cipher the entire block with a public key generated by the receiver. Only the end-user can decrypt the text with his private key and he can split the message obtaining the symmetric key, decrypting the final message [120].

This technique is also applied by cyber-criminals during ransomware attacks: with a symmetric algorithm, more performing in terms of execution time, the data of the victim is encrypted; then the key is encrypted with the public key of the attacker and then destroyed, and only the malevolent user can recover the password after the payment of a ransom [132].

When buying a new home and smart devices, connecting them with the home wifi, we underestimate the potential threat of such actions, as these devices require safer [119] and more efficient [118] communication protocols.

Finding the best trade-off between security, applicability, and energy efficiency is not an easy task, and each protocol requires some compromise.

Different approaches based on Elliptic Curve-based cryptography offer better performance in terms of resource consumption to protect communications, especially these are indicated for resource-constrained IoT devices compared to RSA cryptography [123].

It has been already proved that Blockchain and Smart Contracts can be applied in several contexts to achieve system and network robustness, maintaining a high level of trustiness to the system thanks to the transparency of the implementation details [133, 124, 125].

Other approaches are based on Physically Unclonable Function (PUF) to generate a digital fingerprint used as a unique identifier for a device [121]. However such solutions offer weak protection against the SPoF scenario, and the server responsible to deliver messages based on this identity key can be compromised and messages could be sent to unintended nodes.

Compared to the previous mentioned research activities, this work proves, through a practical implementation, how Blockchain technology can prevent the limitation of the SPoF and reduce the cyber threats, offering a trusted distributed service for IoT and low-resource devices.

4.4.2 Motivation

Two important limitations have been studied in this research which involves the systems that implement the X3DH protocol: i) a weak defense against distributed attacks, and ii) resource-efficiency for battery-powered devices. This section provides a detailed description

of them.

Weak defense against distributed attacks

Services infrastructure based on central architecture is vulnerable to many network security threats which can cause partial or complete service unavailability, being this a SPoF. Some of the most common attacks used by cyber-criminals are Distributed Denial of Service (DDoS) [67] and Man-in-the-middle (MITM) [66] attacks.

A MITM attack consists of the attacker sniffing communications between the two parties, aiming at altering the communication content to a genuine receiver, declining messages to one of the users or impersonating one of the victims, interrupting the message delivery, and the entire communication.

A Denial-of-service attack has the objective to disrupt a service from a server, aiming at getting privileged access to resources or to make the entire end point unusable. This can be done by saturating the network capacity of a service sending a huge amount of data (recent attacks have reached a throughput of 3,47 terabit per second (Tbps)), making the server unable to satisfy all the requests.

In X3DH protocol, the potential victim of such attacks is the server that manages the key exchange between parties, which is also responsible to store the public keys of all users. If an attacker takes control of the system, he can alter the data stored making the key exchange process unusable.

The arguments discussed in this paper show that Blockchain might be a viable, dependable, and safe alternative to traditional client-server architecture, including inherent security characteristics that would be difficult to develop and maintain with a traditional implementation.

Resource-efficiency for battery-powered devices

The key exchange process is one of the building pillars of network security, but many small and resource-constrained IoT devices suffer from the possibility to spend enough resources to spend on key generation.

In the X3DH protocol, both participants calculate the public/private key pair and send the public part to the central trusted server for the execution of the key exchange algorithm. However, this is not possible for low-cost embedded devices that typically have very limited memory, processing, and energy resources. This limitation has been analyzed and

demonstrated how X3DH protocol can be executed entirely on-chain, making it feasible for low-resource devices too [13].

Furthermore, small embedded devices, which battery runs on solar energy, might not spend resources for a key generation as their main objective is to collect data from sensors and send them.

This research compares the results obtained for constrained devices in a Smart City context, carrying out an energy analysis in terms of resources, battery capacity, and device lifecycle to validate the applicability of Blockchain-based X3DH solutions for low-resources devices.

4.4.3 System Design

In the X3DH implementation, the server responsible for the key exchange process must be trusted and the keys of the counterparty should be sent only to the intended receiver. An attacker who manages to control the server makes the system unusable, invalidating the entire security process.

Starting from the X3DH diagram presented in Section 4.3.3, the battery-efficient *BCB-X3DH* approach has been explored to solve the drawback of the central server architecture and the limitation for low-resource and battery-powered devices.

The Blockchain network replaces the central server to facilitate the key exchange process through the use of a Smart Contract. Any User or a generic IoT Device can seamlessly request a *SK* generation by submitting a request to the Blockchain network which will generate the Public Keys bundle for the requester, storing it to the Blockchain to be publicly available. When IoT Device B wants to share a *SK* to secure communication with IoT Device A, it needs to provide a pre-shared identifier of the requested partner. If the Public bundle is available for IoT Device A, the Smart Contract combines the available Keys of both the participants to generate the *SK*. If the bundle is not available because IoT Device A has not registered itself to the Blockchain network yet, the Smart Contract provides a ticket that is unique for the request that can be used at a later moment to provide the *SK* when IoT Device A's Public keys are available. When both Devices have received the *SK*, secure encrypted communication can start.

The Blockchain network operates as a trusted server and immutably stores data. The users participating in the exchange process are Blockchain nodes.

Any full node participating in the mining process can invoke the key generation requests and store data into the network. IoT nodes participate in the network as light nodes without

altering the system configuration. Of course, it must be guaranteed that the network has a reasonable number of full Blockchain nodes with mining capability to ensure the validity of the network.

The Smart Contract implemented in BCB-X3DH is designed to generate the public/private key pair, to store the public part on the Blockchain network, and encrypt the private part to be sent to the requester via a secure communication channel (for obvious security and privacy reasons, the private key is never stored on the Blockchain).

This approach makes the protocol congenial to low-resource devices to secure human-to-machine and machine-to-machine communications.

The details of the system implementation, based on the Ethereum Public Blockchain network, are discussed in Section 4.4.4.

4.4.4 Implementation

The implementation of the BCB-X3DH protocol, specifically oriented for IoT and small devices scenarios, is based on the Ethereum network, but its design can be used for private Blockchain frameworks as well if the private network has enough nodes to resist distributed cyber-attacks. This approach is meant for IoT and low-resource devices and the only requirement is that they must be able to perform secure HTTP requests. The components used to implement this solution are all based on Docker [134] containers, to ensure the solution portability through the virtualization paradigm.

The client communicates with the Blockchain through Infura [83], ensuring a simple communication pattern for low-resource devices. The Smart Contract must store the main user data and identification to perform the key exchange process. The complexity of the Smart Contract can be easily related to its gas consumption, which depends on the function complexity and the size of the data to be stored on the network.

To begin the key exchange process between users or devices A and B, they only need to know counter-party ID. If a device has enough resources, it can autonomously generate its key pair, or leave this computation to the Smart Contract. Then the Smart Contract needs to be invoked to store the public keys of both participants and to perform the agreement part. It has to be noted that each user or device pays a fee with Ethereum cryptocoin (ETH) only to store its public key into the Blockchain network, and every data retrieval, i.e. to get counter-party public key, has no cost. In other words, a device can exchange a secure key with unlimited devices paying only the first public key submission.

When the Smart Contract completes its execution and the Secret Key is received by both

users or devices, it can be used to secure any communication over untrusted channels.

4.4.5 Experiments

The experiments discussed in this section aim at verifying the efficiency, in terms of resources, for a Smart City scenario based on Blockchain technology where multiple low-resource devices need to communicate over a secure channel. The tests focus on the comparison of the power consumed during the generation of the key pairs locally and on-chain through the Blockchain-based X3DH solution proposed in this paper.

The two implementations compared are detailed as follows: in the classic X3DH approach based on a single third-party server, the identity key IK_a, the signed prekey SPK_a, the one-time prekey OPK_a and, finally, the Secret Key SK, are generated exploiting the IoT Device resources; in the Blockchain-based X3DH solution proposed, the key pairs are generated on the Blockchain network through the Smart Contract and the IoT Device's computation effort is negligible, as it only requires to perform an HTTP request.

The energy evaluation has been performed using a Raspberry Pi 4 as a control unit connected to the Texas Instruments INA219 sensor through I2C protocol. A sample schema of the circuit used is depicted in Figure 4.20. This protocol is a synchronous bus that requires two serial communication lines: the Serial DAta (SDA) communicates the data itself and the Serial CLock (SCL) for the clock. A part from this two connection, a reference line GND and a power connection line V must be used. The INA219 sensor measures the voltage difference before and after the shunt, and along with the value of the resistor shunt (0.1 Ω) it determines the current flow in amps.

Performance evaluation focuses on the time spent to complete the key exchange process in both the implementations. All tests are performed using two local IoT devices and a remote server running a full Blockchain node. The IoT devices used are a Raspberry Pi 4 running Raspbian OS with a Quad-core Cortex-A72 64-bit SoC at 1.5GHz and 4 GB RAM and a Raspberry Pi 3 Model B+ equipped with a Quad-ore 1.2GHz and 1GB RAM. The remote server used to run the Ethereum Blockchain node is a Quad-core Intel® Xeon® at 2.7GHz running Ubuntu Server 18.04 equipped with 8 GB RAM. To evaluate both the approaches, 10 and 100 reads and write tasks are considered at 95% confidence intervals and the average values are presented. To simulate a real server load status with more than 300 active nodes, Ropsten has been chosen as the official Ethereum public network for testing purposes.

In Figure 4.21, the different average bus current required to perform the Key Exchange process in the two implementations is shown. The x-axis reports the execution, both locally

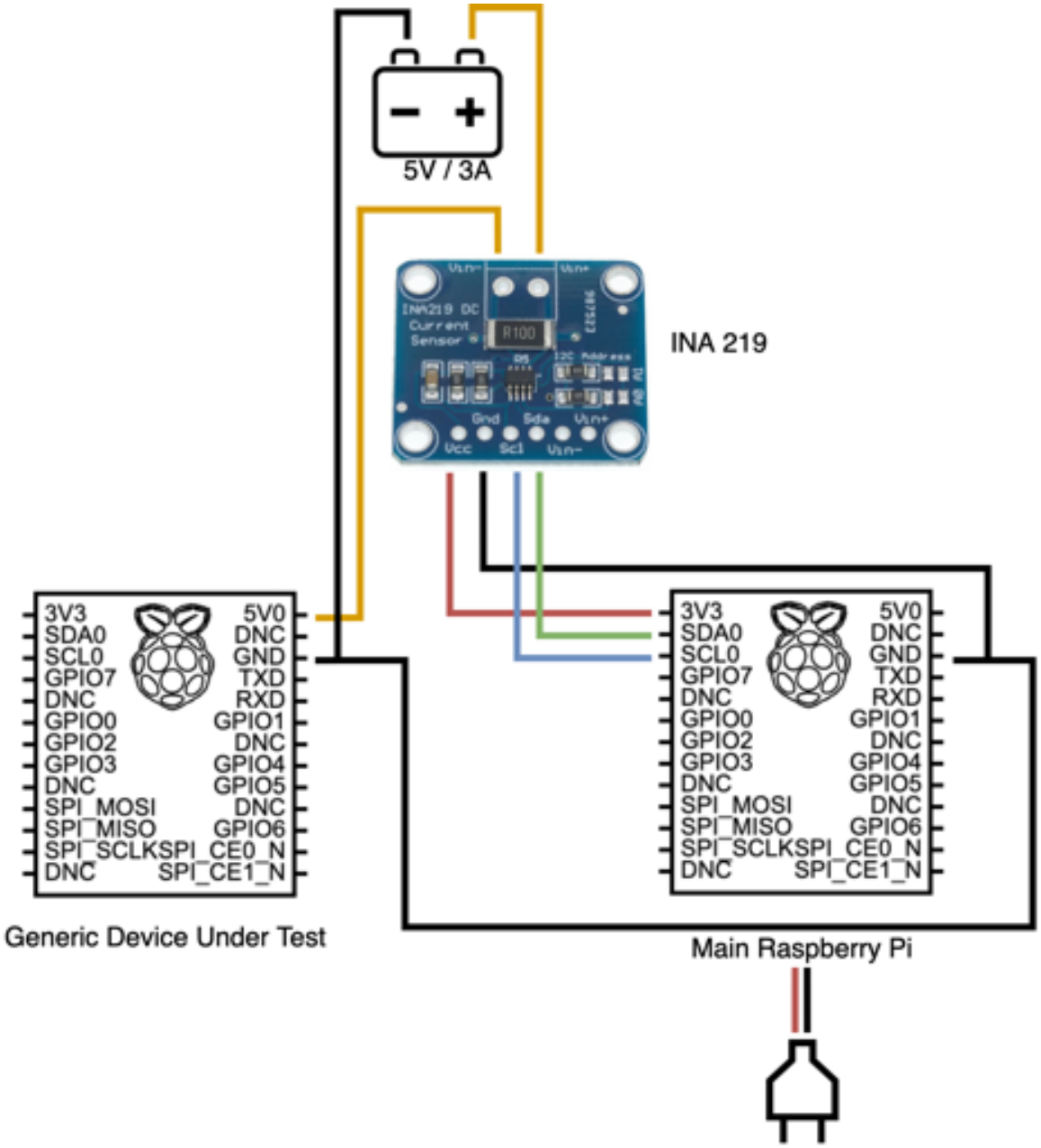


Figure 4.20: Schema of the Texas Instruments INA219 sensor with Raspberry Pi 4 as a control unit.

and remotely via the Smart Contract, of 10 complete Key Exchange processes performed on a Raspberry Pi 4 and a Raspberry Pi 3. The y-axis reports the current intensity expressed in milliamps (mA). From the graph, it is possible to appreciate that the current intensity is always lower for the execution on the Raspberry Pi 3 as the resource consumed is always lower at idle state and significantly less for the remote execution based on the Ethereum Blockchain network for both the IoT Devices.

Similar behavior can be appreciated in Figure 4.22 where the different average power consumption required to perform the Key Exchange process in the two implementations is

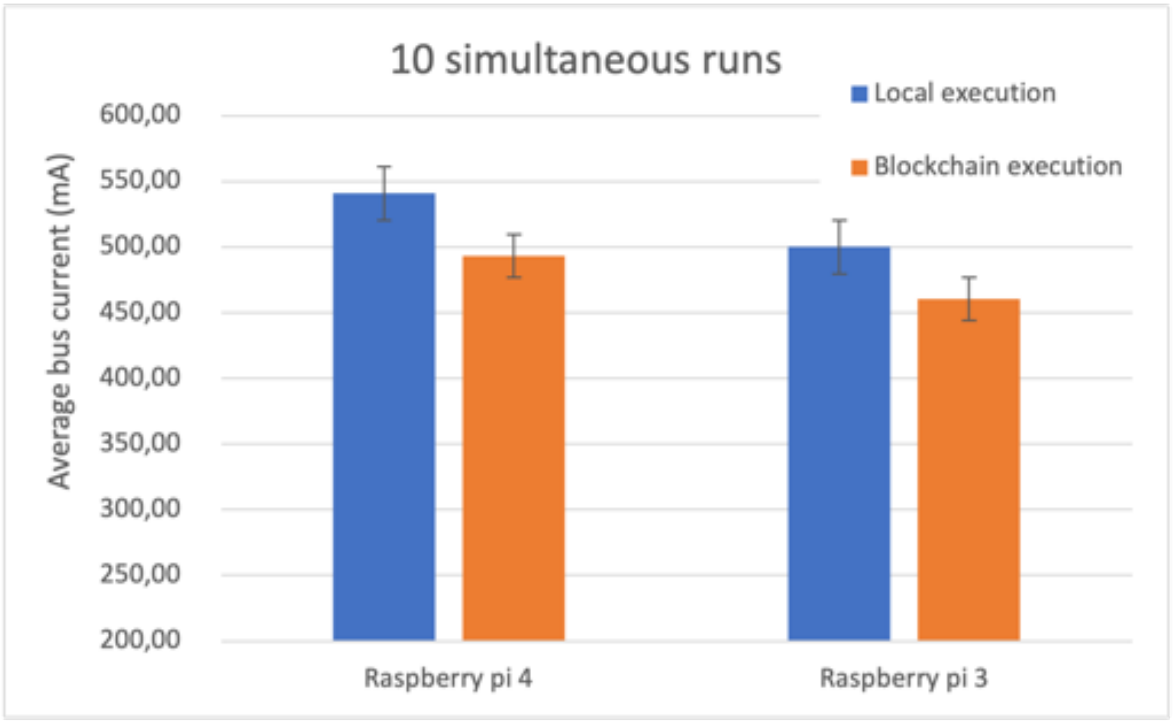


Figure 4.21: Average bus current expressed in mA comparing the current intensity for key generation task performed locally and via the Smart Contract for different models of Raspberry Pi for 10 simultaneous runs.

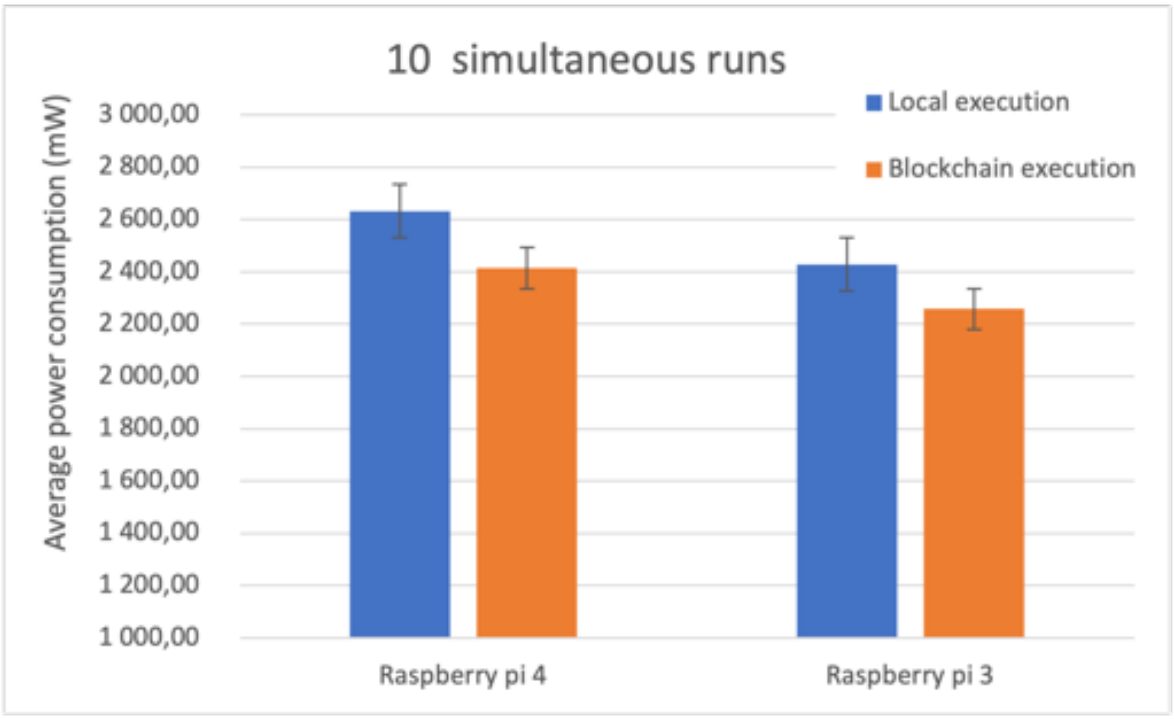


Figure 4.22: Average power consumption expressed in mW comparing the power consumed for key generation task performed locally and via the Smart Contract for different models of Raspberry Pi for 10 simultaneous runs.

shown. The x-axis indicates the execution of 10 complete Key Exchange processes performed on the two mentioned IoT Devices. The y-axis reports the power consumption expressed in milliwatts (mW). The graph shows a similar behavior, where the resources consumed on the Raspberry Pi 3 are always less compared to the other IoT devices, especially for what concerns the remote execution via Smart Contract.

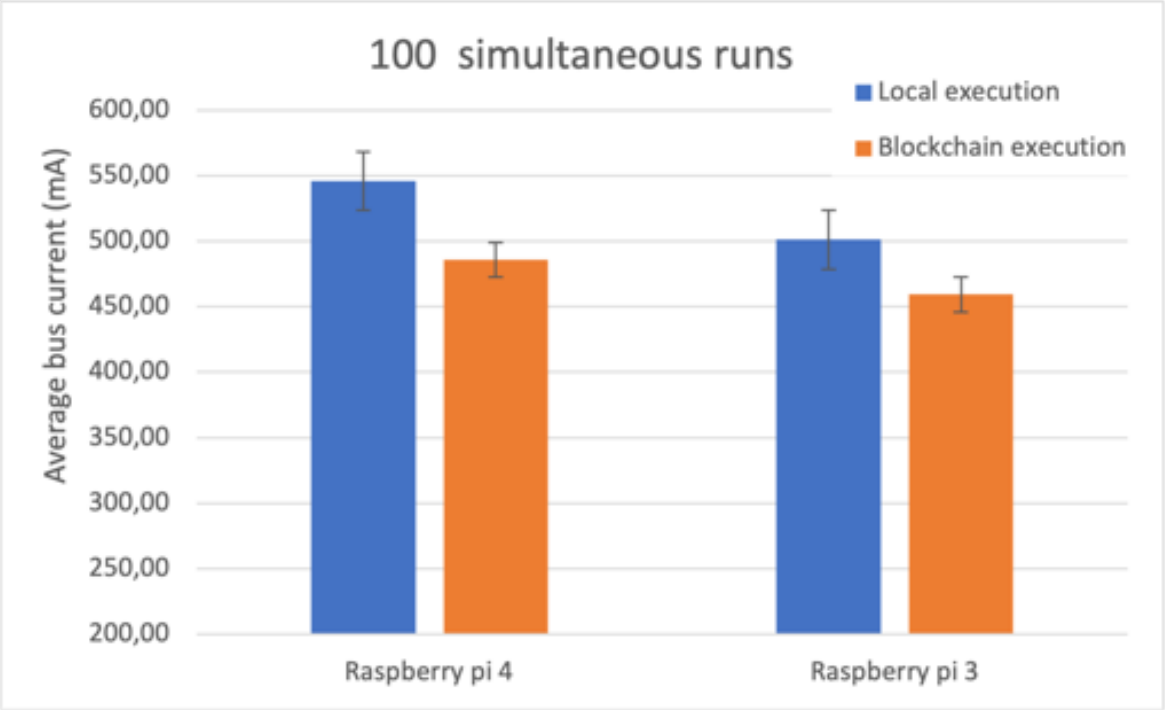


Figure 4.23: Average bus current expressed in mA comparing the current intensity for key generation task performed locally and via the Smart Contract for different models of Raspberry Pi for 100 simultaneous runs.

The same test performed for 100 simultaneous runs presents a similar behavior: Figure 4.23 shows the different average bus current required by the two Raspberry Pi models. The x-axis reports the execution of 100 complete Secret Key generations performed on a Raspberry Pi 4 and a Raspberry Pi 3. The y-axis reports the current intensity expressed in milliamps (mA).

For what concerns the average power consumption, results can be seen in Figure 4.24. The x-axis reports the execution of 100 complete Key Exchange processes. The y-axis reports the power consumption expressed in milliwatts (mW).

From the above results, it can be affirmed that the remote Ethereum Blockchain-based approach consumes less power resulting in an immediate power energy saving, allowing resource-constrained battery-powered IoT Devices to increase the running time.

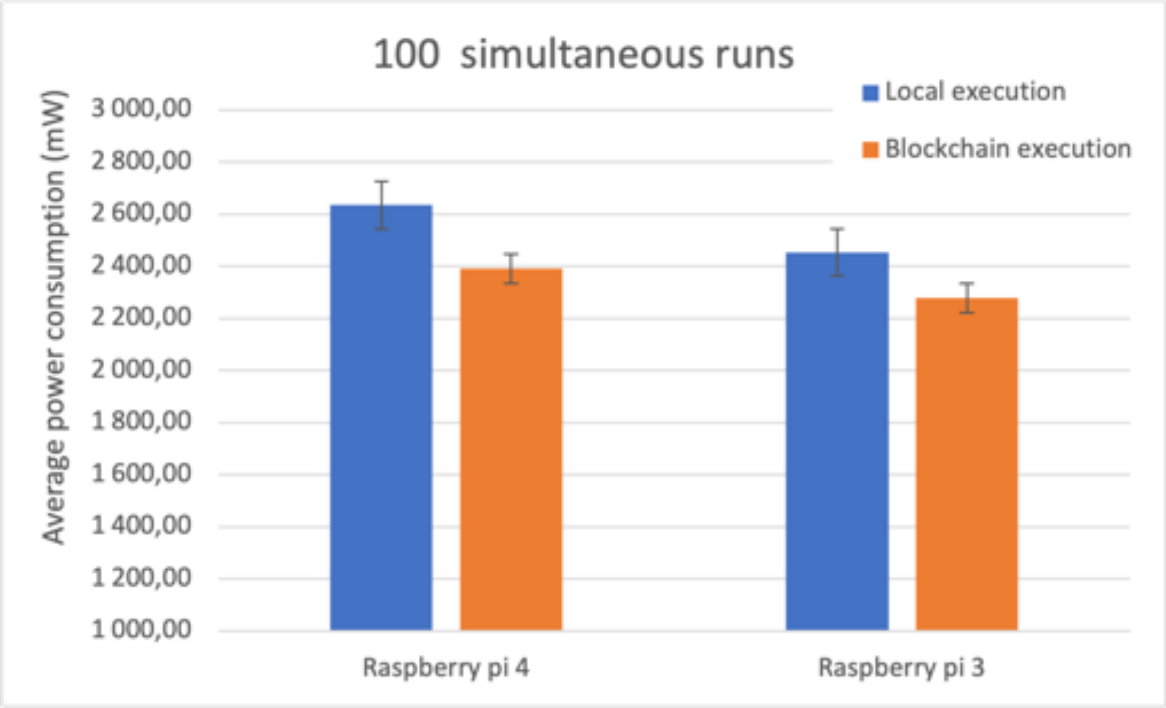


Figure 4.24: Average power consumption expressed in mW comparing the power consumed for key generation task performed locally and via the Smart Contract for different models of Raspberry Pi for 100 simultaneous runs.

4.4.6 Some remarks

This Section analyses the energy consumption on IoT Devices when performing a complete eXtended Triple Diffie-Hellman Key Exchange protocol to share a Secret Key to secure communication over untrusted channels. The two approaches analyzed are based on local execution and a remote Ethereum Blockchain node running on a dedicated Ubuntu Server. This latter approach, allows the IoT Device to obtain a secure Secret Key consuming the energy required to perform an HTTP request only, leaving all the crypto-computation on the remote server. The outcome of this is an immediate resource-saving of the IoT Device which can last longer in the case of battery-powered rechargeable through solar panels scenario.

4.5 Overall consideration

This Chapter analyses how Blockchain technology can be implemented to increase security in public connections. In particular, Section 4.1 proposes a complete re-engineering approach based on independent microservices to perform the 2FA virtually in every context. The MBB-OTP protocol, in fact, can offer significant security improvements in every scenario where digital identity needs to be verified.

Sections 4.2, 4.3 and 4.4 focus on a new implementation of the Extended Triple Diffie-Hellman (X3DH) protocol, thought to reduce the risk of MiTM and DoS attacks, with a negligible upfront investment. Experimental results demonstrate how the BCB-X3DH protocol can ensure integrity and accountability during the key exchange process for human-to-human interaction, machine-to-machine communication, and even for low resource devices because it is meant to run on everything that can perform a simple HTTPS request on both private and public Blockchain networks. In the latter case, the cost to be paid for transaction mining is trivial considering the the cost to setup a secure infrastructure.

CHAPTER 5

Improving the healthcare sector with the Blockchain

In recent years, numerous research studies have been conducted in healthcare field with particular attention to the application of Blockchain technologies [135].

The characteristics of the Blockchain can offer a unique solution for healthcare assistance thanks to wearable devices and IoT (Internet of Things). The healthcare sector has a growing demand for Blockchain developments and the traditional industry is actively exploring new avenues for using the Blockchain to meet its critical needs.

The main functionality of the Blockchain applied to health data is certainly the immutability in order to guarantee the clinical history documentation and the results of clinical studies.

As it has been discussed in [72], Blockchain technologies have been increasingly recognized as a tool to address existing information access problems. It is in fact possible to improve access to health services by using the Blockchain technologies to achieve greater transparency, security and privacy, traceability and efficiency.

5.1 Use of DSS to improve prediction of clinical analyses

Several cases are reported every year where the prescribed therapy results incompatible with the patient's medical history, leading to worsening of clinical condition or death. Some technologies and processes to prevent this misbehaviour already exist, but a concrete solution is not available in hospitals yet. This research presents a Decision Support System (DSS) that can be easily integrated into a typical health workflow at hospitals and provides feedback

on the possible prescription of drugs at a patient with specific diseases. The DSS is based on a Big Data analysis algorithm able to check drugs and diseases relationships and detect possible failures in drugs prescriptions. A prototype of the proposed solution is developed, implementing the DSS system and setting up the necessary Big Data management tools for the effective adoption of the DSS system. Performance evaluations to assess the efficacy and the response time of the DSS algorithm are discussed.

The demographic growth of the last century combined with increased life expectancy and reduction of specialized doctors caused access to proper medical treatment a major concern of the last decade. For example, several cases are reported every year where the prescribed therapy results incompatible with the patient's medical history, leading to worsening of clinical condition or death [136]. Some technologies and processes to prevent this misbehaviour already exist, but a concrete solution is not available in hospitals yet.

This research presents a Decision Support System (DSS) that can overcome these issues. It provides feedback on the prescription of drugs at a patient with specific diseases through a Big Data analysis algorithm able to check drugs and diseases relationships and detect possible failures in drugs prescriptions. It can be easily integrated into a typical health workflow at hospitals, and, thus, can provide a useful tool for therapy prescription in a hospital centre. To have a concrete implementation of our DSS algorithm, we validate the treatment and isolate possible intolerances considering a new drug-disease weighted relationship approach between diseases, identified by the International Classification of Diseases ICD-9 [137] or ICD-10 [138], and medicine, identified by the Anatomical Therapeutic Chemical Classification System ATC [139] codes.

This research investigates the adoption of the DSS solution in the treatment of a patient from the hospitalization step until his/her dismissal, supporting doctors in the planning of pharmaceutical treatment. The research describes all the necessary workflow steps and how the proposed DSS system can be integrated together with useful technologies and Big Data management tools for the setup of a concrete solution.

A quantitative performance evaluation has been performed to assess the automated medical prescription system running over distributed Big Data oriented technologies. The experimental results show how a distributed approach can be helpful to analyze large data in a short response time.

5.1.1 Background and Related Work

This research starts from medical evidence, analysing a common layer in which hospital admission and worsening of the clinical condition could have been prevented by knowing the risk of concomitant therapies thanks to the use of different technologies applied in healthcare.

To provide an example, in the research of Gupta et al. it is evident how aspirin is associated with the risk of gastrointestinal toxicity, leading to ulceration and bleeding [140]. For patients taking aspirin, who are at risk of gastrointestinal events, concomitant use of proton pump inhibitor (PPI) is recommended, however, it is under-prescribed in these patients.

Furthermore, Uchiyama et al. highlight how aspirin should be used for the prevention of cardiovascular (CV) events by the risk-benefit balance [141]. However, a study was conducted to clarify CV and bleeding events in Japanese aspirin users with a history of CV diseases. As reported in the research of Hwang et al. it is demonstrated how the widespread application of glucocorticoid therapy for the autoimmune disease has led to the concurrent therapy-limiting discovery of many adverse metabolic side effects [142].

In past years, several DSS for healthcare have been proposed but no one of these focuses on therapy prescription. Classification of advantages of DSS in healthcare have been grouped in four sectors: i) learning; ii) patient satisfaction; iii) risk mitigation and iv) efficiency [143], while a different classification is also proposed: i) case-based; ii) experience-based and iii) guideline-based [144]. Also machine learning-based DSS for healthcare have been studied, but focusing only on influenza and influenza-like illnesses, with a system able to predict the future behaviour of flu [145]. On another aspect of healthcare DSS, researches have been made to support doctors during surgery [146], with systems able to create a 3D-model of the aorta by using the computed tomography scan in order to facilitate the preoperative operations.

The logistic-based approach applied to healthcare DSS was performed considering Tunisian Hospitals, with a focus to optimize the stock of drugs reducing costs [147].

Differently from the aforementioned scientific initiatives, that are mainly focused on limited aspects of the healthcare scenario, in this paper we focus on how Big Data technologies can be adopted to prevent worsening of clinical condition, reducing hospital admissions and, potentially, saving lives.

5.1.2 Health Workflow design

The DSS algorithm described in the previous section is useful to identify mismatching in drugs prescriptions. However, the algorithm can be usefully adopted to support the medical staff during the hospitalization period of a patient combined with a healthcare workflow. Specifically, the system's workflow we considered in this paper is as follows:

1. **Hospitalization**: patient reaches the hospital and personal details, date and type of visit are recorded;

2. **Analysis**: patient follows the procedures to ascertain the nature of the disease (e.g. blood tests, clinical examinations, possible Computerized Tomography (CT) scans, RX and laboratory tests) and the results of the analysis are saved on a Cloud storage space inside the hospital managed on a dedicated directory for the patient and visit identification code;

3. **MD evaluation**: doctor analyzes the results of clinical analysis, CT scans, etc. and prepares a report with the therapy to be followed;

4. **DSS execution**: the DSS algorithm automatically evaluates if the indicated therapy can cause undesirable effects due to allergies or intolerances, as well as the therapy in similar clinical cases have caused deaths. If some incompatibility is detected, the DSS system can propose a reaction activity, such, as, for example, the automatic setup of a remote consulting session. However, the action taken by physicians is arbitrary, and they can even decide to ignore the alert taking the responsibility for the patient health;

5. **Drug administration**: the hospitalized patient is constantly monitored by nurses who administer treatment based on therapeutic indications, and each administration is recorded.

The diagram in Fig. 5.1 shows the reference architecture for the management of the above healthcare workflow.

The system is designed as a web portal to manage all the prescriptions and patient's registration. Information on patient diseases, analyses, therapies and so on are stored in an internal Big Data oriented database (e.g., NoSQL) [148], whereas public datasets on drug-disease relationship are accessed to execute the DSS.

To enhance co-operation between hospitals, guaranteeing personal data anonymization, medical doctors belonging to different hospitals can form a virtual health team able to carry out a healthcare workflow in a secure fashion, namely a Federated Hospital Clouds (FHC)

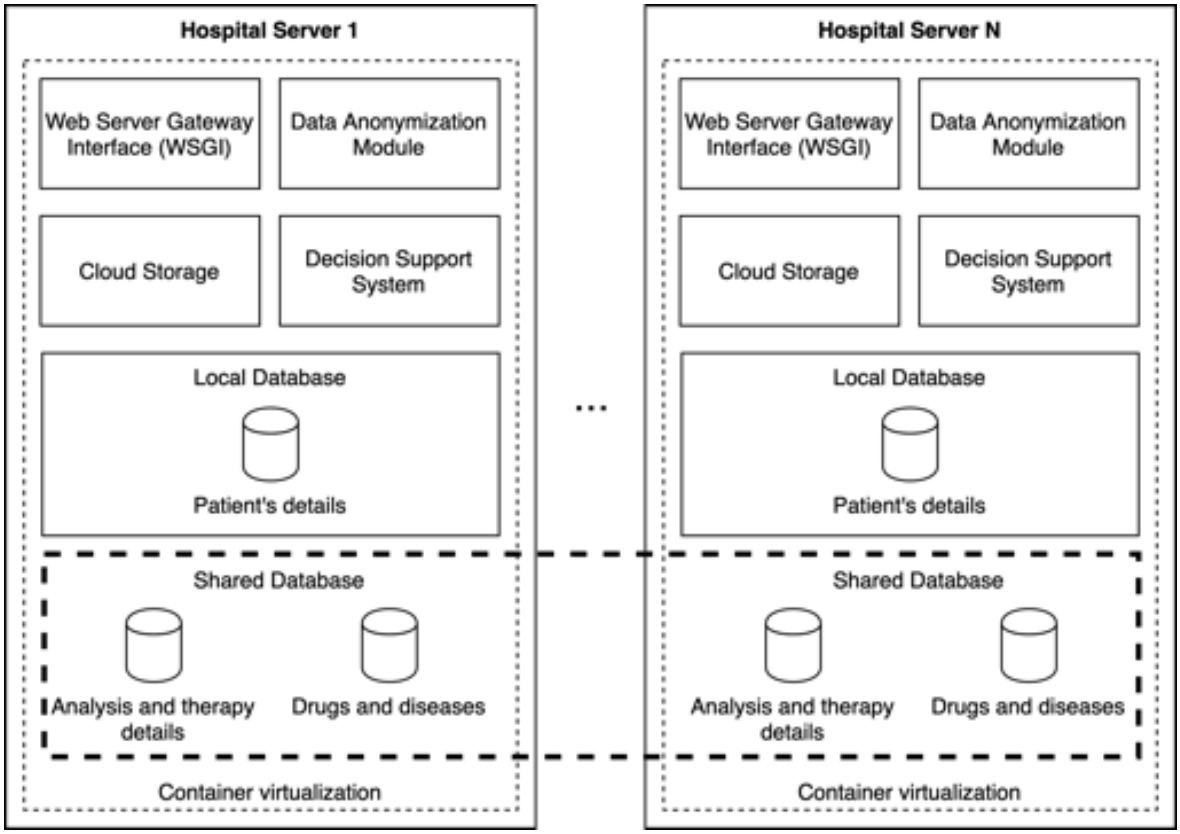


Figure 5.1: Architecture diagram implemented

[149], sharing drugs and diseases and analysis and therapy details databases, while patient's details remain in local hospital server and are never shared.

To develop this system, the following technologies are used:

- **Cloud storage**: to use an open-source and open-architecture file hosting service for archiving and file sharing managed with authorizations
- **NoSQL storage**: to exploit the potential of a document-oriented database to manage patient data and diseases through tags for a fast and efficient search and to store links to files stored in the cloud storage
- **Decision Support System (DSS)**: to search for the optimal therapy by analyzing allergies and intolerances in the history of the patient's medical record

In the next section we provide information on how we implemented the system for therapy management in a hospital centre.

5.1.3 The Decision Support System Algorithm

Big Data analysis can greatly impact service quality in different application domains [150]. Based on the studies conducted in the clinical field, in this paper we propose a novel approach for medical treatments considering as much as possible the adverse conditions and comorbidities between diseases and therapies with concurrent medication and drug administration, to have an autonomous verification step with a shared and growing database supported by Medical Specialist around the World.

This research aims to harmonize health procedures with new technologies to guarantee patients' safety, verifying that the prescribed therapy does not conflict with other diseases or drugs used by the patient and, in the history of the hospital's clinical records, the same therapy did not lead to death.

When physicians visit a patient they can add a prescription treatment indicating the disease to cure in the form of ICD9-ICD10, a medicine identified by ATC and a description with dosage and mode of usage.

Of course, a wrong diagnosis can lead to worsening of clinical condition or death, thus data access control is mandatory to ensure safety.

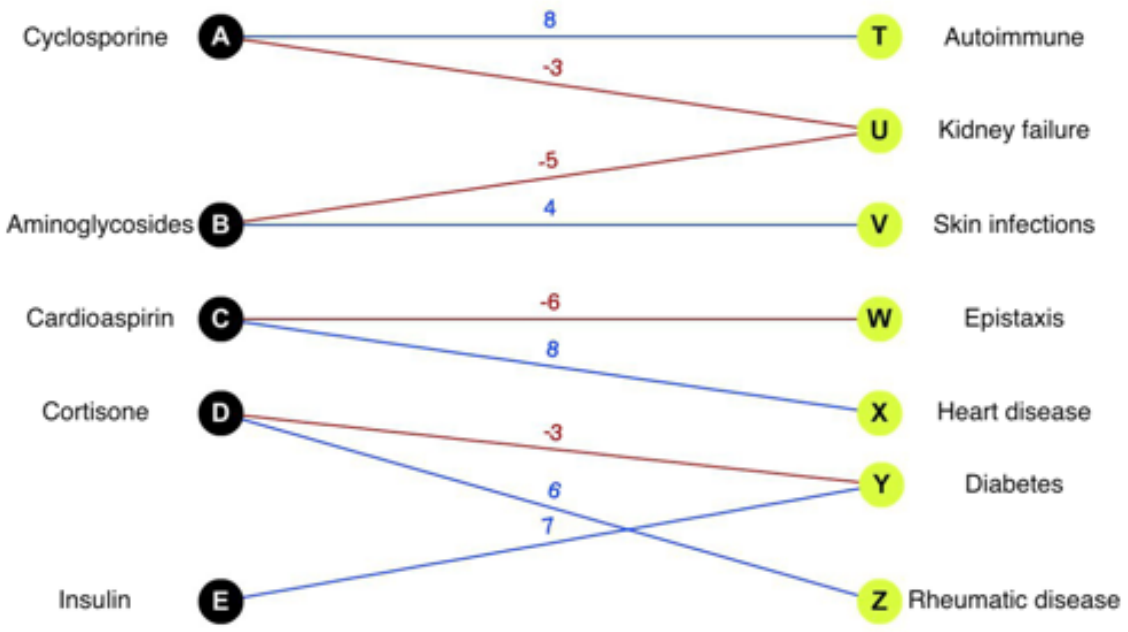


Figure 5.2: Drug-disease relationship

To allow the automatic verification step executed via a custom DSS algorithm the following scheme in Fig. 5.2, to be considered as a demonstrative example based on the analysis provided in [140] and [141] and [142], is used as a base to implement the weighted-score drug-disease relationship.

In a bipartite graph all nodes on the left-hand side represent drugs and all nodes on the right-hand side represent diseases. The weighted score edges represent the efficacy of a treatment (with a positive score) or the collateral effect (with a negative score). For example, Cardioaspirin is used in patients with a clinical record of heart diseases (hence a positive score is provided), but due to its anticoagulant effect has contraindications in case of epistaxis and bleedings (indicated with a negative score). Also, aminoglycoside is used for skin infections but due to its hepatotoxicity it can lead to a worsening of the clinical picture in patients with kidney failure.

The DSS algorithm considers the total sum of all the therapies concerning a given disease and raises an alert message if the resulting value is below a threshold value. The threshold value can be set according to the security level we want to reach and/or the strategies that are implemented to address alerts (e.g. the setup of a videoconferencing tool for remote consulting).

5.1.4 Implementation

As described in Fig. 5.1, a web interface implemented with HTML, CSS and JavaScript serves as an entry point for the entire system. All the requests flow through this interface and are elaborated by a server built in Python3 leveraging Flask as Web Server Gateway Interface (WSGI) ad Gunicorn to handle multiple requests with a Production-ready setup.

All the components are configured as Docker containers to take the advantages of virtualization technology allowing service portability, resiliency and automatic updates that are typical of a Cloud Infrastructure as a Service (IaaS). The WSGI provides a front-end that allows retrieving all existing patients' information (such as personal details, disease (ICD9-ICD10) and pharmaceutic (ATC) codes and links to clinical documentation), add new patient and submit new treatments specifying all the required pieces of information.

All the information are stored in a NoSQL MongoDB database. All the clinical documentation produced is uploaded in a local instance of NextCloud storage using a folder per treatment which does not contain any patient personal data rather than the patient's anonymized identification number (for GDPR compliance rules). Every change in the files or content of the folder will be tracked making it possible to keep a history of the documentation and its modifications.

During treatment creation, the DSS analyzes the therapy prescribed for the given list of pathologies and, if an incompatibility is found, sends an alert to the doctor highlighting that a similar treatment (i.e. same ICD and ATC as per current registration) led to death in at

least one case. The recommendation is to start a meeting with other medical doctors of the department or FHC, but it is left to MD whether to proceed.

5.1.5 Performance Assessment

Experiments were focused on the response time of our IaaS implementation in order to asses the performance of the DSS algorithm evaluation system. In particular, the system assessment has been conducted analyzing the total execution time required to perform a varying number of requests simulating an urban hospital with several wards exploiting the DSS algorithm.

Table 5.1: Summary of experiments performed.

Parameter	Value
Number of requests tested	[1; 10; 100; 1000]
Test executed for each run	30
Confidence interval	95%
Total patients in db	1.000.000

Table 5.1 summarizes the performed experiments. The reference environments is a hospital clinic with many patients already registered and a selected number of patients presenting similar clinical conditions to the case under test. Hardware (HW) and Software (SW) characteristics of these environments are shown in Table 5.2.

Table 5.2: Hardware and Software characteristics of testbeds.

Parameter	Value
RAM	4 GB
CPU	Intel® Core® i3-4170 CPU @ 3.70GHz, 2 core
Operative System	Ubuntu Server 18.04

Fig. 5.3 shows the execution time difference expressed in seconds to evaluate the DSS algorithm over an increasing number of comorbidities: Hypertension, Diabetes and Cardio-Vascular (CV) disease. On the x-axis we reported the number of DSS analysis requests, whereas on the y-axis we reported the processing time expressed in milliseconds. Looking at the graph, we can observe that for up to 100 requests the system response time is below a few seconds, and it considerably increases with a higher number of requests. The main aspect to appreciate is that the execution time remains nearly constant regardless of the number of comorbidities to evaluate.

Test results demonstrate how the DSS algorithm can be adapted to support MD evaluation

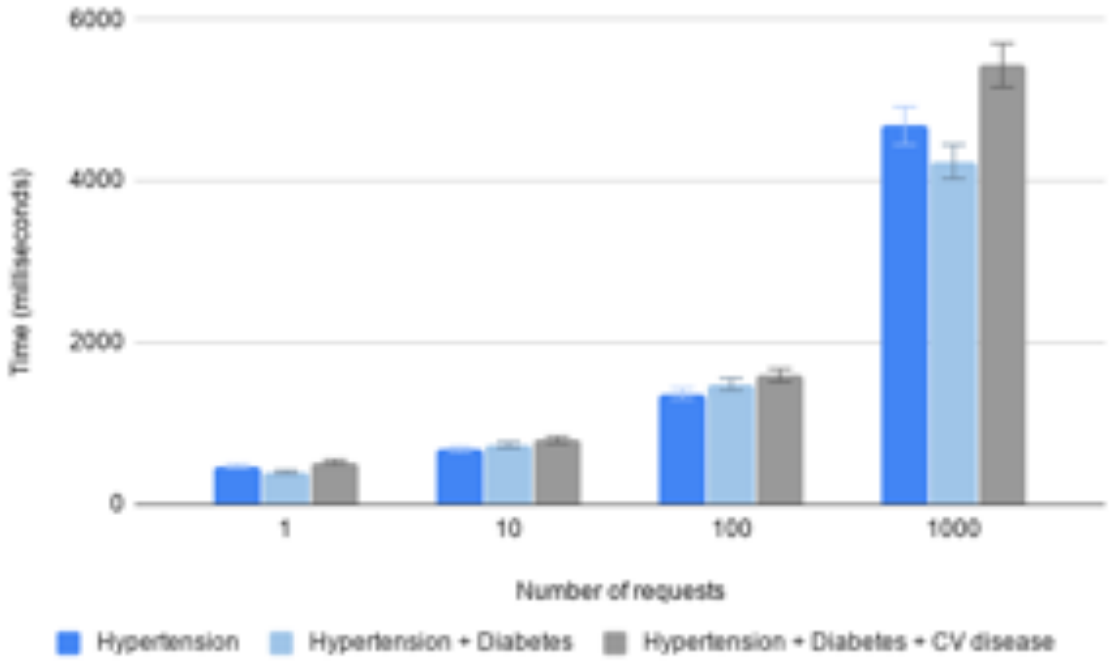


Figure 5.3: Execution time variation for the described scenario

in daily scenario leveraging the capability to evaluate a huge number of data in seconds combined with distributed and always up-to-date data when collected from FHC.

5.1.6 Some remarks

This Section demonstrates how therapies and treatments can leverage the use of ICT in hospital centers providing a fast and clear response to support medical decisions in complex treatments. A proper database structure is required to perform such operations but Big Data analysis can provide improvements where data are not structured. It has been demonstrated how a distributed approach can be helpful with large data to explore and to not stress system resources, whilst it generally requires additional time to complete the job. This work will be extended integrating secure base solutions to safely store all the information and decisions made during the course of the visit, making it possible to develop a worldwide supported and anonymized algorithm that uses trusted and certified data, ensuring anti-tampering and alteration are safeguarded.

5.2 Improving Healthcare Workflow with Blockchain

In a pandemic situation such as that we lived between 2020 and 2022, due to the Covid-19 virus, the need of tele-healthcare service becomes dramatically fundamental to reduce the movement of patients, thence reducing the risk of infection. Leveraging the recent Cloud computing and Internet of Things (IoT) technologies, this research aims at proposing a tele-medical laboratory service where clinical exams are performed on patients directly in a hospital by technicians through IoT medical devices and results are automatically sent via the hospital Cloud to doctors of federated hospitals for validation and/or consultation. In particular, it is discussed a distributed scenario where nurses, technicians and medical doctors belonging to different hospitals cooperate through their federated hospital Clouds to form a virtual health team able to carry out a healthcare workflow in secure fashion leveraging the intrinsic security features of the Blockchain technology. In particular, both public and hybrid Blockchain scenarios are discussed and assessed using the Ethereum platform.

Recent advancements in Information and Communication Technology (ICT) have paved the way toward new innovative tele-healthcare services able to face the growing demand of even more accessible medical treatments [151] [152]. Moreover, in the pandemic condition such as that we are living at the time of writing of this paper due to the Covid-19 virus, the need of tele-healthcare service becomes dramatically fundamental to reduce the movement of patients, thence reducing the risk of infection. However, the recent innovation brought by Cloud computing and Internet of Things (IoT) paradigms have been only partially taken into consideration by hospitals and more in general by medical centers so far. In this regard, a crucial aspect that has slowed down the wide adoption of such ICT paradigms in hospitals has regarded integrity, security and privacy of exchanged data. Considering the healthcare domain, it is fundamental that shared pieces of clinical data must be certified and not corrupted to prevent intentional or accidental illegal data manipulation. Furthermore, patients' privacy must be guaranteed.

In recent years, Cloud computing and IoT paradigms along with the concept of the federation have been combined so that different variants born. The first paradigm variation regarded Cloud federation that was defined as a mesh of Cloud providers that are interconnected to provide a universal decentralized computing environment where everything is driven by constraints and agreements in a ubiquitous, multi-provider infrastructure [153]. With the advent of IoT, the IoT Cloud paradigm raised. It was defined as a distributed system consisting of a set of smart embedded devices interconnected with a remote Cloud infras-

tructure, platform, or software through the Internet able to provide IoT as a Service (IoTaaS). Furthermore, the natural evolution of the latter brought to the concept of IoT Cloud federation referred as an ecosystem composed of small, medium, and large IoT Cloud providers able to federate themselves to gain economies of scale and to enlarge their processing, storage, network, sensing and actuating capabilities to arrange more flexible IoTaaS [154]. The healthcare domain can benefit from these paradigms to improve clinical services and push down management costs through the creation of Hospital IoT Clouds [155] able to federate themselves.

This research focuses on medical laboratory as a case study. It is an applied science laboratory typically placed in a hospital or in a clinical centre where clinical pathology exams are carried out on clinical samples to obtain information about the health of a patient to make diagnosis, treatment, and prevention of diseases. Blood tests (e.g., complete blood count (CBC), glycaemia, and so on) are performed by biomedical laboratory health technicians directly in the clinical laboratory and results are validated and analyzed by doctors to make therapies. Specifically, leveraging the IoT Cloud federation paradigm we propose the emerging concept of tele-medical laboratory. It is a medical laboratory where clinical exams are performed on patients directly in a hospital by technicians through IoT medical devices interconnected with a Hospital Cloud system and results are automatically sent through the hospital Cloud to doctors of federated hospitals for validation and/or consultation. Biomedical laboratory health technicians, nurses, doctors and other clinical personnel belonging to different Federated Hospital IoT Clouds (FHCs) cooperate to form a virtual healthcare team able to carry out a healthcare workflow.

However, one of the major concern about the accomplishment of such a workflow regards how to guarantee the non-repudiation and immutability of all health decisions [156]. In recent years different solutions have been proposed to solve such an issue: among these, the Blockchain technology, thanks to its intrinsic features of data non-repudiation and immutability, has aroused a great interest in both scientific and industrial communities. One of the major applications of Blockchain regards smart contract, i.e., a computer protocol aimed to digitally facilitate, verify, and enforce the negotiation of an agreement between subjects without the need of a certification third party. Blockchain has been increasingly recognized as a technology able to address existing information access problems in different applications domains including healthcare. In fact, it can potentially enhance the perception of safety around medical operators improving access to hospital Cloud services that are guaranteed by greater transparency, security, privacy, traceability and efficiency. Considering

the tele-medical laboratory scenario, smart contracts can make the transactions related to the healthcare workflow trackable and irreversible.

Specifically, an architecture blueprint and a system prototype of FHC service enabling to address the healthcare workflow of a tele-medical laboratory scenario is proposed. In particular, a special emphasis is given to Blochchain comparing both public and hybrid (private/public) network scenarios using the Ethereum platform to assess both processing time and economic cost. In particular, the latter is necessary because the Ethereum public network platform available over the Internet requires that users (i.e., in our case federated Hospitals) pay a fee to perform each transaction.

5.2.1 Related Work

Recently, the role of Blockchain technology in the healthcare domain has been surveyed in several scientific works [135], [157], [158], [159], [160]. Blockchain can drastically improve the security of hospital information systems as discussed in [161],[162],[163],[164]. However, up to now, most of the scientific initiatives are either theoretical or at an early stage and it is not always clear which protocols and pieces of a framework should be used to carry out system implementations that can be deployed in real healthcare environments.

One application of Blockchain regards the supply chain in the pharmaceutical sector and the development of measures against counterfeit drugs. While the development of new drugs involves substantial costs related to studies to evaluate the safety and updating of the drug, the use of smart contracts guarantees informed consent procedures and allows in certifying the quality of data [165]. An efficient data-sharing scheme, called MedChain is proposed in [166]. It combines Blockchain, digest chain, and structured P2P network techniques to overcome the efficiency issues in healthcare data sharing. Experiments show that such a system can carry out higher efficiency and satisfy the security requirements of data sharing. As discussed in [167], different medical workflows have been designed and implemented using the Ethereum Blockchain platform which involves complex medical procedures like surgery and clinical trials. In particular, the smart contract system for healthcare management as been studied, also estimating associated costs in terms of feasibility. A piece of framework that integrates IoT networks with a Blockchain to address potential privacy and security risks for data integrity in healthcare is discussed in [168]. A Medical IoT Device represented by a Raspberry Pi 3 Model B+ is attached to the patient's body to monitor his/her vital parameters that are stored in an Off-Chain Database that is accessed by doctor, pharmacy and insurance company via a DApp. All transactions take place utilizing smart contracts in a permissioned

Blockchain system implemented employing Ethereum.

Differently from the aforementioned scientific initiatives, that are mainly based either on a public Blockchain networks approach, this research focuses on how a hybrid Blockchain network approach (mixing both private and public ones) can be used to carry out the healthcare workflow of a tele-medical laboratory running in a FHC environment. Moreover, in the following sections it is demonstrated that the Blockchain hybrid network approach allows reducing the number of required transactions, hence enhancing processing time and reducing economic cost.

In this Section, after a brief overview of recent advances in medical laboratory devices, we discuss the advantages of tele-medical laboratory service.

5.2.2 Recent Advancements Medical Laboratory Devices

A medical laboratory device is an equipment able to perform several blood tests including CBC, Basic Metabolic Panel, Complete Metabolic Panel, Lipid Panel, Thyroid Panel, Enzyme Markers, Sexually Transmitted Disease Tests, Coagulation Panel and DHEA-Sulfate Serum Test. Currently, there are many medical laboratory devices available on the market. A classification can be done considering "connected" and "not connected" devices. For "connected" devices we intend medical laboratory equipment including USB and network (wired and/or wireless) interfaces and able to export and send results to other devices, whereas for "not connected" devices we intend medical laboratory devices without any interface for data transmission. In the following, we provide an overview of the major "connected" medical laboratory devices that are based on future tele-medical laboratory services. Telemedcare Clinical Monitoring Unit (CMU) [169] is a medical device able to perform blood pressure, pulse oximetry and blood glucose exams. Enverse [170] is a device able to perform continuous glucose monitoring. It consists of a chip that is installed subcutaneously on the patient that is connected with a mobile app. Med-Care [171] is an integrated solution for the auto-monitoring of glycemia that works with both web and mobile systems and that can send alerts via email or SMS. HemoScreen [172] is a low-cost portable haematology analyzer which performs a complete blood count at the point of care including a local web interface. Samsung Labgeo PT10S [173] is a portable clinical chemistry analyzer that improves efficiency by saving time for clinicians and patients through fast, easy and accurate blood analysis. It includes an ethernet interface for export exam result in an external Personal Computer (PC). All the aforementioned devices requires a blood sample in order to perform exams. Currently, alternative non-invasive experimental devices able to perform blood tests are the argument

of study for both academic and industrial healthcare communities.

5.2.3 Towards Tele-Medical Laboratory

Tele-medical laboratory allows performing blood exams, results validation, diagnosis and therapy assignment tasks in departments located in different hospitals. This is possible utilizing the creation of a virtual healthcare team composed of biomedical laboratory health technicians and doctors belonging to different federated hospitals. Cooperation is possible through a federation of FHCs. Hospital federation can involve several satellite clinics belonging either to the same healthcare organization or to different ones. An example of healthcare organization including different hospital is the provincial healthcare organization of Messina (Italy), also referred to ASP Messina. As shown in Figure 5.4 it includes eight health districts including, Messina, Taromina, Milazzo, Lipari, Barcellona Pozzo di Gotto, Mistretta and Sant'Agata di Militello. The health district of Lipari is placed on the island

Figure 5.4: Federation of hospitals: clinical data is shared across participants for cooperation

of Lipari and provides a limited number of health services. It offers a first aid to patients using an emergency room and a medical laboratory of clinical pathology. Due to the limited number of health departments, patients with particular diseases are typically transferred

in the near health districts of Milazzo or Barcellona Pozzo di Gotto (indeed by helicopter for urgent cases) if required. In this scenario, a tele-medical laboratory service could help the accomplishment of a clinical workflow involving a virtual healthcare team including technicians and doctors belonging, for example to the Lipari, Milazzo and Barcellona Pozzo di Gotto districts. In particular, blood tests could be performed in the medical laboratory of Lipari by biomedical laboratory health technicians and results transmitted using the FHC ecosystem to a doctor of the Barcellona Pozzo di Gotto district for validations. Furthermore, leveraging the FHC environment an additional consultation could be done with a doctor of the Milazzo district.

Defining with the term "home hospital" the hospital that is physically reached by the patient, the generic healthcare workflow accomplishing the aforementioned scenario implies the following phases:

1. **Hospitalization**: patient reaches a home hospital; personal details, date and type of visit are recorded; the patient is identified by a visit identification code; a doctor schedules all exams required to verify the nature of the disease. A virtual healthcare team is created involving technicians, nurses and doctors belonging to the home hospital and other federated hospitals;

2. **Clinical Analysis**: if required, a nurse of the home hospital takes a blood sample from the patient; a biomedical laboratory health technician of the home hospital performs blood tests and results are saved by IoT medical devices automatically or by the technician manually on the FHC storage in a dedicated patient's directory;

3. **Validation**: a doctor belonging to another federated hospital analyzes and validate the results of clinical analysis;

4. **Consultation**: a selected pool of doctors belonging to the virtual healthcare team establish a teleconference to clarify the patient's clinical situation; the patient's health data and clinical analysis are shared among doctors hiding sensitive data;

5. **Monitoring**: the hospitalized patient is constantly monitored by nurses who apply treatments based on therapeutic indications; each treatment is recorded until the dismissal.

Non-repudiation and immutability of all health decisions is a fundamental concern that must be addressed for the accomplishment of such a healthcare workflow. In this regard,

the Blockchain technology using smart contracts can make all transactions related to the healthcare workflow trackable and irreversible. In the remainder of this paper, we will focus on such an aspect.

5.2.4 System Design

It is important to guarantee that only authorized members of the virtual healthcare team are allowed to take actions because a wrong decision can lead to a worsening of clinical condition or death of a patient.

Therefore, the FHC system has to guarantee that all actions performed by virtual healthcare team members are trackable and irreversible. To achieve such a goal, the following technologies are fundamental:

- **Blockchain engine**: to use the features of a decentralized and distributed certification system with the technology offered by the development and coding of a smart contract.
- **Cloud storage**: to use an open-source and open-architecture file hosting service for file sharing managed with authorizations to archive all the files required to support the analysis of disease causes such as blood tests, Computed Tomography (CT) scans and laboratory tests;
- **NoSQL database**: to exploit the potential of a document-oriented distributed database able to store and manage patient's data and diseases through tags for a fast and efficient search and to store Blockchain transaction hashes and links to files stored in Cloud Storage;

Figure 5.5 describes the FHC architecture. The system entry point of each hospital Cloud is a dedicated Web Server Gateway Interface (WSGI) where pieces of electronic healthcare record data are created manually by the clinical personnel or automatically collected by IoT devices that can be spread over different federated hospitals.

A Data Anonymization Module (DAM) is responsible for hiding the patient's personal data in electronic health records. This is possible by decoupling the patient's personal data from electronic health records storing related pieces of information in different databases. To bind the electronic health record with the patient, a patient's anonymized identification number (patient_id) is generated from the DAM and stored within the electronic health record. Thus, only electronic health records are shared between FHCs, whereas pieces of

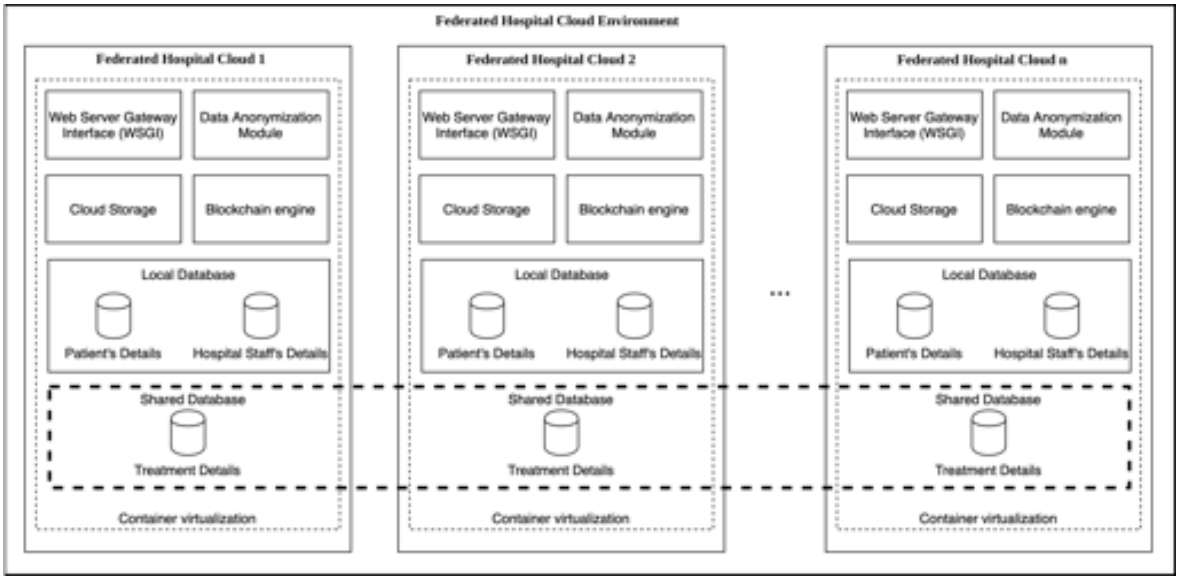


Figure 5.5: FHC architecture.

patients' personal data are never shared to preserve their privacy. All the produced clinical documentation, including electronic health records, is uploaded on a Cloud Storage containing a patient_id to hide patient's data. A Blockchain engine is responsible to store information on the Blockchain to certify data non-repudiation and immutability of treatment details guaranteeing accountability and authenticity. The transaction hash resulting from the mining process is stored in the NoSQL document-oriented database as an attribute of the treatment. A local database instance containing both patients and hospital personnel data is isolated from other federated hospitals because these never require to be shared. Treatments' details are anonymized and stored in a database shared with other participants to the FHC environment. The whole architecture deployed using container virtualization to simplify installation, configuration and maintenance in each FHC.

Hospitals belonging to a Federation cooperate ensuring that appropriate therapies and procedures are carried out. Figure 5.6 shows the sequence diagram describing an example of healthcare workflow including three federated hospitals and where a tele-medical laboratory service is provided. A generic patient who requires a medical visit reaches the emergency room of a home hospital that after evaluating the urgency of the case, identifies an available doctor. At his/her turn, the patient is visited by the assigned doctor who prescribes some clinical analysis such as blood tests which can be done in the home hospital medical laboratory by a biomedical laboratory health technician. If the doctor responsible for the medical laboratory is not available (as the case of small hospital districts) a doctor belonging to the federated hospital 1 validates the results shared via federated Cloud storage. Thence, such a

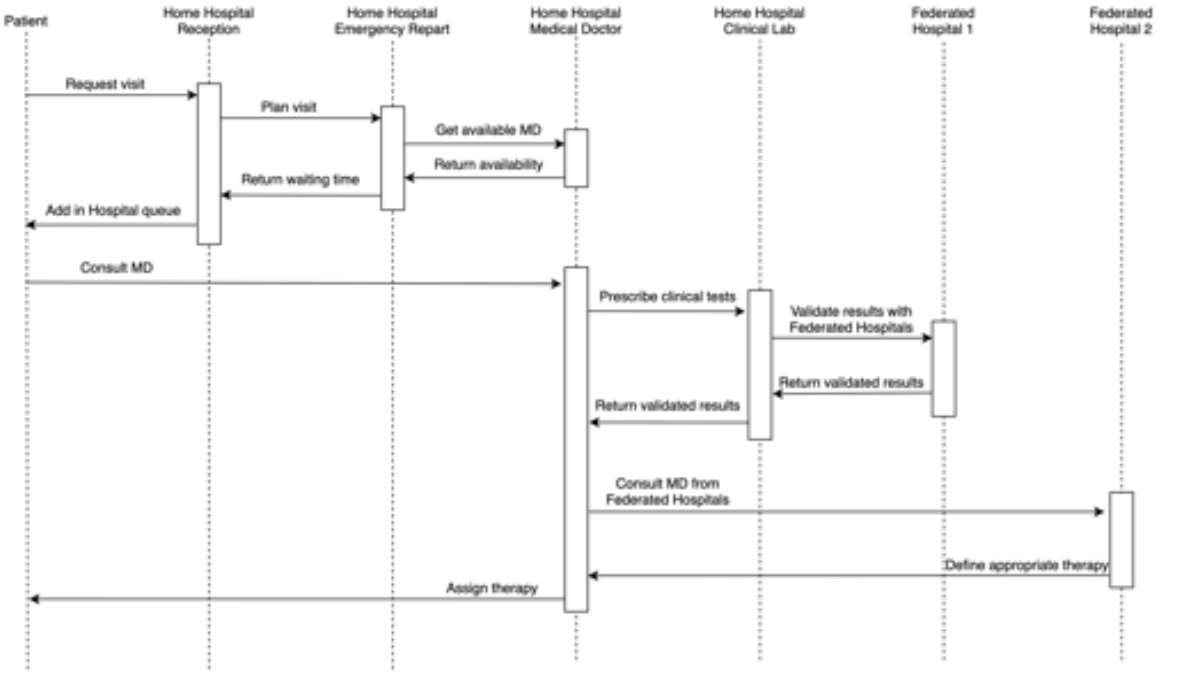


Figure 5.6: Sequence diagram describing an example of healthcare workflow accomplished in a Federated Cloud hospital environment.

doctor joins the virtual healthcare team along with involved medical personnel of the home hospital. Since the doctor of the home hospital has a doubt the therapy to prescribe, he/she consults a doctor of federated hospital 2 via teleconference. After this consultation, a therapy is assigned to the patient.

5.2.5 System prototype

The FHC architecture was designed to enable a virtual healthcare team to carry out every healthcare workflow such as that described in the previous Section. Figure 5.7 shows the main software components of a possible system prototype implementation.

All requests coming from patients, nurses, technicians and doctors flow through the WSGI interface developed with the Python web application framework Flask and deployed on the Gunicorn Python WSGI HTTP server. All the components are configured as Docker containers to take the advantages of virtualization technology allowing service portability, resiliency and automatic updates that are typical of a Cloud Infrastructure as a Service (IaaS). The WSGI provides a front-end that allows retrieving all existing patients' information (such as personal details, disease and pharmaceutic codes, links to clinical documentation and Blockchain hash verification); adding new patients; and submit new treatments specifying all the required pieces of information. Specifically, a web page is dedicated to the registration of a new patient for saving his/her primary personal information and another web page is dedicated to the

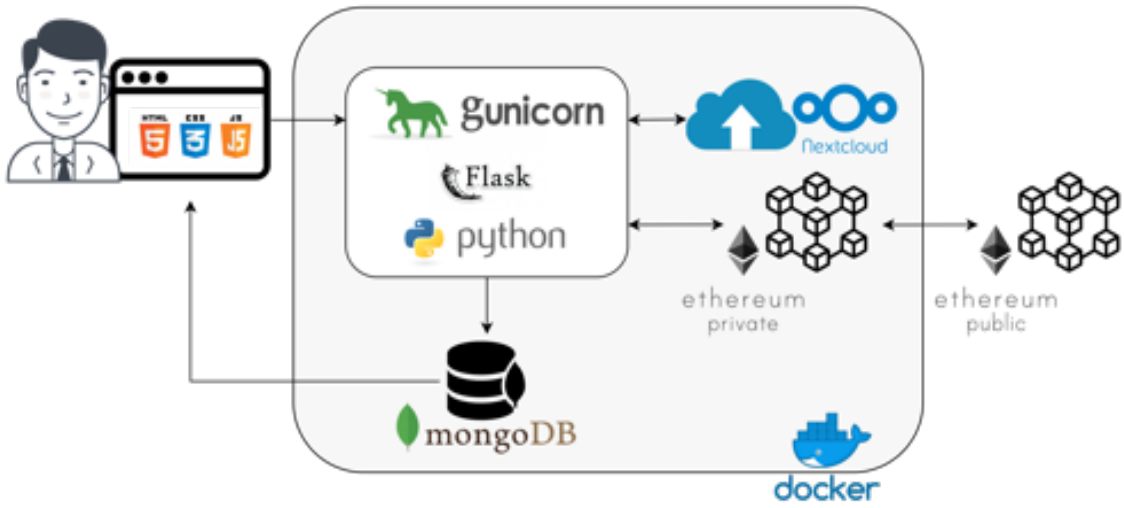


Figure 5.7: Hospital Cloud software components.

registration of a new treatment. It is possible to select the medical examination date, patient and doctor who does the registration.

All the produced clinical documentation is uploaded in a local instance of NextCloud storage using a folder for each treatment which does not contain any patient's personal data, but a patient's anonymized identification number. Every change in the files or content of the folder will be tracked making it possible to keep a history of the documentation and its modifications. Recently, a few related works were published for data anonymization in a Multi-Cloud storage environment considering a healthcare scenario, however, these do not guarantee that pieces of data are trackable and irreversible [174], [175]. Since patients' sensitive data must be anonymized and health records and treatments must be trackable and irreversible, related pieces of information where stored combining a MongoDB NoSQL DataBase Management System (DBMS) with the Ethereum Blockchain platform. Therefore, all pieces of information are stored in both MongoDB and in the Ethereum network through smart contracts developed in Solidity. For experimental purposes, the Ethereum network was implemented in both public and hybrid configurations. In the first case, all FHCs share the public Ethereum network available over the Internet, whereas, in the second case each FHC hosts a private Ethereum network to store local transactions and a public Ethereum network to synchronize the local transactions performed in each FHC.

The smart contract accepts the input parameters such as anonymized patient id and doctor id, disease and pharmaceutic codes and stores these pieces of information in a simple data structure. The hash code resulting from the mining of each transaction is stored in the MongoDB database and can be used for verification using services like etherscan.io.

This service is capable of detecting any modification occurred to files or folder using a listener called *External script*. It is then possible to store the fingerprint and timestamp of each modification in the database thus making it possible to track the history of each treatment. This feature is important to guarantee the data integrity.

5.2.6 Experiments

Currently, the most adopted Blockchain configuration is based on a public network approach. With this regard, Ethereum is one of the major Blockchain platforms. A public Ethereum instance is available over the Internet and requires the payment of a fee for the execution of each transaction. However, the Ethereum platform can be also downloaded and installed in a private network. In this paper, using Ethereum , the objective of our experiments was to verify if the Blockchain system of a FHC ecosystem including a tele-medical laboratory service can be optimized in terms of both processing time and economic cost using the proposed hybrid network approach. Apart from processing time, it is important to highlight that considering the Ether (ETH) cryptocoin concurrency used in Ethereum, a small saving of ETH can result in putting aside a relevant amount of money (e.g., USD or EUR) in just a few months. In the following, we provide a description of both considered approaches.

- **Ethereum public network**: each healthcare treatment is recorded in the public Ethereum Blockchain network where time-to-mine and cost are subject to Ethereum network traffic (depending on the queue size, more time is required to extract transaction data, higher is the cost for transaction management.).
- **Ethereum hybrid network**: each healthcare treatment is recorded in a private instance of Ethereum Blockchain network consisting of at least one node for each FHC and only one hash code, calculated as the MD5 of the last one-hundred concatenated treatments' transaction hash, is written in the public Ethereum Blockchain network. In case the number of daily treatments is less than one-hundred, the MD5 hash code is calculated as the concatenation of the last 24 hours' treatments transactions hash. The result of this is a negligible waiting queue and ETH cost but, on the other hand, there is a reduction of the mining power as a reduced number of miners are present in the private network as compared to the public one.

The system assessment has been conducted analyzing the total execution time required to perform a varying number of transactions in healthcare workflows. Each FHC was simulated considering a server with following hardware/software configuration: Intel® Xeon® E3-12xx

v2 @ 2.7GHz, 4 core CPU, 4 GB RAM running Ubuntu Server 18.04. Each test has been repeated 30 times considering 95% confidence intervals and the average results are plotted.

Table 5.3 summarizes experiments setup and average outcomes.

Parameter	Value
Number of transactions tested	[1; 100; 1000]
Test executed for each run	30
Confidence interval	95%
Gas price (Gwei)	2
Average cost per transaction (ETH)	0.0002
Average mining time (s)	130

Table 5.3: Summary of experiments performed.

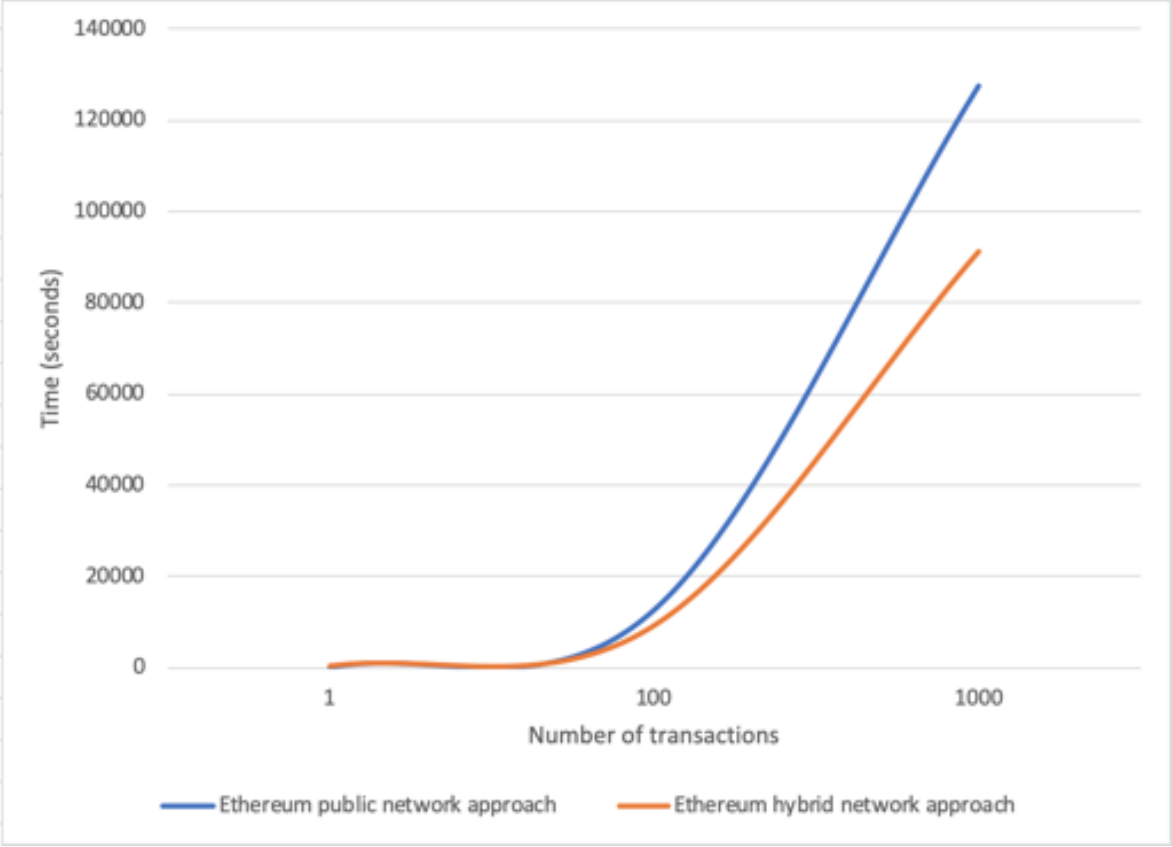


Figure 5.8: Time comparison for public and hybrid Ethereum network approaches considering a varying number of treatment registration requests.

Figure 5.8 shows the time-to-mine difference expressed in seconds between the two approaches considering new treatment registration requests. On the x-axis we reported the number of treatment registration requests, whereas on the y-axis we reported the processing time expressed in seconds. Looking at the graph, we can observe that for roughly 80 treatment registration requests both configurations present a similar trend, whereas increasing the

number of requests the hybrid Ethereum network shows better performances than the public one thanks to the reduced waiting time in the mining queue.

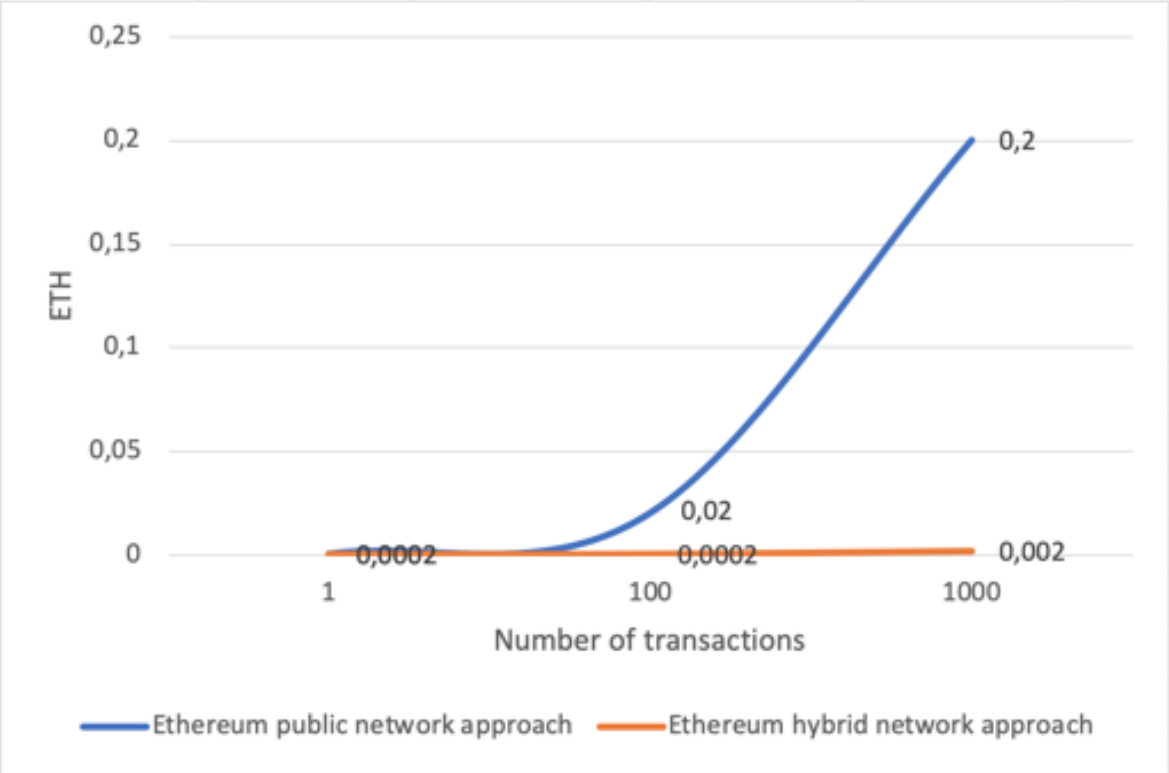


Figure 5.9: Cost comparison for public and hybrid Ethereum network approaches.

Figure 5.9 describes a cost comparison in Ether (ETH) for the two approaches. On the x-axis we reported the number of treatment registration requests, whereas on the y-axis we reported the cost expressed in Ether (ETH). From our tests, we appreciated that the average cost of a single transaction (i.e., a simple Smart Contract representing a new treatment memorization or modification) that has to be written in the public Ethereum Blockchain, is roughly 0.0002 ETH. It can be noted that for a small number of transactions there is not a perceptible convenience in preferring the proposed hybrid approach. This is because at least one transaction per day is written in the public Ethereum Blockchain. However, it is clear that the money-saving increases exponentially increasing the number of transactions because only one public transaction every one-hundred of private treatments will be paid with ETH cryptocurrency, resulting in an important cost-saving for the FHC ecosystem.

Test results demonstrate how the Ethereum hybrid network approach can be adopted to improve both processing time and cost-saving maintaining the same level of accountability and data certification as the public approach through certification on Blockchain.

5.2.7 Some remarks

This Section demonstrated how a tele-medical laboratory service can be developed through a healthcare workflow running in a FHC environment leveraging Blockchain. Experimental results highlight that the performance of the Ethereum hybrid network certification system is improved in terms of cost and response time compared to an alternative public approach.

Definitely, the Blockchain technology is destined to evolve shortly improving system capabilities and robustness, and public test instances with different consensus protocols will be made available with benefits on performance and scalability.

In the pandemic condition that authors are living at the time of writing of this paper due to the Covid-19 virus, the need of tele-healthcare service becomes dramatically fundamental to reduce the infection risks for patients, thence reducing their movement. We hope that with this paper, we succeeded in stimulating the attention of both academic and industrial communities toward the adoption of Blockchain in the healthcare context to speed up the development of innovative tele-healthcare services.

In future developments, this work can be extended integrating a comprehensive healthcare scenario with different involved organizations, such as pharmaceutical companies registering in the Blockchain all the phases of drug production until the sealing of final package and shipment. Thus, when a patient buys a prescribed medicine it is possible to link the patient with the medicine box, which would mean an important step towards the end of drugs' falsification and an important assurance for the end-user who can be identified in case a specific drug package has been recalled.

5.3 Certify data in eHealth using Blockchain

With the latest demand of devices connected to internet and the increasing need of on-demand services and real-time tracking, security is one of the biggest concerns for the diffusion of online services. If in the past security was concerning the access to the data, moving from different levels of password security algorithms to more advanced data biometric access, the new important need is to certify data exchanged, making it not mutable and access-controlled. Critical environments such as hospitals are evolving to accommodate the need of a connected world and it is becoming crucial to guarantee that data is authentic, reliable and certificated.

Since the advent of Bitcoin in 2009 and Smart Contracts in 2015 healthcare has been one of

the main fields of study for Blockchain applications, and General Data Protection Regulation (GDPR) helped to improve researches on patient's privacy data protection.

This research proposes an application of inherit features of security, anti-tempering and reliability of Blockchain Technology to address data authenticity of clinical treatments to hospitalized patients describing all phases, from the analysis to the design and implementation, of a web interface serving as entry point for a federation of hospitals. Furthermore, the solution has been load tested to store all the relevant treatment information in a private instance of Ethereum Blockchain.

The demographic growth of the last century combined with increased life expectancy and shortage of specialized doctors in Europe [151] [152] caused access to proper medical treatment a mayor concern of the last decade. The advent of technology and especially Information and Communication Technology (ICT) has been slowly taken in use in hospitals and main medical structures [176], despite so many research papers have been written on the subject from both Engineers and Doctors.

A crucial point in the dissemination of information sharing in the healthcare area, however, lies in the security of the data exchanged.

It is essential that the shared healthcare data are certified and their integrity guaranteed to prevent invalid information being intentionally loaded in order to invalidate any use.

In recent years different solutions have been proposed to solve the problem, among these the Blockchain thanks to its characteristics of non-repudiation and immutability of the data, has aroused much interest in the scientific community.

Founded in 2009 as the technology behind Bitcoin [15] has completely revolutionized traditional encryption-based security systems, introducing a new concept of hash-based encryption in which information is saved on blocks and each block is linked to the previous one via a hash coding.

Blockchain technologies have been increasingly recognized as a tool to address existing information access problems. Thanks to these it is in fact possible to improve access to healthcare services to achieve greater transparency, security and privacy, traceability and efficiency, as well as increase the perception of safety in the healthcare sector.

This research describes all the phases from the analysis to the design and implementation of a web interface serving as entry point for a federation of hospitals that cooperates to provide the optimal clinical treatments to hospitalized patients. In particular the proposed solution tracks the treatment of the patient from the hospitalization until the dismissal, supporting doctors in the planning of pharmaceutical treatment.

5.3.1 Related Work

The healthcare sector has a growing demand for Blockchain developments and the traditional industry is actively exploring new avenues for using the Blockchain to meet its critical needs.

Blockchain can offer a unique solution for healthcare assistance thanks to wearable devices and IoT (Internet of Things) and applications are described in [161], [162], [163] and [164] for the management of patient privacy in the health sector. However, the technologies and protocols to be used for effective implementation are not specified.

Researches are focusing on applications of Blockchain to guarantee only authorized access to the patients medical information [177] but our work explores ways to preserve anonymized data through immutability and anti-tampering of clinical history documentation and the results of clinical studies.

This work describes a practical implementation on how Blockchain can be used to improve medical analysis treatments empowering collaboration among a group of federated hospitals.

5.3.2 Motivations

This research aims to recommend new approaches to harmonize health procedures with new technologies to guarantee patients' safety and therapeutic certification, verifying that every doctor's choice is recorded, to guarantee that all hospital protocols have been scrupulously followed.

Furthermore, the system is developed to automatically propose a teleconference with a selected group of doctors to clarify the patient's clinical status in critical situation. The anonymized patient's health data and clinical analyses are shared with the other doctors participating in the federation of hospitals while the patient's data are never shared.

Fig. 5.10 describes a scenario where patient's clinical data is shared across participants to a federation of hospitals for cooperation and knowledge sharing, and the data exchanged is certified on a private Blockchain where all participants are known and trusted.

Specifically, the workflow of the system is the following:

1. **Hospitalization**: patient reaches the hospital and personal details, date and type of visit are recorded;

2. **Analysis**: patient follows the procedures to ascertain the nature of the disease (eg. blood tests, clinical examinations, possible CT scans, RX and laboratory tests) and the results

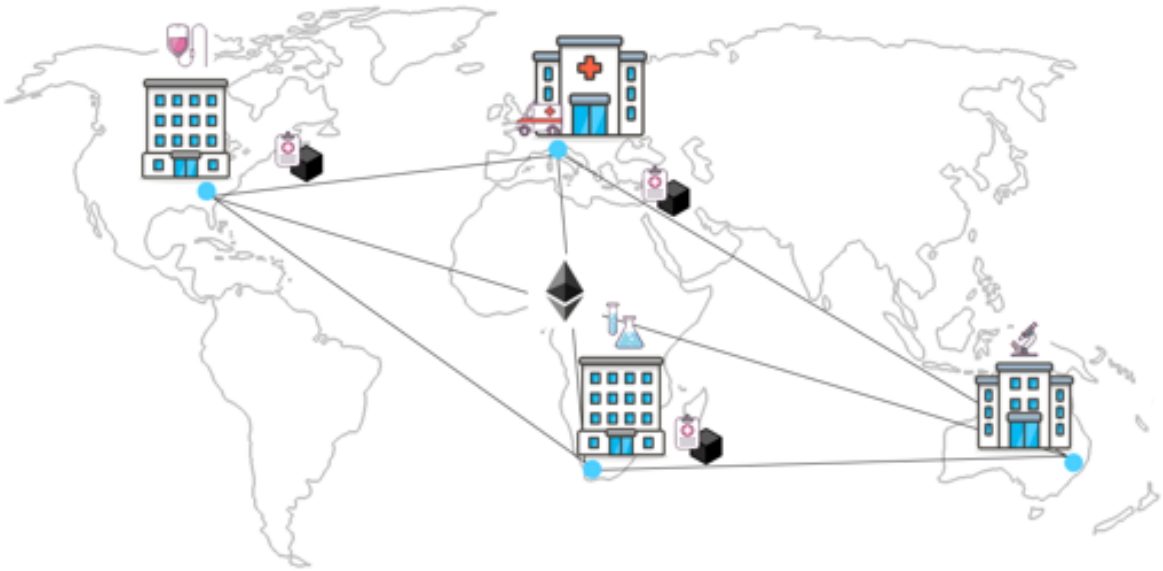


Figure 5.10: Federation of hospitals where clinical data is shared across participants for cooperation

of the analyzes are saved on a Cloud storage space inside the hospital managed on a dedicated directory for the patient and visit identification code;

3. **MD evaluation**: doctor analyzes the results of clinical analysis, CT scans, etc. and prepares a report with the therapy to be followed;

4. **Teleconference**: a selected pool of doctors is invited to participate in a teleconference to clarify the patient's clinical situation. The patient's health data and clinical analysis are shared with the other doctors inside the hospital or a federated hospital; the patient's details are never shared;

5. **Drug administration**: the hospitalized patient is constantly monitored by nurses who administer treatment based on therapeutic indications, and each administration is recorded.

5.3.3 Architectural Design

When medical doctor visits the patient he/she can add a prescription treatment indicating the disease to cure, a medicine and a description with dosage and mode of use.

It is important to guarantee that only the medical doctor is allowed to create new prescription or to update existing one, because a wrong diagnosis can lead to worsening of clinical condition or death and so it becomes mandatory to know who created a new record.

The system is designed as a web-portal where patient's records are stored and treatments are added for specific diseases resulting from a medical examination, aiming to harmonize

health procedures with new technologies in terms of safety and data certification as well as non-repudiation.

To develop this system, the following technologies are used:

- **Blockchain engine**: to use the features of a decentralized and distributed certification system with the technology offered by the development and coding of Smart Contract;
- **Cloud storage**: to use an open-source and open-architecture file hosting service for file sharing managed with authorizations to archive all the files required to support the analysis of the nature of the disease such as blood tests, CT scans and laboratory tests;
- **Storage NoSQL**: to exploit the potential of a document oriented database to manage patient data and diseases through tags for a fast and efficient search and to store blockchain transaction hashes and links to files stored in a Cloud storage.

5.3.4 Implementation

To provide a practical implementation of the proposed design a web interface is provided to manage prescriptions and patient's registration. Every information is stored in a NoSQL database and the most relevant attributes are stored in the Blockchain.

A web interface implemented with HTML, CSS and JavaScript serves as an entry point for the entire system. All the requests flow through this interface and are elaborated by a server built in Python3 leveraging Flask as Web Server Gateway Interface (WSGI) and Gunicorn to handle multiple requests with a Production-ready setup. All the components are configured as Docker containers improving application portability and supporting automation to delivery updates.

A Python3 web server has been developed to support the above-described architecture. It mainly consists of a front-end where it is possible to retrieve all existing patients' information (such as personal details, disease and pharmaceutic codes, links to documentation and Blockchain hash verification), add new patients and submit new treatments specifying all the required information.

A web page is dedicated to registering a new patient, saving his/her primary personal information, and a separate web page is dedicated to the registration of a new treatment. It is possible to select the medical examination date, patient and doctor who does the registration to be chosen from the patients already registered and available in the database.

All the information are stored in a NoSQL MongoDB database and an Ethereum private network through a Smart Contract developed in Solidity. It has been chosen to use Ethereum

with a private network installation considering what has been reported in [178] highlighting the impossibility for Hyperledger Fabric Blockchain to scale above 16 nodes, which results in an important limitation for the scope of this work which aims to create a trusted and federated network among multiple hospitals, and considering that Ethereum is more mature in terms of its codebase, user base and developer community.

The Smart Contract accepts the input parameters such as anonymized patient id and doctor id, disease and pharmaceutic codes and stores these in a simple data structure. The hash code resulting from the mining of each transaction is stored in the MongoDB database and can be used for verification using services like etherscan.io.

All the clinical documentation produced is uploaded in a local instance of Next-Cloud storage using a folder per treatment which does not contain any patient personal data rather than the patient's anonymized identification number. Every change in the files or content of the folder will be tracked making it possible to keep a history of the documentation and its modifications.

This service is capable of detecting any modification occurred to files or folder using a listener called *External script*. It is then possible to store the fingerprint and timestamp of each modification in the database thus making it possible to track the history of the treatment.

5.3.5 Performance Assessment

Experiments conducted are described in this section to provide an overview of capabilities and performances of an automated medical prescription system with different technology approach.

The following experiments have been conducted analyzing the total execution time required to perform a varying number of transactions through Ethereum Block-chain and its throughput.

The server used for this experiment presents the following configuration: Intel® Xeon® E3-12xx v2 @ 2.7GHz, 4 core CPU, 4 GB RAM running Ubuntu Server 18.04.

All analysis have been performed by sending transactions to the server varying the number of total and simultaneous requests.
All requests invoke new treatment registration and Ethereum Blockchain transaction mining for treatment registration. Table 5.4 describes the approach taken for these measurements:

To simulate a real private instance of Ethereum Blockchain, all tests have been performed using Ropsten Ethereum public test network, leveraging 300+ available nodes with a real server load status.

Table 5.4: Summary of experiments performed.

Num of Transactions	Num of Accounts
100, 250, 500	25, 50, 100

It must be considered that Ethereum Blockchain Ropsten environment is based on Proof of Work (PoW) consensus protocol which makes difficult to obtain scalability and system speed.

Figure 5.11 shows how performance are consistent on the basis of simultaneous requests and does not vary on the number of transactions.

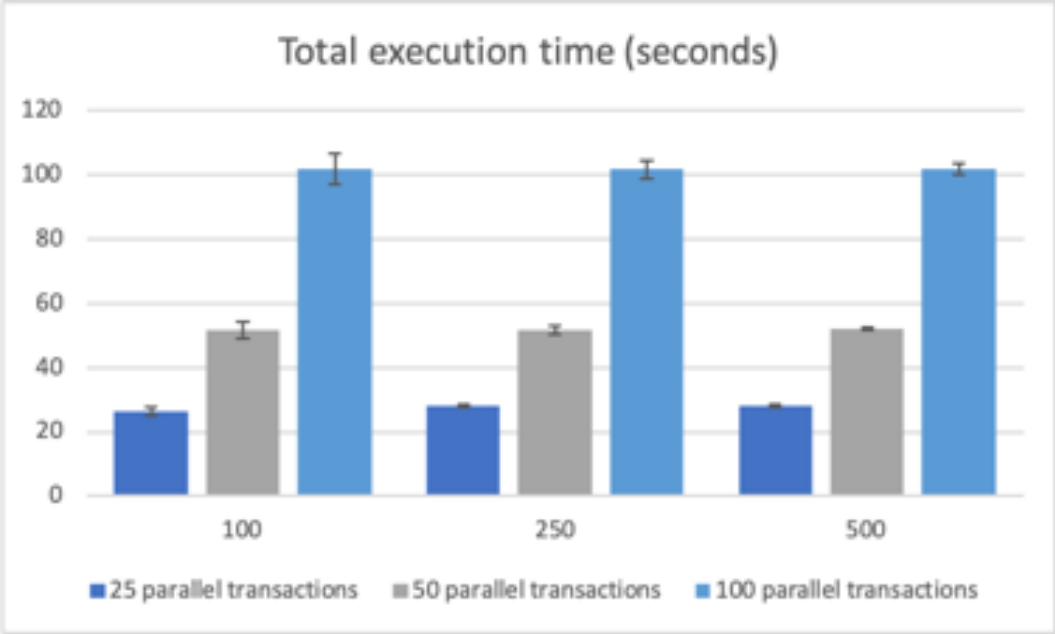


Figure 5.11: Total execution time variation for the described scenario without Blockchain mining

Figure 5.12 shows a linear degradation of the system as compared to the requests made without Ethereum Blockchain mining and also compared to the total number of transactions sent.

5.3.6 Some remarks

This Section demonstrates how Blockchain can be used in the healthcare environment to improve hospital workflow guaranteeing the authenticity of data stored.

Without doubts, the Blockchain technology will keep evolving in the future years improving system capabilities and robustness, and public test instances with different consensus protocols will be made available with benefits on performance and scalability.

This work can be extended integrating a comprehensive healthcare scenario with diverse

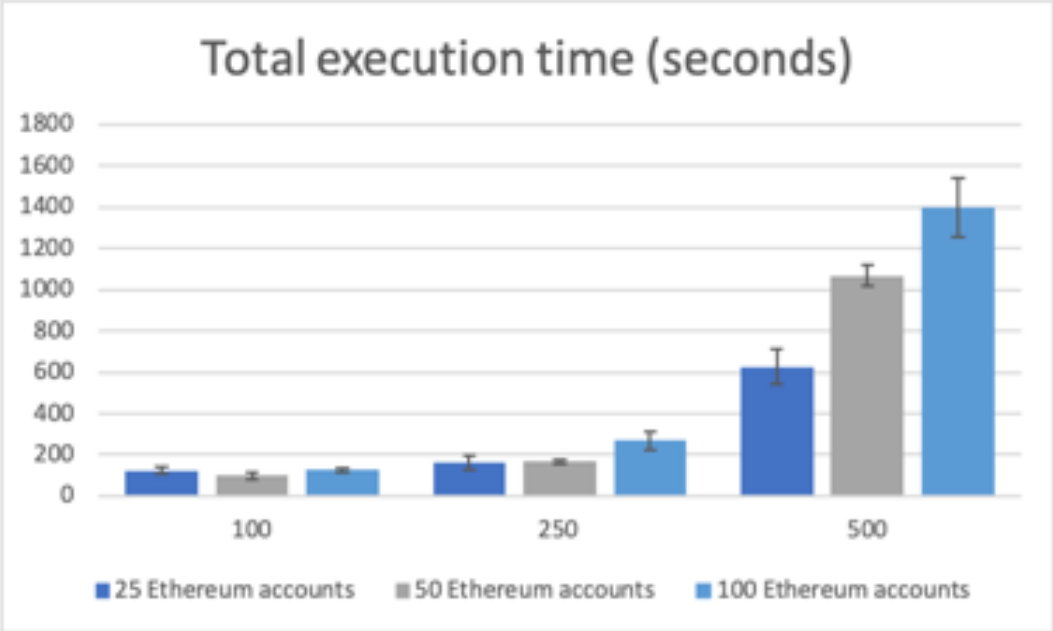


Figure 5.12: Total execution time variation for the described scenario

organizations involved, such as pharmaceutical company registering in the Blockchain all the phases of drug production until sealing of final package containing drugs and the same leaves the warehouse to be shipped to the pharmacy, so when patient buys a prescribed medicine it is possible to link the patient with the medicine box, which would mean an important step towards the end of drugs' falsification and an important assurance for the end-user who can be identified in case a specific drug package has been recalled.

5.4 Strategies to avoid fake data in eHealth

Every year the healthcare sector suffers from incorrect therapies and an increasing number of patients analysis, which causes congestion in the hospitals and, potentially, worsening of patient's clinical conditions. Extending the concept of the Decision Support System already investigated by the authors, this work advances the state of the art of Reinforcement Learning (RL) via Markov Decision Process formulation, considering an agent acting in his environment motivated by the achievement of the maximum individual objective by appropriate incentives. Transparency, security and privacy of the model are guaranteed by the adoption of Blockchain to enhance the perception of safety around medical operators improving access to hospital services. Experiments focused on the Smart Contract execution time and resources usage have proved the goodness of the proposed model considering both private and public Blockchain configurations.

Models projected towards the future of healthcare systems are nowadays founded on the assumption of leaving behind the old provider-centric viewpoint in favor of a patient-centric model. This shift involves the definition of new approaches where patients are driven towards forms of autonomous and collaborative management of their own self-care, even better if aided by adequate technological support, to meet practical and relational needs. The different actors of an innovative healthcare framework can enhance their skills thanks to the reliable effects of the Self-Regulating Learning (SRL) approach and theory applied to a suitably defined telehealth program. To achieve this goal, the healthcare environment can be thought of as a Reinforcement Learning (RL) setting and, if supported by advanced technological integration, it may collect a considerable amount of observations useful to identify the optimal strategy for the actors of the telehealth system.

By introducing the Blockchain paradigm, every action can be immutably recorded into a distributed digital system. This guarantees the monitoring and verification of all activities, provides a tool for integrated clinical decision support and risk analysis, and it can potentially enhance the perception of safety around medical operators, thus improving access to hospital services that are guaranteed by greater transparency, security, privacy, traceability and efficiency.

One of the major applications of Blockchain regards smart contract, i.e., a computer protocol aimed to digitally facilitate, verify, and enforce the negotiation of an agreement between subjects without the need of a certification third party. Considering the SRL scenario, smart contracts can make the transactions related to the healthcare workflow trackable and irreversible. The clinical decision support system for the benefit of the medical staff aims at validating the effectiveness of drug treatments and isolate possible intolerances based on ad-hoc big data analysis algorithms and relevant health information via Blockchain technology. Decentralized Blockchains are immutable and each partner taking part in the blockchain is able to update, manage and coordinate distributed registers independently but under the consensual control of the other nodes of the network. Data control and verification algorithms create a historical memory to manage, control and verify the transactions and exchanges that have been made in order to guarantee traceability and data integrity.

In this paper we address the study of a Reinforcement Learning model for SRL. Starting from the collection and analysis of real and uncorrupted health data, the model is able to provide accurate predictions around cause and effect relationships and comorbidity without human involvement. Data authentication and certification are required to efficiently work with Artificial Intelligence (AI). Altered or corrupted data are the main threat to develop

a good model. The Blockchain-based strategy proposed in this paper ensures that the "AI" performed inside a Machine Learning (ML) model of Remote Healthcare is real and trustworthy. The introduction of Blockchain technologies integrated with AI algorithms inside a training system belonging to the domain of healthcare is designed in association with the advantageous aspects set out below. First, the proposed strategy aims to guarantee maximum transparency both in the logic of the algorithm and in the certain identification of the actors involved in the dynamics and the responsibilities associated with the actions they make. In fact, one of the most common problems related to AI is the risk of altered data [179] that invalidates, fully or partially, the training data leaving the entire process unusable. Moreover, data anonymization is adopted to hide the patient's personal data in electronic health records, to protect the safety of the patient's data. In addition, protection from third parties concerning potential attacks or distortion of hackable inputs is included, and the prospect of involving several hospitals in the consortium, benefiting from outsourcing, involves increasing data reliability and potential scalability. This can be mitigated with the implementation of Blockchain technology to certify the data, ensuring the Confidentiality, Integrity and Availability (CIA) [78] are guaranteed.

The reasons described suggest that Blockchain can be a valid reliable and secure mechanism for data validation and certification, offering intrinsic security features that would be expensive to achieve and maintain with other kinds of implementation.

5.4.1 Related work

Many innovative technologies and methodologies are applied in telemedicine for the treatment of outpatient patients in recent years. In [180], AI, interconnected digital sensors, and mathematical modeling are integrated into a continuous cycle of learning and adaptive description of the evolutionary trend of the process of acquiring health skills and increasing awareness and competence in the care of one's personal health. Moreover, ML techniques strengthen the infrastructure with the integration of cognitive skills for the benefit of both the medical staff, validating the treatment, and the patient, to support the self-regulated learning process to achieve individual objectives.

The privacy-sensitive nature of medical information makes data related to health procedures, which need to be accessible to medical practitioners, prone to malicious entities. In such a sense, a federated approach using Blockchain-based security, involving distributed and locally stored data for the use of all actors of the healthcare sector, represents an effective solution for AI algorithms. These processes can be applied for further analysis and diagnosis

to ensure that the data is certified and the model is not altered with fake data introduction. Thus, the Internet of Medical Things can benefit from combing artificial intelligence techniques and blockchain advantages to implement a privacy-preserving federated learning framework for improving classifiers while protecting the data for training [181].

In recent years numerous research studies have been conducted in the healthcare field with particular attention to the application of Blockchain technologies, and there is a growing demand for Blockchain developments and the traditional industry is actively exploring new avenues for using the Blockchain to meet its critical needs [182, 1].

In the field of public health, a very relevant issue in terms of the damage caused by medical misconceptions is represented by the spread of fake information in the health domain, so much so that approaches using Description Logics (DLs) to explain inconsistencies have been developed to address the topic [183].

The phenomenon of fake data in publicly available AI training datasets is disturbing several sectors, from healthcare to autonomous vehicles [184, 185, 186]. Authors have identified the importance of data preserving with Blockchain to mitigate the risk of model poisoning with fake data introduction. However, a real Blockchain implementation based on Smart Contracts has not been provided.

Differently from the above mentioned recent scientific initiatives, this paper proposes a practical implementation of how Blockchain and Smart Contracts can be adopted to implement a trusted distributed system aimed at enhancing the entire AI process by protecting data from alteration and tampering, eliminating the risk of unusable and imprecise results, demonstrating that performance overhead introduced by the new security layer is acceptable in terms of time and resource consumption.

5.4.2 Motivations

Blockchain innovation offers compelling tools also in the healthcare sector in terms of access and data security, allowing the definition of an efficient system in which the information exchanged is authentic and certified. The development of a suitable blockchain-based risk analysis system is aimed at supporting remote healthcare and providing cutting-edge telemedicine services.

The analysis of appropriate databases is essential in this sense to validate the control of the medical treatment approach that takes into account as much as possible the adverse conditions and comorbidity between diseases and therapies in situations of co-administration of drugs. In addition, the ability to analyze clinical information deriving from big data

processing following onward steps of dematerialization of information content, healthcare digitization, and remote management of controls and treatment plans is configured as an essential centerpiece of the innovative approach proposed in this paper.

The autonomous verification project of health treatment is specifically focused on drug safety and efficacy evaluation in case of different may be concomitant, pathologies and on the identification of non-evident relationships between clinical parameters. The validation process is based on the application of advanced technical-scientific methods for the analysis of information consensually provided to a shared database. We point out that of course, from a clinical and diagnostic point of view, the risk analysis system described in this paper is meant for serving and enhancing the quality of health care and is configured as a mere tool for validation, comparison and support that cannot be separated from integration with the hypothetical-deductive clinical approach of medical specialists.

In this paper we focused on Reinforcement Learning via Markov Decision Process formulation, specifically we consider an agent acting in his environment motivated by the achievement of the maximum individual objective by appropriate incentives. The environment is, therefore, defined as space where the agent can take actions, get rewards, and learn, to reach the optimal stage which guarantees the achievement of maximum rewards over time. Reinforcement Learning is the area of AI concerned with how intelligent agents ought to take the most profitable actions based on training the agents themselves to account for observations and rewards from the environment until the task goal is fulfilled. For a comprehensive discussion on Reinforcement Learning, please refer to [187].

5.4.3 Design

The health database derives from the collection of clinical information relating to a group of patients involved in the clinical trial planned for IoT Smart SRL in support of Remote Healthcare [180], and from the procedure of digitalization and secure delivery over the air of telematics data coming from clinical laboratory test reports.

The data from the measurements of health parameters carried out using remote monitoring devices are collected through the smartphone app named “T-Check”, developed within and for the Italian PON project “TALIsMAn” [188], and are available, together with all the data of interest relating to the patients involved in the trial, on a web platform also developed in the aforementioned context. The existing IoT-based big data storage system offers the possibility to manage the entire amount of data in such a way to allow information to be collected in an integral and therefore totally non-flawed manner.

Q-learning in MDP Environment

Training environment for an actor may be expressed using a Markov Decision Processes (MDP) in terms of state-dependent quantities accounting for future effects of actions on immediate and subsequent situations. Suitably defined states, actions and rewards are connected throughout some deterministic function completely characterizing the dynamics of the model. We consider two different Markov decision process models whose states and actions are referred to the doctor's and patient's point of view. For a schematic view to the diagrams describing the dynamics of the processes see Fig. 5.13 and Fig. 5.14, respectively. The interpretative approach used derives from few general assumptions on methods of diagnosis and therapeutic approach in medical practice, on the one hand, and self-regulating behavior in remote management of personal healthcare, on the other side. According to the best practices, doctors carrying out some clinical investigation by requesting clinical analysis are used to start from an idea of presumed diagnosis based on which received results will be interpreted in search of a confirmation or denial of the hypotheses that were made *a priori*. Concerning clinical investigations through laboratory tests, a preliminary survey on hospital habits shows how some departments are used to require comprehensive screening, resulting in a large amount of data, while others carry out the checks in a more delayed manner, so leaving more leeway to the clinical monitoring of the time evolution of the patient's health status. The subsequent follow-up, which entails that clinical data are kept in memory for about ten years (albeit to fulfill legal obligations), also provides a long-term validation mean for the analysis methodology. We focused the attention on how, in both scenarios, clinical investigations using laboratory tests of various kinds, with possible expansion of the timing, although constituting a tool of fundamental importance and undoubted validity, are to be inevitably associated with both their specific cost, sometimes onerous, that the hospital structure has to bear and with the physical/psychological stress that the patient who undergoes them has to endure. In this sense, an appropriate and certified cost-benefit assessment that strengthens the justification of the need and effectiveness of diagnostic methodologies can constitute a valid support tool both for the structure and for the fragile individual.

Doctor's MDP The actor is the doctor, and the overall goal is to enhance the management of remote care to improve the patient's general health conditions in such a way to contain current expenses for the hospital, waiting times and stress for involved fragile subjects. We take into account two different states: M ("monitoring") and I ("investigating"); the possible actions the doctor does are: d ("diagnose"), u ("update" the protocol of care), and w ("wait" or leave

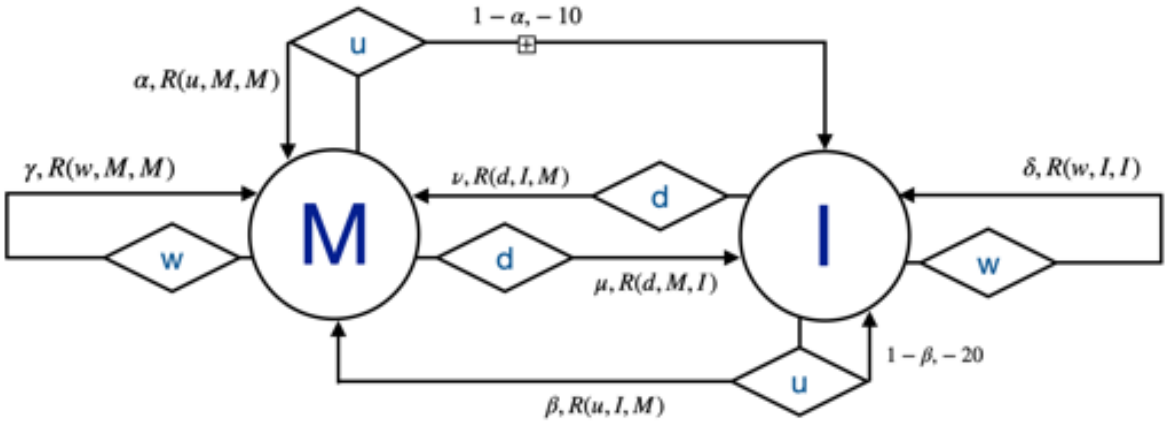


Figure 5.13: Transition graph for doctor's MDP.

the same treatment protocol). The doctor can make a diagnosis starting from observation, say when the actor's state is M, rather than investigation through clinical examinations, that is when the actor's state is I. Positive rewards are associated with the chances of moving to the other state from the current one as a consequence of the action performed or staying until the next action. In the case of the model shown in Fig. 5.13, penalties are increasingly associated with two scenarios: the least disadvantageous situation is represented by the case in which the doctor, who was monitoring the patient's condition in state M, decides to update the protocol while proceeding to investigate with clinical laboratory tests (the actor enters the state I); the most unfavorable scenario, on the other hand, consists of subsequent updates of the therapeutic protocol while staying in the investigation state I without interspersing a period of observation of the evolution of the clinical scenario (the actor proceeds by groping). Train the agent provided the above keys of interpretation is aimed at helping the medical staff to find the best policy to maximize the cumulative reward over time when some action is taken in each specific state.

Based on the above-defined model, the following transition matrix Table 5.5 describes the states change for the Doctor's MDP. For convention, a score below 0.5 represents a negative reward, whilst any value above 0.5 included represents a positive reward.

Table 5.5: Transition Probabilities and Rewards $(P(s,a,s'), R(s,a,s'))$ associated to state change in correspondence to possible actions for the Doctor's MDP.

		(Current → Next)			
	States	**M → M**	**M → I**	**I → M**	**I → I**
Action	d	(0, 0)	(1, 5)	(1, 10)	(0, 0)
	u	(0.7, 3)	(0.3, -10)	(0.8, 4)	(0.2, -20)
	w	(1, 2)	(0, 0)	(0, 0)	(1, 1)

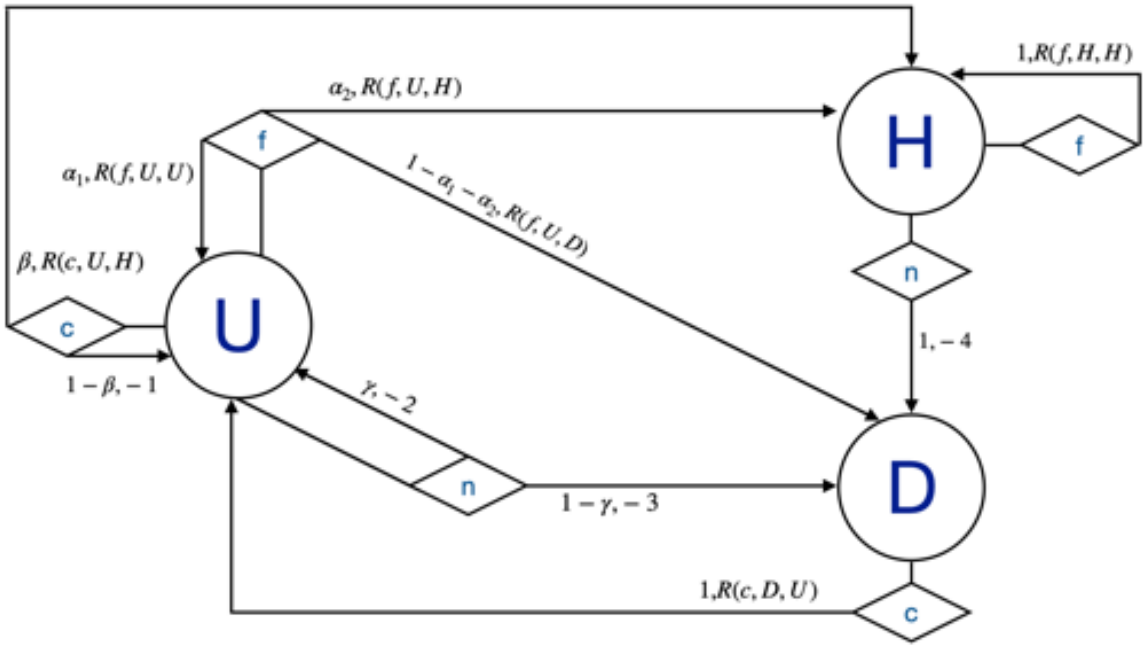


Figure 5.14: Transition graph for patient's MDP.

Patient's MDP In this scenario, the actor is the patient, and the overall goal is the enhancement of one's personal health by improving self-care skills supported by e-health technologies and remote monitoring devices. We take into account three different states: U ("under treatment"), H ("healthier"), D ("declining", *i.e.* the approach is unsuccessful and patient's health is worsening); the possible actions for the patient are: f ("follows" the course of treatment), n ("neglects" the prescriptions), c ("contacts" for supporting the medical staff). Positive rewards are associated with the chances of gaining an improved health evaluation, moving from a treatment to a healthier condition. A disadvantageous situation is represented by the case in which the clinical condition worsens and if the treatment is not effective or in the case of other concomitant pathologies. This scenario is summarized in the model shown in Fig. 5.14.

Based on the above-defined model, the following transition matrix Table 5.6 describes the states change for the Patient's MDP. For convention, a score below 0.3 represents a negative reward, whilst any value above 0.3 included represents a positive reward.

Blockchain-oriented design and execution of clinical trials

The system aims to guarantee the traceability and accountability of all the states changes. To achieve such a goal, the Blockchain engine is responsible to store information on the Blockchain to certify data non-repudiation and immutability, guaranteeing accountability

Table 5.6: Transition Probabilities and Rewards $(P(s,a,s'), R(s,a,s'))$ associated to state change in correspondence to possible actions for the Patient's MDP schematized in Fig. 5.14

(Transition Probability, Reward)	
State	**Score**
$(\alpha_1, R(f,U,U)$	(0.4, 6)
$(\alpha_2, R(f,U,H)$	(0.5, 8)
$(\beta, R(c,U,H))$	(0.2, 2)
$(\gamma, R(n,U,U))$	(0.3, -2)
$(1-\alpha_1-\alpha_2, R(f,U,D)$	(0.1, 1)
$(1-\beta, R(c,U,U)$	(0.8, -1)
$(1, R(c,D,U))$	(1, 4)
$(1, R(f,H,H))$	(1, 10)

and authenticity, through the development and coding of a Smart Contract.

This approach permits to guarantee the trustiness of the developed logic, that can be accessible and verifiable by any hospital, pharmaceutical or insurance company that requires a secure and trusted environment to validate the proper treatment recommendation are scrupulously followed. In fact, as only trusted actors can update information in the Blockchain, the risk of fake input AI data is drastically reduced.

The system is organized in the following phases:

1. **Define the transition matrix for a model:** A group of selected MD defines the set of rules, scores and rewards for the treatment of a selected group of patients for the trials.

2. **Smart Contract execution:** Each transition is transparently and automatically calculated by the Smart Contract. It is signed with the private key (for data non-repudiation and accountability for any future investigation) of the MD that follows a patient during the treatment and it is immutably recorded in the Blockchain.

3. **Data retrieval and analysis:** At any time, the hospital or any partner can access the data to analyze and improve procedures and methods based on the certified clinical trials.

Fig. 5.15 describes the architecture implemented.

This approach can be implemented with either Public or Private Blockchains, such as Ethereum or Hyperledger Fabric because the maturity of the network and the Smart Contract capabilities of both the instances permit to obtain the desired achievements.

5.4.4 Case study in practice

In this section, the implementation of the Blockchain-based Reinforcement Learning protocol is described. The development is based on the private Blockchain network Hyperledger

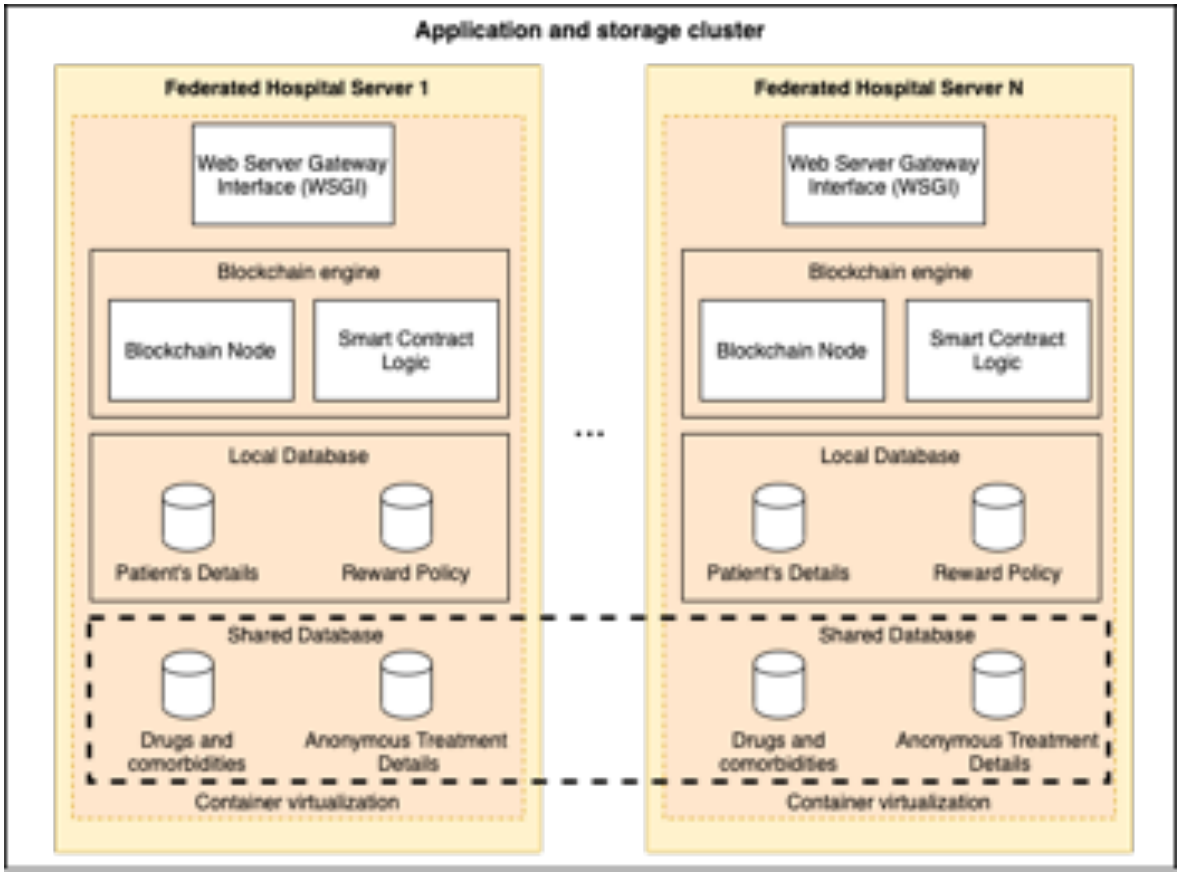


Figure 5.15: Blockchain-based Reinforcement Learning architecture implemented.

Fabric, to enhance trust in the data and the system because only authorized and certified users can participate to submit transactions. However, it is required that the number of nodes participating in the network is sufficiently large to guarantee resistance to distributed attacks.

The entire implementation is based on microservice through the programming language Python 3, the Flask framework and the Gunicorn Python Web Server Gateway Interface (WSGI) HTTP server. Furthermore, each microservice is deployed within a Docker container to take the advantage of the virtualization technology allowing service portability, resiliency, and automatic updates that are typical of the distributed system acting at the Cloud or the Edge [102].

The two different MDPs are based on two Smart Contract applications. The *Doctor MDP* is responsible for attributes a reward, being it positive or negative, to the Doctor's decision, validating the transaction states considering as input the doctor id, the patient id and the destination state. The pseudo-code of this implementation is provided in Algorithm 5.

The *Patient MDP* attributes a reward to the Patient based on the transaction states using as input the patient id and the destination state. The pseudo-code of this implementation is provided in Algorithm 6. This process is separate from the Doctor MDP and the services

can be activated on-demand based on hospital needs. The *Blockchain Engine* component is implemented with Hyperledger Fabric Private Blockchain, but the same architecture can be deployed with Ethereum Public Blockchain according to the specific application scenario and security policy.

The Private network implementation uses Fabric 2.2 with an architecture based on four peer nodes distributed in four remote servers and four orderer nodes. The peer node receives updates and broadcasts them to the other peers in the network. The orderer node is responsible for consistent Ledger state across the network, it creates the block and delivers that to all the peers.

Algorithm 5 Doctor MDP reward Algorithm Pseudo-code

Class *DoctorReward*
method $moveToState(doctor\_id, patient\_id, to\_state)$
  $current\_state \leftarrow \emptyset$
  $reward \leftarrow \emptyset$
  $current\_state \leftarrow getCurrentState(patient\_id)$
  $reward \leftarrow getDoctorRewardforTransition(current\ \_state, to\_state)$
  $updateState(patient\_id, to\_state)$
  $updateDoctorReward(doctor\_id, reward)$

Algorithm 6 Patient MDP reward Algorithm Pseudo-code

Class *PatientReward*
method $moveToState(patient\_id, to\_state)$
  $current\_state \leftarrow \emptyset$
  $reward \leftarrow \emptyset$
  $current\_state \leftarrow getCurrentState(patient\_id)$
  $reward \leftarrow getPatientRewardforTransition(current\ \_state, to\_state)$
  $updateState(patient\_id, to\_state)$
  $updatePatientReward(patient\_id, reward)$

5.4.5 Experiments

The system prototype assessment was conducted analyzing the overhead introduced by the Blockchain considering the implementation with both private and public Blockchain approaches. Specifically, we tested and compared the Smart Contract execution time and resources usage in terms of CPU. It is important to consider that, with the private Blockchain approach, each transaction executed does not include any cost in terms of money or cryptocurrency. Whereas, with the public Blockchain approach, every information is sent to a public Blockchain node to be added to the queue for mining and immutable storage and it is subject to a fee. Time-to-mine and Ethereum (ETH) cryptocoin currency transaction costs are

subject to Ethereum network traffic. It is important to consider that the cost of a transaction in Ethereum is calculated based on the operations required and the complexity of the Smart Contract. These are the element to be considered to calculate the gas used for a transaction, and the cost to be paid is derived from these.

The testbed was arranged using a remote server with the following features: Intel® Xeon® E3-12xx v2 @ 2.7GHz, 2 core CPU, 4 GB RAM running Ubuntu Server 18.04. For private Blockchain, tests have been performed with 2 CLI, 4 peers and 4 orderers distributed across 4 different hosts. For public Blockchain, all tests have been performed using Ropsten Ethereum public Blockchain test network, leveraging 300+ available nodes with a real server load status. Tests have been executed for 100, 200 and 300 read and write operations, considering 95% confidence intervals and the average values. It must be considered that the Ethereum Blockchain Ropsten environment is based on Proof of Work (PoW) consensus protocol which makes it difficult to obtain scalability and system speed. A summary of experiments setups is reported in Table 5.7.

Table 5.7: Summary of experiments performed.

Parameter	**Value**
Contract address	0xa1032b39180538D7e17 23AedC39154C89A113BE5
Test executed for each scenario	[100, 200, 300]
Confidence interval	95%
Gas used by transaction	32,000
Gas price (Gwei)	200
Average cost per transaction (ETH)	0.0064967

Figure 5.16 shows the execution time difference expressed in seconds between the two system implementations to complete a single Markov Decision Processes (MDP). On the x-axis, it is reported the two approaches for private and public Blockchain. On the y-axis it is reported the processing time is expressed in seconds. As expected, it is possible to observe that the time required to validate the MDP is considerably greater in the Public Blockchain approach, due to the mining time required to commit the transaction in the Public Blockchain.

Figure 5.17 shows the execution time difference expressed in seconds between the two system implementations to complete Doctor's and Patient's MDP. On the x-axis, it is reported the differences to perform 100, 200 and 300 requests. On the y-axis it is reported the processing time is expressed in seconds. From the graph, it is possible to observe that the time required to complete the processes is largely different in the two implementations and raise with a linear behavior. This is due to the mining time required to store the data in the public Blockchain

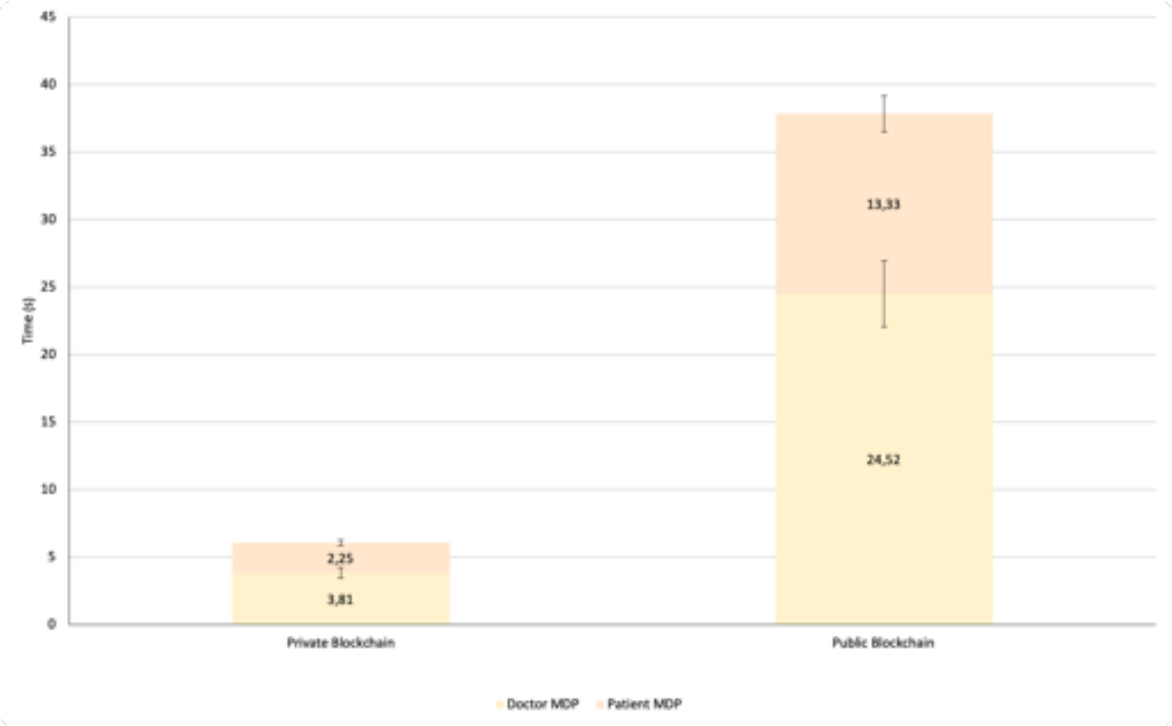


Figure 5.16: Execution time comparison for a single Markov Decision Processes (average) between private and public Blockchain.

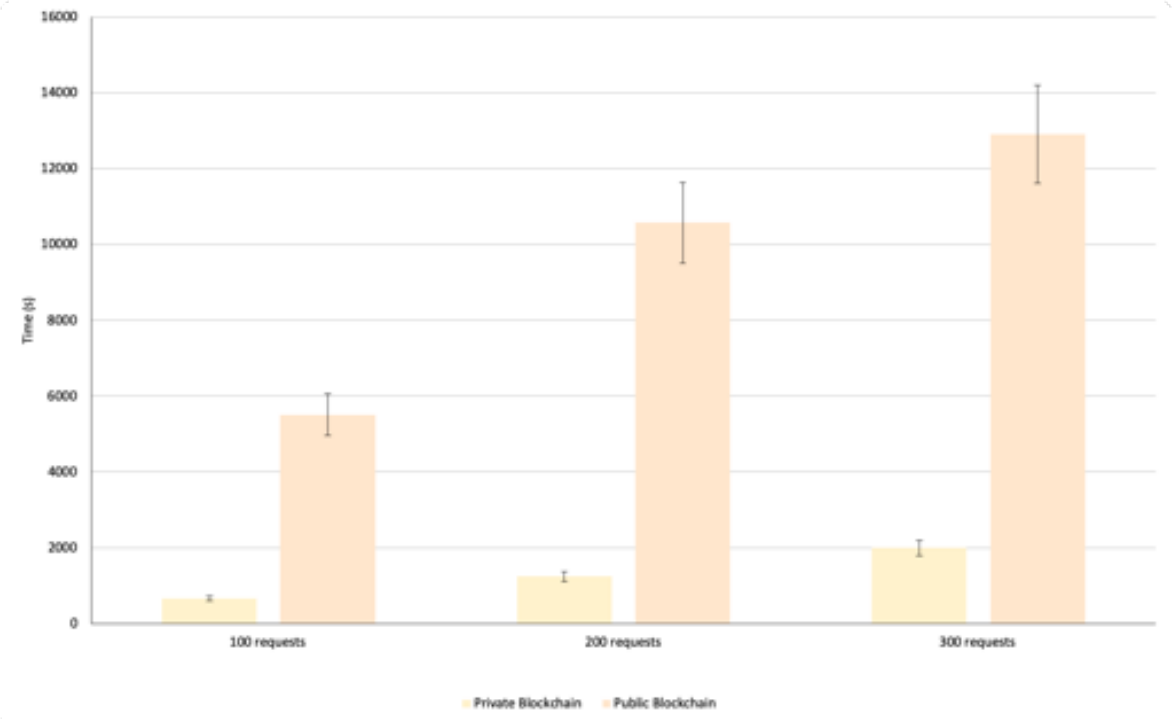


Figure 5.17: Execution time comparison between private and public Blockchain implementations for a Markov Decision Processes execution.

that is considerably different from the private Blockchain approach.

Figure 5.18 shows the CPU usage % difference between the two system implementations

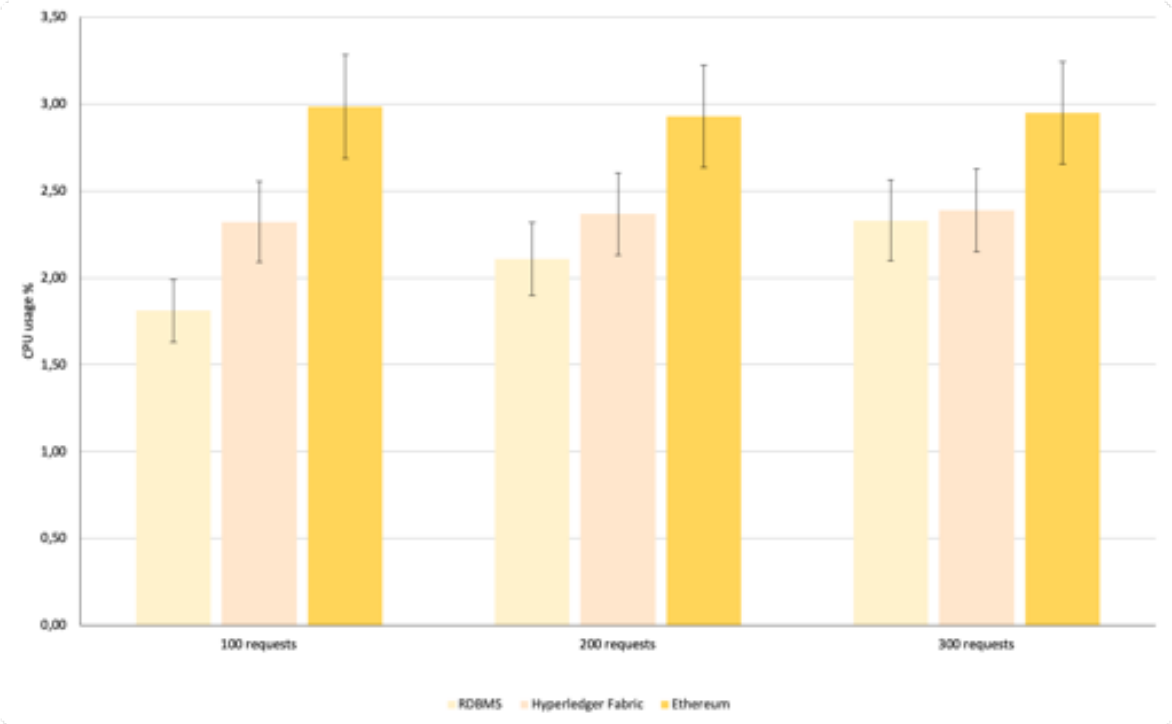


Figure 5.18: CPU usage % comparison between Private Blockchain and Public Blockchain approach implementations for a Markov Decision Processes execution.

for the OTP generation. On the x-axis, it is reported the differences to perform 100, 200 and 300 requests. On the y-axis it is reported the CPU usage %. It is possible to observe three different scenarios: a server that does not use the Blockchain, but stores the seed in a local MySQL DBMS instance that is compared with the private and public Blockchain implementations. It can be appreciated that the CPU usage % is kept in low range values and it is comparable in the three implementations.

5.4.6 Some remarks

The solution has been prototyped and tested with both private and public Blockchain approaches. In the first case, although the trusted network needs to be built with a reasonable number of nodes to guarantee sufficient level availability and reliability, we observed low response times. Instead, in the second case, considering a high level of availability and reliability thanks to the larger number of nodes, we observed an acceptable degradation of response times.

Experimental results highlighted that the CPU usage % remains roughly unchanged if using a DBMS such as MySQL to store the information, a private or public Blockchain, demonstrating that this approach can be implemented in any scenario without any performance degradation in terms of CPU usage.

5.5 Overall consideration

This Chapter analyses the potentiality of the Blockchain technology in the healthcare domain, from personnel accountability to data certification. Section 5.1 demonstrates how a Decision Support System (DSS) can be the key to success in medical prevention.

Specifically, after analyzing the results of clinical analysis, medical doctor prepares a report with the therapy to be followed, and the DSS algorithm automatically evaluates if the indicated therapy can cause undesirable effects due to allergies or intolerances, as well as the therapy in similar clinical cases have caused deaths. If some incompatibility is detected, the DSS system can propose, for example, the automatic setup of a remote consulting session among selected physician.

Sections 5.2 and 5.3 propose a new comprehensive Healthcare workflow to ensure every step is scrupulously follow.

In particular, a federation of hospitals can cooperate to form a virtual healthcare team able to carry out a healthcare workflow, guaranteeing the non-repudiation and immutability of all health decisions trough a Federated Blockchain network.

Finally, Section 5.4 demonstrates how a Markov Decision Process (MDP) can be applied jointly with Blockchain to enforce data quality to avoid fake data in Artificial Intelligence training, using two different Smart Contract, one for Medical MDP and one for Patient MDP. This ensures that only authorized and certified users can participate in the network to submit data in the form of transactions.

It is worth to remark that, although Blockchain transaction writing comes with a cost, this is considered negligible as compared to the cost of preparing, configuring and maintaining a secure network infrastructure. In fact, the described solutions and prototypes can perform over 50,000 transactions spending as much as a basic server configuration.

CHAPTER 6

Conclusion and Future Works

This thesis discusses innovative systems for Smart Cities. It is divided in two parts, the first part analyses the basic requirements and infrastructure; the second part provides solutions for specific use-cases. In particular, it analyzes three aspects: Chapter 3 focuses on the importance of IoT devices in a real Smart City scenario, from a comparison of a traffic monitoring camera and a non-expensive Raspberry Pi equipped with a cheap webcam to monitor the traffic in a metropolitan city during peak hours. Furthermore, the potentiality of low-resource device reconfiguration in Edge-IoT mesh networks is analyzed, obtaining the final goal of implementing a new approach for service orchestration based on Blockchain, able to modify the behavior of IoT devices and microservices on demand with a secured and anti-tampering approach.

The second aspect of this thesis is discussed in Chapter 4: security in public connections has always been a key pillar in human and device communication, and researches have been conducted to identify new strategies and protocols to increase it through the use of Blockchain technology. In particular, new Blockchain-based protocols are proposed to guarantee a secure One Time Password (OTP) generation with a seed stored in the network, and an improved version of the Extended Triple Diffie-Hellman Protocol, extended for low resource and general IoT devices that are usually limited in spendable resources to generate secret keys, with a comprehensive analysis of the energy consumed using an energy sensor.

Chapter 5 concludes the thesis describing the contributions of the most recent research activities in the healthcare scenario, with focus on the use of a decision support system to

improve prediction of clinical analyses, and an innovative healthcare workflow supported by Blockchain technology integrated with a data certification system. Finally, with the objective to mitigate the risk of fake data in machine learning training models, a Markov Decision Process has been prototyped and discussed.

Bibliography

[1] **Armando Ruggeri**, Maria Fazio, Antonio Celesti, and Massimo Villari. Blockchain-based healthcare workflows in federated hospital clouds. In *European Conference on Service-Oriented and Cloud Computing*, pages 113–121. Springer, 2020. (Cited at pages x, 4, 77, 80, 112, 124, 136 e 179)

[2] Antonio Celesti, **Armando Ruggeri**, Maria Fazio, Antonino Galletta, Massimo Villari, and Agata Romano. Blockchain-based healthcare workflow for tele-medical laboratory in federated hospital iot clouds. In *Blockchain in the Internet of Things: Opportunities, Challenges and Solutions*. MDPI, 2020. (Cited at pages x, 4 e 121)

[3] **Armando Ruggeri**, Maria Fazio, Antonino Galletta, Antonio Celesti, and Massimo Villari. A decision support system for therapy prescription in a hospital centre. In *2020 IEEE Symposium on Computers and Communications (ISCC)*, pages 1–4, 2020. (Cited at pages x e 4)

[4] Antonino Galletta, **Armando Ruggeri**, Maria Fazio, Gianluca Dini, and Massimo Villari. Mesmart-pro: Advanced processing at the edge for smart urban monitoring and reconfigurable services. *Journal of Sensor and Actuator Networks*, 9(4), 2020. (Cited at pages x e 3)

[5] **Armando Ruggeri**, Antonio Celesti, Maria Fazio, Antonino Galletta, and Massimo Villari. Bcb-x3dh: a blockchain based improved version of the extended triple diffie-hellman protocol. In *2020 Second IEEE International Conference on Trust, Privacy and Security in Intelligent Systems and Applications (TPS-ISA)*, pages 73–78, 2020. (Cited at pages x e 4)

[6] **Armando Ruggeri**, Antonino Galletta, Antonio Celesti, Maria Fazio, and Massimo Villari. An innovative blockchain based application of the extended triple diffie-hellman protocol for iot. In *2021 8th International Conference on Future Internet of Things and Cloud (FiCloud)*, pages 278–284, 2021. (Cited at page xi)

[7] **Armando Ruggeri**, Antonio Celesti, Maria Fazio, and Massimo Villari. An innovative blockchain-based orchestrator for osmotic computing. *Journal of Grid Computing*, 20(1):1–17, 2022. (Cited at pages xi e 3)

[8] Alessio Catalfamo, **Armando Ruggeri**, Antonio Celesti, Maria Fazio, and Massimo Villari. A microservices and blockchain based one time password (mbb-otp) protocol for security-enhanced authentication. In *2021 IEEE Symposium on Computers and Communications (ISCC)*, pages 1–6. IEEE, 2021. (Cited at pages xi e 3)

[9] **Armando Ruggeri**, Rosa Di Salvo, Maria Fazio, Antonio Celesti, and Massimo Villari. Blockchain-based strategy to avoid fake ai in ehealth scenarios with reinforcement learning. In *2021 IEEE Symposium on Computers and Communications (ISCC)*, pages 1–7, 2021. (Cited at pages xi e 5)

[10] Lorenzo Carnevale, **Armando Ruggeri**, Francesco Martella, Antonio Celesti, Maria Fazio, and Massimo Villari. Multi hop reconfiguration of end-devices in heterogeneous edge-iot mesh networks. In *2021 IEEE Symposium on Computers and Communications (ISCC)*, pages 1–6. IEEE, 2021. (Cited at pages xi e 3)

[11] **Armando Ruggeri** and Massimo Villari. Improving the key exchange process of the extended triple diffie-hellman protocol with blockchain. In *European Conference on Service-Oriented and Cloud Computing*. (Accepted), 2022. (Cited at page xi)

[12] **Armando Ruggeri**, Antonino Galletta, Lorenzo Carnevale, and Massimo Villari. An energy efficiency analysis of the blockchain-based extended triple diffie-hellman protocol for iot. In *2022 IEEE Symposium on Computers and Communications (ISCC)*. (Accepted), 2022. (Cited at page xi)

[13] Armando Ruggeri, Antonio Celesti, Antonino Galletta, Maria Fazio, and Massimo Villari. An innovative blockchain based application of the extended triple diffie-hellman protocol for iot. In *2021 8th International Conference on Future Internet of Things and Cloud (FiCloud)*. IEEE, 2021. (Cited at pages 4 e 139)

[14] INA219, Bidirectional Current/Power Monitor With I2C Interface, Available online:`https://www.rototron.info/wp-content/uploads/INA219_datasheet.pdf` (accessed on 22 July 2021). (Cited at page 5)

[15] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system, 03 2009. (Cited at pages 11, 76, 111 e 170)

[16] Thanmayee Rao. 51% attacks on cryptocurrencies: A case study, 05 2019. (Cited at pages 24 e 81)

[17] Basilio Filocamo, Antonino Galletta, Maria Fazio, Javier Alonso Ruiz, Miguel Ángel Sotelo, and Massimo Villari. An innovative osmotic computing framework for self adapting city traffic in autonomous vehicle environment. In *2018 IEEE Symposium on Computers and Communications (ISCC)*, pages 01267–01270. IEEE, 2018. (Cited at page 34)

[18] V. S. Nagmode and S. M. Rajbhoj. An iot platform for vehicle traffic monitoring system and controlling system based on priority. In *2017 International Conference on Computing, Communication, Control and Automation (ICCUBEA)*, pages 1–5, 2017. (Cited at pages 35 e 36)

[19] Manuj Darbari, Diwakar Yagyasen, and Anurag Tiwari. Intelligent traffic monitoring using internet of things (iot) with semantic web. In Suresh Chandra Satapathy, A. Govardhan, K. Srujan Raju, and J. K. Mandal, editors, *Emerging ICT for Bridging the Future - Proceedings of the 49th Annual Convention of the Computer Society of India (CSI) Volume 1*, pages 455–462, Cham, 2015. Springer International Publishing. (Cited at pages 35 e 36)

[20] Sameer Parekh, Nilam Dhami, Sandip Patel, and Jaimin Undavia. Traffic signal automation through iot by sensing and detecting traffic intensity through ir sensors. In Suresh Chandra Satapathy and Amit Joshi, editors, *Information and Communication Technology for Intelligent Systems*, pages 53–65, Singapore, 2019. Springer Singapore. (Cited at pages 35 e 36)

[21] Johan Barthélemy, Nicolas Verstaevel, Hugh Forehead, and Pascal Perez. Edge-computing video analytics for real-time traffic monitoring in a smart city. *Sensors*, 19(9), 2019. (Cited at pages 35 e 36)

[22] A. Mehrabi, M. Siekkinen, and A. Ylä-Jaaski. Qoe-traffic optimization through collaborative edge caching in adaptive mobile video streaming. *IEEE Access*, 6:52261–52276, 2018. (Cited at pages 36 e 37)

[23] S. Y. Nikouei, Y. Chen, S. Song, and T. R. Faughnan. Kerman: A hybrid lightweight tracking algorithm to enable smart surveillance as an edge service. In *2019 16th IEEE Annual Consumer Communications Networking Conference (CCNC)*, pages 1–6, 2019. (Cited at pages 36 e 38)

[24] Y. Zhao, Y. Yin, and G. Gui. Lightweight deep learning based intelligent edge surveillance techniques. *IEEE Transactions on Cognitive Communications and Networking*, pages 1–1, 2020. (Cited at pages 36 e 38)

[25] Maria Fazio, Maurizio Paone, Antonio Puliafito, and Massimo Villari. Huge amount of heterogeneous sensed data needs the cloud. In *International Multi-Conference on Systems, Signals & Devices*, pages 1–6. IEEE, 2012. (Cited at pages 36 e 40)

[26] L. U. Khan, I. Yaqoob, N. H. Tran, S. M. A. Kazmi, T. N. Dang, and C. S. Hong. Edge computing enabled smart cities: A comprehensive survey. *IEEE Internet of Things Journal*, pages 1–1, 2020. (Cited at pages 36 e 77)

[27] X. Xu, Q. Huang, X. Yin, M. Abbasi, M. R. Khosravi, and L. Qi. Intelligent offloading for collaborative smart city services in edge computing. *IEEE Internet of Things Journal*, pages 1–1, 2020. (Cited at pages 37, 77 e 79)

[28] S. S. Kafıloğlu, G. Gür, and F. Alagöz. Cooperative caching and video characteristics in d2d edge networks. *IEEE Communications Letters*, pages 1–1, 2020. (Cited at page 37)

[29] D. Wang, Y. Peng, X. Ma, W. Ding, H. Jiang, F. Chen, and J. Liu. Adaptive wireless video streaming based on edge computing: Opportunities and approaches. *IEEE Transactions on Services Computing*, 12(5):685–697, 2019. (Cited at page 37)

[30] Y. Zhao, Q. Chen, W. Cao, W. Jiang, and G. Gui. Deep learning based couple-like cooperative computing method for iot-based intelligent surveillance systems. In *2019 IEEE 30th Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, pages 1–4, 2019. (Cited at page 38)

[31] M. Liu, F. R. Yu, Y. Teng, V. C. M. Leung, and M. Song. Distributed resource allocation in blockchain-based video streaming systems with mobile edge computing. *IEEE Transactions on Wireless Communications*, 18(1):695–708, 2019. (Cited at page 38)

[32] W. A. Okaishi, I. Atouf, and M. Benrabh. Real-time traffic light control system based on background updating and edge detection. In *2019 International Conference on Wireless Technologies, Embedded and Intelligent Systems (WITS)*, pages 1–5, 2019. (Cited at page 39)

[33] K. Yang, Y. Zhou, S. Han, Y. Fang, X. Ma, J. Zhou, and K. Yao. Video-based traffic flow monitoring algorithm for single phase position at an intersection. In *2019 International Conference on Security, Pattern Analysis, and Cybernetics (SPAC)*, pages 101–105, 2019. (Cited at page 39)

[34] Z. Wang, G. Cheng, and J. Zheng. Road edge detection in all weather and illumination via driving video mining. *IEEE Transactions on Intelligent Vehicles*, 4(2):232–243, 2019. (Cited at page 39)

[35] L. Baresi and D. Filgueira Mendonça. Towards a serverless platform for edge computing. In *2019 IEEE International Conference on Fog Computing (ICFC)*, pages 1–10, 2019. (Cited at pages 39, 77 e 79)

[36] P. K. Gadepalli, G. Peach, L. Cherkasova, R. Aitken, and G. Parmer. Challenges and opportunities for efficient serverless computing at the edge. In *2019 38th Symposium on Reliable Distributed Systems (SRDS)*, pages 261–2615, 2019. (Cited at page 39)

[37] R. F. Hussain, M. A. Salehi, and O. Semiari. Serverless edge computing for green oil and gas industry. In *2019 IEEE Green Technologies Conference(GreenTech)*, pages 1–4, 2019. (Cited at page 39)

[38] A. Palade, A. Kazmi, and S. Clarke. An evaluation of open source serverless computing frameworks support at the edge. In *2019 IEEE World Congress on Services (SERVICES)*, volume 2642-939X, pages 206–211, 2019. (Cited at pages 39, 77 e 79)

[39] D. Mercer. Global connected and iot device forecast update. Technical report, Strategy Analytics, May 2019. (Cited at page 54)

[40] Cisco annual internet report. Technical report, Cisco public, March 2020. (Cited at page 54)

[41] D. Mijić and E. Varga. Unified iot platform architecture platforms as major iot building blocks. pages 6–13, 2018. (Cited at page 57)

[42] Antonio Cilfone, Luca Davoli, Laura Belli, and Gianluigi Ferrari. Wireless mesh networking: An iot-oriented perspective survey on relevant technologies. *Future Internet*, 11(4):99, Apr 2019. (Cited at page 58)

[43] S. Ali, M. Pandey, and N. Tyagi. Wireless-fog mesh: A framework for in-network computing of microservices in semipermanent smart environments. *International Journal of Network Management*, 30(6), 2020. (Cited at page 58)

[44] Xiaofan Jiang, Heng Zhang, Edgardo Alberto Barsallo Yi, Nithin Raghunathan, Charilaos Mousoulis, Somali Chaterji, Dimitrios Peroulis, Ali Shakouri, and Saurabh Bagchi. Hybrid low-power wide-area mesh network for iot applications. *IEEE Internet of Things Journal*, 2020. (Cited at page 58)

[45] Celia Garrido-Hidalgo, Diego Hortelano, Luis Roda-Sanchez, Teresa Olivares, M Carmen Ruiz, and Vicente Lopez. Iot heterogeneous mesh network deployment for human-in-the-loop challenges towards a social and sustainable industry 4.0. *IEEE Access*, 6:28417–28437, 2018. (Cited at page 58)

[46] Archit Gajjar, Xiaokun Yang, Hakduran Koc, Ishaq Unwala, Lei Wu, and Jiang Lu. Mesh-iot based system for large-scale environment. In *2018 International Conference on Computational Science and Computational Intelligence (CSCI)*, pages 1019–1023. IEEE, 2018. (Cited at page 58)

[47] Emanuele Di Pascale, Irene Macaluso, Avishek Nag, Mark Kelly, and Linda Doyle. The network as a computer: A framework for distributed computing over iot mesh networks. *IEEE Internet of Things Journal*, 5(3):2107–2119, 2018. (Cited at page 58)

[48] Mennan Selimi, Adisorn Lertsinsrubtavee, Arjuna Sathiaseelan, Llorenç Cerdà-Alabern, and Leandro Navarro. Picasso: Enabling information-centric multi-tenancy at the edge of community mesh networks. *Computer Networks*, 164:106897, 2019. (Cited at page 59)

[49] Xiliu He and Fang Deng. Research on architecture of internet of things platform based on service mesh. In *2020 12th International Conference on Measuring Technology and Mechatronics Automation (ICMTMA)*, pages 755–759. IEEE, 2020. (Cited at page 59)

[50] Marino Linaje and Enrique Carlos Mesías. A new wsn mesh protocol for more transparent iot devices. In *International Workshop on Gerontechnology*, pages 94–106. Springer, 2018. (Cited at page 59)

[51] D. Huynh-Van, K. Tran-Quoc, and Q. LE-TRUNG. An empirical study on approaches of internet of things reconfiguration. In *2018 IEEE Seventh International Conference on Communications and Electronics (ICCE)*, pages 57–62, 2018. (Cited at page 59)

[52] S. Pareek, S. Shrivastava, S. Jhala, J. A. Siddiqui, and S. Patidar. Iot and image processing based forest monitoring and counteracting system. In *2020 4th International Conference on Trends in Electronics and Informatics (ICOEI)(48184)*, pages 1024–1027, 2020. (Cited at page 60)

[53] S. Awadallah, D. Moure, and P. Torres-González. An internet of things (iot) application on volcano monitoring. *Sensors*, 19(21):4651, Oct 2019. (Cited at page 60)

[54] L. Terray, L. Royer, D. Sarramia, C. Achard, E. Bourdeau, P. Chardon, A. Claude, J. Fuchet, PJ Gauthier, D. Grimbichler, and et al. From sensor to cloud: An iot network of radon outdoor probes to monitor active volcanoes. *Sensors*, 20(10):2755, May 2020. (Cited at page 60)

[55] Glusterfs official documentation. `https://docs.gluster.org`. Last access: December 29, 2020. (Cited at page 63)

[56] Kumarnda Arun Bhukya, Somula Ramasubbareddy, K Govinda, and T Aditya Sai Srinivas. Adaptive mechanism for smart street lighting system. In *Smart Intelligent Computing and Applications*, pages 69–76. Springer, 2020. (Cited at page 73)

[57] Zhiwen Hu, Zixuan Bai, Yuzhe Yang, Zijie Zheng, Kaigui Bian, and Lingyang Song. Uav aided aerial-ground iot for air quality sensing in smart city: architecture, technologies, and implementation. *IEEE Network*, 33(2):14–22, 2019. (Cited at page 73)

[58] Jongmin Yim, Rafael Angelo Cadiente, Gian Paolo Mayuga, and Elmer R Magsino. Integrated plate recognition and speed detection for intelligent transportation systems. In *2020 IEEE 10th Symposium on Computer Applications & Industrial Electronics (ISCAIE)*, pages 6–11. IEEE, 2020. (Cited at page 73)

[59] Toby Velte, Anthony Velte, and Robert Elsenpeter. *Cloud computing, a practical approach*. McGraw-Hill, Inc., 2009. (Cited at page 74)

[60] Flavio Bonomi, Rodolfo Milito, Jiang Zhu, and Sateesh Addepalli. Fog computing and its role in the internet of things. In *Proceedings of the first edition of the MCC workshop on Mobile cloud computing*, pages 13–16, 2012. (Cited at page 74)

[61] Weisong Shi, Jie Cao, Quan Zhang, Youhuizi Li, and Lanyu Xu. Edge computing: Vision and challenges. *IEEE internet of things journal*, 3(5):637–646, 2016. (Cited at page 74)

[62] Ioana Baldini, Paul Castro, Kerry Chang, Perry Cheng, Stephen Fink, Vatche Ishakian, Nick Mitchell, Vinod Muthusamy, Rodric Rabbah, Aleksander Slominski, et al. Serverless computing: Current trends and open problems. In *Research Advances in Cloud Computing*, pages 1–20. Springer, 2017. (Cited at page 74)

[63] M. Villari, M. Fazio, S. Dustdar, O. Rana, D. N. Jha, and R. Ranjan. Osmosis: The osmotic computing platform for microelements in the cloud, edge, and internet of things. *Computer*, 52(8):14–26, 2019. (Cited at page 74)

[64] Mohammad Shahrad, Jonathan Balkind, and David Wentzlaff. Architectural implications of function-as-a-service computing. In *Proceedings of the 52nd Annual IEEE/ACM International Symposium on Microarchitecture*, pages 1063–1075, 2019. (Cited at page 75)

[65] L. Baresi and D. Filgueira Mendonça. Towards a serverless platform for edge computing. In *2019 IEEE International Conference on Fog Computing (ICFC)*, pages 1–10, 2019. (Cited at page 75)

[66] Avijit Mallik, A Ahsan, M Shahadat, and J Tsou. Man-in-the-middle-attack: Understanding in simple words. *International Journal of Data and Network Science*, 3(2):77–92, 2019. (Cited at pages 75, 78, 112 e 138)

[67] Kriti Bhushan and Brij B Gupta. Distributed denial of service (ddos) attack mitigation in software defined network (sdn)-based cloud computing environment. *Journal of Ambient Intelligence and Humanized Computing*, 10(5):1985–1997, 2019. (Cited at pages 75, 78, 113 e 138)

[68] Atakan Aral, Ivona Brandic, Rafael Brundo Uriarte, Rocco De Nicola, and Vincenzo Scoca. Addressing application latency requirements through edge scheduling. *Journal of Grid Computing*, 17(4):677–698, 2019. (Cited at pages 77 e 79)

[69] Mingli Wu, Kun Wang, Xiaoqin Cai, Song Guo, Minyi Guo, and Chunming Rong. A comprehensive survey of blockchain: From theory to iot applications and beyond. *IEEE Internet of Things Journal*, 6(5):8114–8154, 2019. (Cited at page 77)

[70] Gowri Sankar Ramachandran and Bhaskar Krishnamachari. Blockchain for the iot: Opportunities and challenges. *arXiv preprint arXiv:1805.02818*, 2018. (Cited at pages 77 e 79)

[71] Ali Hassan Sodhro, Sandeep Pirbhulal, Muhammad Muzammal, and Luo Zongwei. Towards blockchain-enabled security technique for industrial internet of things based decentralized applications. *Journal of Grid Computing*, pages 1–14, 2020. (Cited at pages 77, 80 e 136)

[72] Peng Zhang, Jules White, Douglas Schmidt, Gunther Lenz, and S. Rosenbloom. Fhirchain: Applying blockchain to securely and scalably share clinical data. *Computational and Structural Biotechnology Journal*, 16, 07 2018. (Cited at pages 77, 80, 136 e 147)

[73] Petar Kochovski, Vlado Stankovski, Sandi Gec, Francescomaria Faticanti, Marco Savi, Domenico Siracusa, and Seungwoo Kum. Smart contracts for service-level agreements in edge-to-cloud computing. *Journal of Grid Computing*, pages 1–18, 2020. (Cited at pages 77 e 80)

[74] Armando Ruggeri, Antonio Celesti, Maria Fazio, Antonino Galletta, and Massimo Villari. Bcb-x3dh: a blockchain based improved version of the extended triple diffie-hellman protocol. In *2020 Second IEEE International Conference on Trust, Privacy and Security in Intelligent Systems and Applications (TPS-ISA)*, pages 73–78. IEEE, 2020. (Cited at pages 77, 80, 123, 128 e 135)

[75] Jing Liu and Zhentian Liu. A survey on security verification of blockchain smart contracts. *IEEE Access*, 7:77894–77904, 2019. (Cited at pages 77 e 112)

[76] Ruchi Vishwakarma and Ankit Kumar Jain. A survey of ddos attacking techniques and defence mechanisms in the iot network. *Telecommunication Systems*, 73(1):3–25, 2020. (Cited at pages 78, 111, 113 e 122)

[77] Shengbao Zheng and Xiaowei Yang. Dynashield: reducing the cost of ddos defense using cloud services. In *11th {USENIX} Workshop on Hot Topics in Cloud Computing (HotCloud 19)*, 2019. (Cited at pages 78, 111, 113 e 122)

[78] Chad Perrin. The cia triad. *Dostopno na: http://www. techrepublic. com/blog/security/the-cia-triad/488*, 2008. (Cited at pages 78, 111 e 178)

[79] Ethereum, Available online: https://www.ethereum.org/ (accessed on 11 June 2020). (Cited at pages 84 e 116)

[80] Hyperledger Fabric, Available online: https://www.hyperledger.org/use/fabric/ (accessed on 11 June 2020). (Cited at pages 84 e 116)

[81] Hyperledger Sawtooth, Available online: https://www.hyperledger.org/use/sawtooth/ (accessed on 11 June 2020). (Cited at pages 84 e 116)

[82] Lorenzo Carnevale, Antonio Celesti, Antonino Galletta, Schahram Dustdar, and Massimo Villari. From the cloud to edge and iot: a smart orchestration architecture for enabling osmotic computing. In *2018 32nd International Conference on Advanced Information Networking and Applications Workshops (WAINA)*, pages 419–424, 2018. (Cited at page 84)

[83] Infura, Available online: https://infura.io/ (accessed on 20 May 2021). (Cited at pages 85, 116 e 140)

[84] Kai Hu, Jian Zhu, Yi Ding, Xiaomin Bai, and Jiehua Huang. Smart contract engineering. *Electronics*, 9(12), 2020. (Cited at page 86)

[85] Ethereum Charts and Statistics, Available online: `https://etherscan.io/charts` (accessed on 15 March 2021). (Cited at page 86)

[86] Ulf Melin, Karin Axelsson, and Fredrik Söderström. Managing the development of e-id in a public e-service context. *Transforming Government: People, Process and Policy*, 2016. (Cited at page 97)

[87] Thanasis Petsas, Giorgos Tsirantonakis, Elias Athanasopoulos, and Sotiris Ioannidis. Two-factor authentication: is the world ready? quantifying 2fa adoption. In *Proceedings of the eighth european workshop on system security*, pages 1–7, 2015. (Cited at page 97)

[88] TOTP: Time-Based One-Time Password Algorithm, Available online: https://datatracker.ietf.org/doc/html/rfc6238 (accessed on 12 May 2021). (Cited at page 97)

[89] K Aravindhan and RR Karthiga. One time password: A survey. *International Journal of Emerging Trends in Engineering and Development*, 1(3):613–623, 2013. (Cited at page 98)

[90] Imamah. One Time Password (OTP) Based on Advanced Encrypted Standard (AES) and Linear Congruential Generator(LCG). *2018 Electrical Power, Electronics, Communications, Controls and Informatics Seminar, EECCIS 2018*, pages 391–394, 2018. (Cited at page 98)

[91] Badis Hammi, Achraf Fayad, Rida Khatoun, Sherali Zeadally, and Youcef Begriche. A Lightweight ECC-Based Authentication Scheme for Internet of Things (IoT). *IEEE Systems Journal*, 14(3):3440–3450, 2020. (Cited at page 98)

[92] Samy Kambou and Ahmed Bouabdallah. A strong authentication method for web/-mobile services. In *2019 6th IEEE International Conference on Cyber Security and Cloud Computing (CSCloud)/2019 5th IEEE International Conference on Edge Computing and Scalable Cloud (EdgeCom)*, pages 124–129. IEEE, 2019. (Cited at page 99)

[93] Emir Erdem and Mehmet Tahir Sandikkaya. OTPaaS-One time password as a service. *IEEE Transactions on Information Forensics and Security*, 14(3):743–756, 2018. (Cited at page 99)

[94] Varun Amrutiya, Siddhant Jhamb, Pranjal Priyadarshi, and Ashutosh Bhatia. Trustless two-factor authentication using smart contracts in blockchains. *International Conference on Information Networking*, 2019-Janua(May):66–71, 2019. (Cited at page 99)

[95] Collin Mulliner, Ravishankar Borgaonkar, Patrick Stewin, and Jean-Pierre Seifert. Sms-based one-time passwords: attacks and defense. In *International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment*, pages 150–159. Springer, 2013. (Cited at page 99)

[96] Dimitrios Kallergis, Zacharenia Garofalaki, Georgios Katsikogiannis, and Christos Douligeris. Capodaz: A containerised authorisation and policy-driven architecture using microservices. *Ad Hoc Networks*, 104:102153, 2020. (Cited at page 99)

[97] Yustus Eko Oktian, Sang Gon Lee, and Hoon Jae Lee. TwoChain: Leveraging Blockchain and Smart Contract for Two Factor Authentication. In *2020 3rd International Seminar on Research of Information Technology and Intelligent Systems, ISRITI 2020*, pages 187–191. Institute of Electrical and Electronics Engineers Inc., dec 2020. (Cited at page 99)

[98] Francesco Buccafurri, Vincenzo De Angelis, and Roberto Nardone. Securing MQTT by blockchain-based otp authentication. *Sensors (Switzerland)*, 20(7), 2020. (Cited at page 99)

[99] Eman Alharbi . and Daniyal Alghazzawi . Two Factor Authentication Framework Using OTP-SMS Based on Blockchain. *Transactions on Machine Learning and Artificial Intelligence*, 7(3), 2019. (Cited at page 100)

[100] Mingli Zhang, Liming Wang, and Jing Yang. A Blockchain-Based Authentication Method with One-Time Password. In *2019 IEEE 38th International Performance Computing and Communications Conference, IPCCC 2019*. Institute of Electrical and Electronics Engineers Inc., oct 2019. (Cited at page 100)

[101] Nicola Dragoni, Saverio Giallorenzo, Alberto Lluch Lafuente, Manuel Mazzara, Fabrizio Montesi, Ruslan Mustafin, and Larisa Safina. Microservices: yesterday, today, and tomorrow. *Present and ulterior software engineering*, pages 195–216, 2017. (Cited at page 100)

[102] A. Celesti, M. Fazio, A. Galletta, L. Carnevale, J. Wan, and M. Villari. An approach for the secure management of hybrid cloud–edge environments. *Future Generation Computer Systems*, 90:1–19, 2019. (Cited at pages 104 e 185)

[103] Whitfield Diffie and Martin Hellman. New directions in cryptography. *IEEE transactions on Information Theory*, 22(6):644–654, 1976. (Cited at pages 111, 122, 124 e 135)

[104] Moxie Marlinspike and Trevor Perrin. The x3dh key agreement protocol. *Open Whisper Systems*, 2016. (Cited at pages 111, 114, 122 e 124)

[105] Trevor Perrin and Moxie Marlinspike. The double ratchet algorithm. *GitHub wiki*, 2016. (Cited at page 112)

[106] Katriel Cohn-Gordon, Cas Cremers, Benjamin Dowling, Luke Garratt, and Douglas Stebila. A formal security analysis of the signal messaging protocol. In *2017 IEEE European Symposium on Security and Privacy (EuroS&P)*, pages 451–466. IEEE, 2017. (Cited at page 112)

[107] WhatsApp's Signal Protocol integration, Available online: https://signal.org/blog/whatsapp-complete/ (accessed on 11 June 2020). (Cited at page 112)

[108] Facebook Messenger deploys Signal Protocol, Available online: https://signal.org/blog/facebook-messenger/ (accessed on 11 June 2020). (Cited at page 112)

[109] Deepak Puthal, Nisha Malik, Saraju P Mohanty, Elias Kougianos, and Chi Yang. The blockchain as a decentralized security framework [future directions]. *IEEE Consumer Electronics Magazine*, 7(2):18–21, 2018. (Cited at pages 112 e 136)

[110] Ankush Singla and Elisa Bertino. Blockchain-based pki solutions for iot. In *2018 IEEE 4th International Conference on Collaboration and Internet Computing (CIC)*, pages 9–15. IEEE, 2018. (Cited at page 112)

[111] Robert Muth and Florian Tschorsch. Smartdhx: Diffie-hellman key exchange with smart contracts. *IACR Cryptol. ePrint Arch.*, 2020:325, 2020. (Cited at page 112)

[112] Ethereum Light Sync Mode, Available online: https://docs.ethhub.io/using-ethereum/running-an-ethereum-node (accessed on 11 June 2020). (Cited at page 116)

[113] Alina Buzachis, Antonio Celesti, Antonino Galletta, Maria Fazio, Giancarlo Fortino, and Massimo Villari. A multi-agent autonomous intersection management (ma-aim) system for smart cities leveraging edge-of-things and blockchain. *Information Sciences*, 2020. (Cited at page 121)

[114] AN Prasad, K Al Mamun, FR Islam, and Haq Haqva. Smart water quality monitoring system. In *2015 2nd Asia-Pacific World Congress on Computer Science and Engineering (APWC on CSE)*, pages 1–6. IEEE, 2015. (Cited at pages 121 e 125)

[115] Farzad Samie, Vasileios Tsoutsouras, Lars Bauer, Sotirios Xydis, Dimitrios Soudris, and Jörg Henkel. Computation offloading and resource allocation for low-power iot edge devices. In *2016 IEEE 3rd World Forum on Internet of Things (WF-IoT)*, pages 7–12. IEEE, 2016. (Cited at page 121)

[116] Emanuel Tirado, Brendan Turpin, Cody Beltz, Phillip Roshon, Rylin Judge, and Kanwal Gagneja. A new distributed brute-force password cracking technique. In *International Conference on Future Network Systems and Security*, pages 117–127. Springer, 2018. (Cited at pages 122 e 135)

[117] Dahlia Malkhi. *Concurrency: The Works of Leslie Lamport*. ACM, 2019. (Cited at page 122)

[118] Shreyas Sen. Context-aware energy-efficient communication for iot sensor nodes. In *2016 53nd ACM/EDAC/IEEE Design Automation Conference (DAC)*, pages 1–6. IEEE, 2016. (Cited at pages 124 e 137)

[119] Aellison CT Santos, José L Soares Filho, Ávilla ÍS Silva, Vivek Nigam, and Iguatemi E Fonseca. Ble injection-free attack: a novel attack on bluetooth low energy devices. *Journal of Ambient Intelligence and Humanized Computing*, pages 1–11, 2019. (Cited at pages 124 e 137)

[120] M. S. Henriques and N. K. Vernekar. Using symmetric and asymmetric cryptography to secure communication between devices in iot. In *2017 International Conference on IoT and Application (ICIOT)*, pages 1–4, 2017. (Cited at pages 124 e 137)

[121] Urbi Chatterjee, Rajat Subhra Chakraborty, and Debdeep Mukhopadhyay. A puf-based secure communication protocol for iot. *ACM Trans. Embed. Comput. Syst.*, 16(3), April 2017. (Cited at pages 124 e 137)

[122] D. Fakhri and K. Mutijarsa. Secure iot communication using blockchain technology. In *2018 International Symposium on Electronics and Smart Devices (ISESD)*, pages 1–6, 2018. (Cited at page 124)

[123] M. Suárez-Albela, T. M. Fernández-Caramés, P. Fraga-Lamas, and L. Castedo. A practical performance comparison of ecc and rsa for resource-constrained iot devices. In *2018 Global Internet of Things Summit (GIoTS)*, pages 1–6, 2018. (Cited at pages 124 e 137)

[124] Monafin Afif Fiquaro, Raja Zahilah, Siti Hajar Othman, Marina Md. Arshad, and Sheikh Munir Sheikh Saad. Vaccination system using blockchain technology: A prototype development. In *2021 3rd International Cyber Resilience Conference (CRC)*, pages 1–6, 2021. (Cited at pages 124 e 137)

[125] A. Buzachis, A. Celesti, A. Galletta, M. Fazio, and M. Villari. A secure and dependable multi-agent autonomous intersection management (ma-aim) system leveraging blockchain facilities. In *2018 IEEE/ACM International Conference on Utility and Cloud Computing Companion (UCC Companion)*, pages 226–231, 2018. (Cited at pages 124 e 137)

[126] Geetanjali Rathee, Razi Iqbal, Omer Waqar, and Ali Kashif Bashir. On the design and implementation of a blockchain enabled e-voting application within iot-oriented smart cities. *IEEE Access*, 9:34165–34176, 2021. (Cited at page 124)

[127] Elliptic Curve Library, Available online: https://github.com/witnet/elliptic-curve-solidity (accessed on 11 June 2020). (Cited at page 129)

[128] Jelena Mirkovic and Peter Reiher. A taxonomy of ddos attack and ddos defense mechanisms. *ACM SIGCOMM Computer Communication Review*, 34(2):39–53, 2004. (Cited at page 135)

[129] Armando Ruggeri, Antonino Galletta, Antonio Celesti, Maria Fazio, and Massimo Villari. An innovative blockchain based application of the extended triple diffie-hellman protocol for iot. In *2021 8th International Conference on Future Internet of Things and Cloud (FiCloud)*, pages 278–284, 2021. (Cited at page 135)

[130] Jayasree Sengupta, Sushmita Ruj, and Sipra Das Bit. A comprehensive survey on attacks, security issues and blockchain solutions for iot and iiot. *Journal of Network and Computer Applications*, 149:102481, 2020. (Cited at page 136)

[131] Minhaj Ahmad Khan and Khaled Salah. Iot security: Review, blockchain solutions, and open challenges. *Future generation computer systems*, 82:395–411, 2018. (Cited at page 136)

[132] Savita Mohurle and Manisha Patil. A brief study of wannacry threat: Ransomware attack 2017. *International Journal of Advanced Research in Computer Science*, 8(5):1938–1940, 2017. (Cited at page 137)

[133] Saurabh Singh, Pradip Kumar Sharma, Byungun Yoon, Mohammad Shojafar, Gi Hwan Cho, and In-Ho Ra. Convergence of blockchain and artificial intelligence in iot network for the sustainable smart city. *Sustainable Cities and Society*, 63:102364, 2020. (Cited at page 137)

[134] Docker, Available online: `https://www.docker.com` (accessed on 22 July 2021). (Cited at page 140)

[135] Kristen Griggs, Olya Ossipova, Christopher Kohlios, Alessandro Baccarini, Emily Howson, and Thaier Hayajneh. Healthcare blockchain system using smart contracts for secure automated remote patient monitoring. *Journal of Medical Systems*, 42, 07 2018. (Cited at pages 147 e 158)

[136] John James. A new, evidence-based estimate of patient harms associated with hospital care. *Journal of patient safety*, 9, 07 2013. (Cited at page 148)

[137] International Classification of Diseases,Ninth Revision, Clinical Modification (ICD-9-CM). `https://www.cdc.gov/nchs/icd/icd9cm.htm`. Last access: December 2019. (Cited at page 148)

[138] M. Burkhardt. Icd10 : International classification of diseases, who, 10 2019. (Cited at page 148)

[139] A Skrbo, I Zulić, S Hadzić, and I Gaon. (anatomic-therapeutic-chemical classification of drugs). *Medicinski arhiv*, 53:57–60, 02 1999. (Cited at page 148)

[140] Saurabh Gupta, Emilie Belley-Côté, Bram Rochwerg, Anthony Bozzo, Puru Panchal, Arjun Pandey, Lawrence Mbuagbaw, Shamir Mehta, J-D Schwalm, and Richard Whitlock. Antiplatelet therapy and coronary artery bypass grafting: Protocol for a systematic review and network meta-analysis. *Medicine*, 98:e16880, 08 2019. (Cited at pages 149 e 152)

[141] Shinichiro Uchiyama, Shinya Goto, Hideki Origasa, Naomi Uemura, Kentaro Sugano, Hideyuki Hiraishi, Kazuyuki Shimada, Yasushi Okada, and Yasuo Ikeda. Major cardiovascular and bleeding events with long-term use of aspirin in patients with prior cardiovascular diseases: 1-year follow-up results from the management of aspirin-induced gastrointestinal complications (magic) study. *Heart and Vessels*, 08 2019. (Cited at pages 149 e 152)

[142] Jessica Hwang and Roy Weiss. Steroid induced diabetes: A clinical and molecular approach to understanding and treatment. *Diabetes/metabolism research and reviews*, 30, 02 2014. (Cited at pages 149 e 152)

[143] Rajiv Kohli and Frank Piontek. *DSS in Healthcare: Advances and Opportunities*, pages 483–497. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008. (Cited at page 149)

[144] B. Sekar, J. Lamy, N. Muro, A. U. Pinedo, B. Seroussi, N. Larburu, G. Guézennec, J. Bouaud, F. G. Masero, M. Arrúe, and H. Wang. Intelligent clinical decision support systems for patient-centered healthcare in breast cancer oncology. In *2018 IEEE 20th International Conference on e-Health Networking, Applications and Services (Healthcom)*, pages 1–6, Sep. 2018. (Cited at page 149)

[145] S. Billings and H. Wei. Narmax model as a sparse, interpretable and transparent machine learning approach for big medical and healthcare data analysis. In *2019 IEEE 21st International Conference on High Performance Computing and Communications; IEEE 17th International Conference on Smart City; IEEE 5th International Conference on Data Science and Systems (HPCC/SmartCity/DSS)*, pages 2743–2750, Aug 2019. (Cited at page 149)

[146] L. T. Gaudio, P. Veltri, and G. Fragomeni. Modeling and application of aorta coarctation: support system for pre-operative decision. In *2018 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, pages 2031–2032, Dec 2018. (Cited at page 149)

[147] F. Ben Chihaoui, N. Maddeh, S. B. Layeb, C. Hamouda, and J. Chaouachi. A decision support system for drug inventory management within an emergency department: A case study. In *2019 6th International Conference on Control, Decision and Information Technologies (CoDIT)*, pages 1889–1894, April 2019. (Cited at page 149)

[148] A. Celesti, M. Fazio, A. Romano, A. Bramanti, P. Bramanti, and M. Villari. An oais-based hospital information system on the cloud: Analysis of a nosql column-oriented approach. *IEEE Journal of Biomedical and Health Informatics*, 22(3):912–918, May 2018. (Cited at page 150)

[149] Antonio Celesti, Armando Ruggeri, Maria Fazio, Antonino Galletta, Massimo Villari, and Agata Romano. Blockchain-based healthcare workflow for tele-medical laboratory in federated hospital iot clouds. *Sensors*, 20(9):2590, 2020. (Cited at page 151)

[150] R. Ranjan, O. Rana, S. Nepal, M. Yousif, P. James, Z. Wen, S. Barr, P. Watson, P.P. Jayaraman, D. Georgakopoulos, M. Villari, M. Fazio, S. Garg, R. Buyya, L. Wang, A.Y. Zomaya, and S. Dustdar. The next grand challenges: Integrating the internet of things and data science. *IEEE Cloud Computing*, 5(3):12–26, 2018. (Cited at page 152)

[151] Patrick Hassenteufel, François-Xavier Schweyer, Thomas Gerlinger, Rüdiger Henkel, Caspar Lückenbach, and Renate Reiter. The role of professional groups in policy change: Physician's organizations and the issue of local medical provision shortages in france and germany. *European Policy Analysis*, 2019. (Cited at pages 156 e 170)

[152] Katarzyna Dubas-Jakóbczyk, Alicja Domagała, and Marcin Mikos. Impact of the doctor deficit on hospital management in poland: A mixed-method study. *The International Journal of Health Planning and Management*, 34(1):187–195, 2019. (Cited at pages 156 e 170)

[153] R. Moreno-Vozmediano, E. Huedo, I.M. Llorente, R.S. Montero, P. Massonet, M. Villari, G. Merlino, A. Celesti, A. Levin, L. Schour, C. Vázquez, J. Melis, S. Spahr, and D. Whigham. Beacon: A cloud network federation framework. *Communications in Computer and Information Science*, 567:325–337, 2016. (Cited at page 156)

[154] Antonio Celesti, Maria Fazio, Maurizio Giacobbe, Antonio Puliafito, and Massimo Villari. Characterizing cloud federation in iot. In *2016 30th International Conference on Advanced Information Networking and Applications Workshops (WAINA)*, pages 93–98. IEEE, 2016. (Cited at page 157)

[155] D. Mulfari, A. Celesti, M. Villari, and A. Puliafito. How cloud computing can support on-demand assistive services. In *W4A 2013 - International Cross-Disciplinary Conference on Web Accessibility*, 2013. (Cited at page 157)

[156] J.-W. Lian, D.C. Yen, and Y.-T. Wang. An exploratory study to understand the critical factors affecting the decision to adopt cloud computing in taiwan hospital. *International Journal of Information Management*, 34(1):28–36, 2014. (Cited at page 157)

[157] Haider Dhia Zubaydi, Yung-Wey Chong, Kwangman Ko, Sabri M. Hanshi, and Shankar Karuppayah. A review on the role of blockchain technology in the healthcare domain. *Electronics*, 8(6), 2019. (Cited at page 158)

[158] Cornelius C. Agbo, Qusay H. Mahmoud, and J. Mikael Eklund. Blockchain technology in healthcare: A systematic review. *Healthcare*, 7(2), 2019. (Cited at page 158)

[159] Marko Hölbl, Marko Kompara, Aida Kamišalić, and Lili Nemec Zlatolas. A systematic review of the use of blockchain in healthcare. *Symmetry*, 10(10), 2018. (Cited at page 158)

[160] Y. A. Qadri, A. Nauman, Y. B. Zikria, A. V. Vasilakos, and S. W. Kim. The future of healthcare internet of things: A survey of emerging technologies. *IEEE Communications Surveys Tutorials*, pages 1–53, 2020. (Cited at page 158)

[161] S. Chakraborty, S. Aich, and H. Kim. A secure healthcare system design framework using blockchain technology. In *2019 21st International Conference on Advanced Communication Technology (ICACT)*, pages 260–264, Feb 2019. (Cited at pages 158 e 171)

[162] T. K. Dasaklis, F. Casino, and C. Patsakis. Blockchain meets smart health: Towards next generation healthcare services. In *2018 9th International Conference on Information, Intelligence, Systems and Applications (IISA)*, pages 1–8, July 2018. (Cited at pages 158 e 171)

[163] G. Srivastava, J. Crichigno, and S. Dhar. A light and secure healthcare blockchain for iot medical devices. In *2019 IEEE Canadian Conference of Electrical and Computer Engineering (CCECE)*, pages 1–5, May 2019. (Cited at pages 158 e 171)

[164] K. M. Hossein, M. E. Esmaeili, T. Dargahi, and A. khonsari. Blockchain-based privacy-preserving healthcare architecture. In *2019 IEEE Canadian Conference of Electrical and Computer Engineering (CCECE)*, pages 1–4, May 2019. (Cited at pages 158 e 171)

[165] Jonathan Biggs, Sheri R Hinish, Michael A Natale, and Matt Patronick. Blockchain: Revolutionizing the global supply chain by building trust and transparency. *New Jersey: Rutgers University*, 2017. (Cited at page 158)

[166] Bingqing Shen, Jingzhi Guo, and Yilong Yang. Medchain: Efficient healthcare data sharing via blockchain. *Applied Sciences*, 9(6), 2019. (Cited at page 158)

[167] Asma Khatoon. A blockchain-based smart contract system for healthcare management. *Electronics*, 9(1), 2020. (Cited at page 158)

[168] Krishna Prasad Satamraju and Malarkodi B. Proof of concept of scalable integration of internet of things and blockchain in healthcare. *Sensors*, 20(5), 2020. (Cited at page 158)

[169] Telemedcare Clinical Monitoring Unit (CMU), Available online: https://www.telemedcare.com/ (accessed on 20 March 2020). (Cited at page 159)

[170] Enverse Continous Glucose Monitoring, Available online: https://www.eversensediabetes.com/ (accessed on 20 March 2020). (Cited at page 159)

[171] Med-care, Available online: http://www.t4all.it/portfolio-articoli/med-care/ (accessed on 20 March 2020). (Cited at page 159)

[172] HomoScreen, Available online: https://www.pixcell-medical.com/products/hemoscreen (accessed on 20 March 2020). (Cited at page 159)

[173] Samsung Labgeo PT10S, Available online: https://samsunghealthcare.com/en/products/InVitroDi of-Care/benefit (accessed on 20 March 2020). (Cited at page 159)

[174] Antonino Galletta, Lilla Bonanno, Antonio Celesti, Silvia Marino, Placido Bramanti, and Massimo Villari. An approach to share mri data over the cloud preserving patients' privacy. In *2017 IEEE Symposium on Computers and Communications (ISCC)*, pages 94–99. IEEE, 2017. (Cited at page 165)

[175] Antonino Galletta, Antonio Celesti, Francesco Tusa, Maria Fazio, Placido Bramanti, and Massimo Villari. Big mri data dissemination and retrieval in a multi-cloud hospital

storage system. In *Proceedings of the 2017 International Conference on Digital Health*, pages 162–166. ACM, 2017. (Cited at page 165)

[176] Ashish K. Jha, Timothy G. Ferris, Karen Donelan, Catherine DesRoches, Alexandra Shields, Sara Rosenbaum, and David Blumenthal. How common are electronic health records in the united states? a summary of the evidence. *Health Affairs*, 25(Supplement 1):W496–W507, 2006. PMID: 17035341. (Cited at page 170)

[177] Vidhya Ramani, Tanesh Kumar, An Bracken, Madhusanka Liyanage, and Mika Yliant-tila. Secure and efficient data accessibility in blockchain based healthcare systems. *2018 IEEE Global Communications Conference (GLOBECOM)*, pages 206–212, 2018. (Cited at page 171)

[178] Tien Tuan Anh Dinh, Ji Wang, Gang Chen, Rui Liu, Beng Chin Ooi, and Kian-Lee Tan. Blockbench: A framework for analyzing private blockchains. In *Proceedings of the 2017 ACM International Conference on Management of Data*, page 1085–1100. Association for Computing Machinery, 2017. (Cited at page 174)

[179] Dario Amodei, Chris Olah, Jacob Steinhardt, Paul Christiano, John Schulman, and Dan Mané. Concrete problems in ai safety. *arXiv preprint arXiv:1606.06565*, 2016. (Cited at page 178)

[180] R. Di Salvo, M. Fazio, A. Celesti, D. Santoro, and M. Villari. Mathematical model and ai oriented analysis for self-regulated learning in remote health treatments. In *Proceedings of IEEE GLOBECOM 2020 Workshop on AI-driven Smart Healthcare*, December 2020. (Cited at pages 178 e 180)

[181] Dawid Połap, Gautam Srivastava, Alireza Jolfaei, and Reza M. Parizi. Blockchain technology and neural networks for the internet of medical things. In *IEEE INFOCOM 2020 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pages 508–513, 2020. (Cited at page 179)

[182] A. Ruggeri, M. Fazio, A. Galletta, A. Celesti, and M. Villari. A decision support system for therapy prescription in a hospital centre. In *Proceedings of IEEE Symposium on Computers and Communications*, 2020. (Cited at page 179)

[183] Adrian Groza and Ana-Diana Pop. Fake news detector in the medical domain by reasoning with description logics. In *2020 IEEE 16th International Conference on Intelligent Computer Communication and Processing (ICCP)*, pages 145–152, 2020. (Cited at page 179)

[184] Dawid Połap, Gautam Srivastava, Alireza Jolfaei, and Reza M Parizi. Blockchain technology and neural networks for the internet of medical things. In *IEEE INFOCOM 2020-IEEE conference on computer communications workshops (INFOCOM WKSHPS)*, pages 508–513. IEEE, 2020. (Cited at page 179)

[185] Gábor Kiss. External manipulation of autonomous vehicles. In *2019 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computing, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI)*, pages 248–252. IEEE, 2019. (Cited at page 179)

[186] Aymeric Cretin, Alexandre Vernotte, Antoine Chevrot, Fabien Peureux, and Bruno Legeard. Test data generation for false data injection attack testing in air traffic surveillance. In *2020 IEEE International Conference on Software Testing, Verification and Validation Workshops (ICSTW)*, pages 143–152. IEEE, 2020. (Cited at page 179)

[187] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, second edition, 2018. (Cited at page 180)

[188] Italian PON project TALIs-MAn, Available online: https://www.progettotalisman.it/ (accessed on 11 May 2021). (Cited at page 180)