
Secure interaction with IIoT nodes using new technologies



DOCTORAL THESIS

Ernesto Gómez Marín
Doctoral Programme in Information and Communication
Technologies
Departamento de Electrónica y Tecnología de Computadores.
Universidad de Granada.
November 2023

Este documento está preparado para ser impreso a doble cara.

Secure interaction with IIoT nodes using new technologies

Directed by:

Prof. Encarnación Castillo Morales

Prof. Luis Parrilla Roure

Developed at:

Infineon Technologies AG

BEX RDE RDF EET

Munich, Germany

**Doctoral Programme in Information and Communication
Technologies**
Departamento de Electrónica y Tecnología de Computadores.
Universidad de Granada.

November 2023

Editor: Universidad de Granada. Tesis Doctorales
Autor: Ernesto Gómez Marín
ISBN: 978-84-1195-176-0
URI: <https://hdl.handle.net/10481/89444>

El doctorando / The *doctoral candidate* [**Ernesto Gómez Marín**] y los directores de la tesis / and the thesis supervisor/s: [**Encarnación Castillo Morales y Luis Parrilla Roure**]

Garantizamos, al firmar esta tesis doctoral, que el trabajo ha sido realizado por el doctorando bajo la dirección de los directores de la tesis y hasta donde nuestro conocimiento alcanza, en la realización del trabajo, se han respetado los derechos de otros autores a ser citados, cuando se han utilizado sus resultados o publicaciones.

/

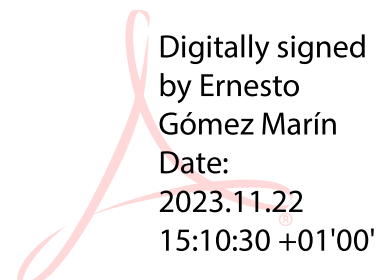
Guarantee, by signing this doctoral thesis, that the work has been done by the doctoral candidate under the direction of the thesis supervisor/s and, as far as our knowledge reaches, in the performance of the work, the rights of other authors to be cited (when their results or publications have been used) have been respected.

Lugar y fecha / Place and date:

Granada, 21/11/2023

Director/es de la Tesis / *Thesis supervisor/s*;

Doctorando / *Doctoral candidate*:



Digitally signed
by Ernesto
Gómez Marín
Date:
2023.11.22
15:10:30 +01'00'

Firma / Signed

Firma / Signed

Acknowledgements

*Our greatest glory is not in never failing,
but in rising every time we fail.*

Confucius

I really want to thank to a lot of people that helped me in these more than four years of PhD. Firstly, to Diego Pedro, who was the person who put his trust in me recommending me on this path. I still remember the nice 20 minutes conversation we had before starting the position. Thanks to him I had Encarni as my tutor, a great professional and above all a great person who has supported me closely in the development of my thesis. And also Luis Parrilla whose experiences have guided and corrected me during these years.

When I arrived at Infineon, I was surprised by the friendliness with which I was welcomed in the IFAG BEX RDE RDF department. My supervisor Antonio Escobar became a great friend with whom I shared beers and barbecues in the evenings. He taught me and guided me through my first years in the professional world. Also, I can't thank Antonio Cabrera, a great computer expert who helped me implement my ideas.

In Infineon, my research was founded by the European Union's Horizon 2020 Research, under grant agreement No. 871518, a project named COLLABS. Thanks to this project I had the opportunity and freedom to develop my ideas to solve the challenges presented and to meet fantastic professionals like Davide, from whom I learned how to make a friendship with coworkers from different companies. There, I also met Valerio, a fantastic project manager who taught me the meaning of being a project manager. The truth is that I could go on thanking people who have been with me and this list would never end, but I can't avoid naming Gianco, Miguel, Nuria, Juan Potencia, MC², Hendrik, Jessica and Evgenia. They have made these four years not only a PhD in Munich but one of the best times of my life.

Finally but not last, I want to thank my girlfriend Maria, who supported me in all the steps of the Thesis.

Abstract

*I think anything is possible if you have
the mindset and the will and desire to do
it and put the time in.*

Roger Clemens

Internet of Things (IoT) is the name given to the ecosystem formed by multiple devices (things) that communicate with each other to achieve one or more common goals. Since its emergence more than 20 years ago, many practical applications have improved people's quality of life. Also, the efficiency and quality of production in the industry have increased with the inclusion of the Industrial-IoT (IIoT). However, IoT environments are usually very insecure due to: a lack of trust in normally vulnerable IoT devices, a huge number of devices with a lack of supervision, the complexity of identifying IoT devices, the easy manipulation of IoT data, the huge volume of data generated and the high number of data readers with different read rights. This insecurity affects this technology's impact due to the restriction of the scenarios in which it can be applied. For example, IIoT nodes are typically deployed in insulated environments, avoiding interaction between devices in different factories or reducing the significance of actions triggered based on the IIoT data. The present research seeks to turn this circumstance around. To overcome this complex challenge, secure hardware has been used to protect IIoT devices and their identities, different cryptographic algorithms were combined to protect data confidentiality, and finally, blockchain was deployed to obtain data integrity even in scenarios involving multiple organizations. Thanks to the combination of these technologies, new data architectures and self-designed Public Key Infrastructure have been created, easing the IIoT deployment in diverse scenarios, involving organizations with different interests, achieving fine-grained authorization for data access and extremely high reliability of the data generated. The results of this research serve to increase the value of all IoT data by increasing the confidence and accessibility of this data. More importantly, they serve to pave the way for countless new applications of IoT that will ultimately affect an increase in the quality of life of the people.

Resumen

Internet de las cosas (Internet of Things, IoT) es el nombre que recibe el ecosistema formado por múltiples dispositivos (cosas) que se comunican entre sí para alcanzar uno o varios objetivos comunes. Desde su aparición hace más de 20 años, han sido muchas las aplicaciones prácticas que han mejorado la calidad de vida de las personas. De la misma manera la eficiencia y calidad de la industria ha mejorado gracias a la inclusión de los IoT Industriales (IIoT, por sus siglas en inglés). Sin embargo, los entornos IoT son normalmente muy inseguros debido a: la falta de confianza en dispositivos IoT normalmente vulnerables, el enorme número de dispositivos que carecen de supervisión, la complejidad de identificar dispositivos IoT, la fácil manipulación de datos IoT, el enorme volumen de datos generados y el elevado número de lectores de datos con diferentes derechos de lectura. Esta inseguridad afecta al impacto de esta tecnología debido a la restricción de los escenarios en los que se puede aplicar. Por ejemplo, los nodos IIoT se despliegan normalmente en entornos aislados evitando cualquier interacción entre Fabricas y reduciendo la importancia de las acciones desencadenadas en base a los datos IIoT. La presente investigación pretende dar respuesta a esta circunstancia. Para superar este complejo reto, se ha utilizado hardware seguro para proteger los dispositivos IIoT y sus identidades, se han propuesto múltiples algoritmos criptográficos combinados para proteger la confidencialidad de los datos y, por último, se ha desplegado *blockchain* para obtener confidencialidad incluso en escenarios en los que intervienen múltiples organizaciones. Gracias a la combinación de estas tecnologías, se han creado nuevas arquitecturas de datos e Infraestructuras de Clave Pública de diseño propio, permitiendo que los dispositivos puedan ser desplegados en cualquier ecosistema, involucrando organizaciones con diferentes intereses, consiguiendo una autorización de grano fino para el acceso a los datos y una altísima fiabilidad de los datos generados. Los resultados de esta investigación servirán para incrementar el valor de todo los datos IoT, aumentando la confianza y la accesibilidad de estos datos. Y lo que es más importante, sirven para allanar el camino a innumerables nuevas aplicaciones de IoT que, en última instancia, repercutirán en un aumento de la calidad de vida de las personas.

Contents

Acknowledgements	VII
Abstract	IX
Resumen	XI
I PhD Dissertation	1
1. Introduction	3
1.1. Motivation	4
1.2. Objectives	5
1.3. Outline	6
2. Methodology	9
2.1. Study of the scenarios	9
2.2. Related technologies	11
2.2.1. Cryptography	11
2.2.2. Hardware Security Modules	11
2.2.3. Blockchain	12
2.3. Methodology for the achievement of the objectives	13
2.3.1. Development of the Schemes	13
2.3.2. Security analysis	13
2.3.3. Prototype development	13
3. Achievements	17
3.1. Research on IoT security in supply-chain	17
3.2. Research on Fine-Grained Authorization in IoT	20
3.3. Research on PKI in IoT and Sensors	23
3.4. Research on remote attestation in IoT	25
References	27

II Publications	33
4. An Innovative Strategy Based on Secure Element for Cyber–Physical Authentication in Safety-Critical Manufacturing Supply Chain	35
4.1. Introduction	36
4.2. Motivation and Scenario	38
4.3. Related Work	40
4.4. Background	42
4.5. Proposed Solution	44
4.5.1. Integration of SE in the SCM	44
4.5.2. Change Ownership Detailed	46
4.6. Implementation	49
4.7. Security Analysis	53
4.7.1. Threat Analysis	53
4.7.2. Attack Analysis	54
4.7.3. Comparison	56
4.8. Conclusions and Future Work	56
References	57
5. Fine-Grained Access Control with User Revocation in Smart Manufacturing	63
5.1. Introduction	64
5.2. Use Case and Requirements	66
5.3. Related Work	69
5.4. Background	73
5.4.1. CP-ABE Functions	73
5.4.2. Blockchain	73
5.4.3. Hardware Security Modules	74
5.5. Proposed Scheme	74
5.6. Details of the Proposed Scheme	77
5.6.1. Attributes	77
5.6.2. Revocation	78
5.6.3. Smart Contracts	81
5.6.4. Phases	81
5.7. Evaluation	83
5.7.1. Performance Evaluation	83
5.7.2. Security Evaluation	84
5.7.3. Requirements Evaluation	87
5.8. Discussion and Future Work	88
5.9. Conclusions	89

References	91
6. Towards sensor measurement reliability in Blockchain	99
6.1. Introduction	100
6.2. Related work	102
6.2.1. Data Security Requeriments	102
6.2.2. PKI in Blockchain.	103
6.2.3. IoT in Blockchain	104
6.2.4. Oracles	105
6.3. Background	107
6.3.1. Secure sensor	107
6.3.2. Ethereum Addresses	108
6.3.3. Blockchain and smart contracts	109
6.3.4. Assumptions	111
6.4. Use case: ensuring respect of the cold chain through smart contracts	111
6.5. Design of the proposed system	112
6.5.1. Proposed system	112
6.5.2. PKI	114
6.5.3. Freshness	116
6.5.4. Inserting reliable information of previous blocks to a smartcontract	118
6.5.5. Detailed process	119
6.6. System Implementation	123
6.7. Security Analysis	125
6.8. Conclusion	126
References	126
7. RESEKRA: Remote Enrollment Using SEaled Keys for Re- mote Attestation	133
7.1. Introduction	134
7.2. Related Work	137
7.3. Background	138
7.3.1. TPM 2.0	138
7.3.2. Integrity Measurement Architecture (IMA)	141
7.3.3. Core Root of Trust of Measurement	141
7.4. RESEKRA Description	142
7.4.1. Roles	142
7.4.2. Certification of Manufacturer	143
7.4.3. Remote Enrollment	145
7.4.4. Attestation	147

7.4.5. Daily Life Functionality	148
7.4.6. Implementation	149
7.5. RESEKRA Use Cases	150
7.5.1. Edge Computing as Trusted Service	150
7.5.2. Edge Computing as Service—No TTP Online	152
7.6. Security Analysis	154
7.6.1. Man-In-the-Middle Attack (MIM attack)	154
7.6.2. Impersonation and Replay Attack	155
7.6.3. Wormhole Attack	155
7.6.4. Interference Attacks	156
7.6.5. Time-Of-Check-To-Time-Of-Use Attack	156
7.6.6. Verifier-Based DoS Attack	156
7.6.7. Non-Verification DoS Attack	157
7.7. Conclusions	157
References	159
III Conclusions	165
8. Conclusions	167
8.1. Future trends	169
References	171

List of Figures

2.1. Flow diagram of the used methodology.	15
3.1. Two-step transaction infrastructure used to change the ownership of a package in the smart contract in the blockchain. [49]	19
3.2. High-level scheme of factory scenario [40].	21
3.3. High-level scheme of a SS installed in a package of a supply chain sending data to the Blockchain [37].	24
3.4. Experimental setup. Raspberry Pi 4 with TPM (green hardware) and pressure sensor (red hardware) [42].	26
4.1. Overview of the proposed interactions.	45
4.2. Two-step transaction infrastructure used to change the ownership of a package in the smart contract in the blockchain. The numbers in the figure refer to the enumerated steps in the text.	47
4.3. Secure architecture of a modern blockchain-based SCM applied to manufacturing use case.	50
4.5. Prototype setup for laboratory evaluation.	52
4.4. New lines used to identify the parts using their public key instead of their IDs.	52
5.1. High-level scheme of factory scenario.	68
5.2. High-level scheme of the proposed solution.	75
6.1. High level schematic of Secure Sensor.	108
6.2. Abstract class "Smart contract" following the Unified Modeling Language.	110
6.3. High-level scheme of a SS installed in a package of a supply chain sending data to the Blockchain.	113
6.4. Class diagram of the complete infrastructure following the UML.	115
6.5. Graphic representation of the variables in (6.2).	117

6.6.	Sequence diagram of the complete infrastructure following the UML.	121
6.7.	Graphical representation of the "Dangerous Zone", "Margin Zone" and "Secure Zone" of the temperature of a product in a cold chain.	124
7.1.	Remote enrollment process. The scheme represents all the enrollment process from the device manufacturer to the supervised creation of the sealed key.	144
7.2.	Output of the script for sealed key creation.	146
7.3.	Section of the Verifier code where it computes the policies for using the SeK and signs it if and only if the Attestor passed through the attestation process.	148
7.4.	Output of the script using the sealed key to sign a measured value after satisfying the policies.	149
7.5.	Experimental setup. Raspberry Pi 4 with TPM (green hardware) and pressure sensor (red hardware).	149
7.6.	Visualization of edge computing as trusted service. A car reaching a location needs the local services of several edge and IoT devices. All these devices have many SeKs available validated by an SC each. The car just trusts in some of the SCs, those which belong to subset G.	153
7.7.	Visualization of the two use cases, edge computing as service (in blue) and edge computing as service—No TTP online (in purple).	154

List of Tables

4.1. High-level threats related to the cyber–physical link.	39
4.2. Data assets and protection policies.	51
4.3. HSM performances while interrogated through NodeJS API.	53
4.4. Threat analysis.	54
4.5. Comparison of decentralized-based SCMs.	57
5.1. Evaluation of the requirements.	87
5.2. Comparison of blockchain-based ABE protocols.	88
6.1. Measurements of non-optimized SS	124
6.2. Comparison of Oracle protocols	126

Part I

PhD Dissertation

Chapter 1

Introduction

*Men in general are quick to believe that
which they wish to be true.*

Julio Caesar

Humankind soon stood out from the rest of the species for its ability to reason, and to make the right decisions based on events, even if these decisions were not straightforward. With accurate information and good reasoning skills, humans can excel as generals, leaders, architects, traders, or scientists. In the pursuit of excellence, mankind has sought to perfect reasoning by defining algorithms and theories that allow us to make the best decisions based on the information we have. With the invention of computers, data is now processed with more complex and heavy algorithms to make better decisions, and in some cases, it is even the machines themselves that make the final decisions. It does not take much imagination to see that in the future, the perfection of reasoning will reach a new level with the use of artificial intelligence, and with the use of blockchain, many decisions and responsibilities will be given to machines. But we must not forget the important role of information for correct reasoning; any conclusion is wrong if the information is inaccurate. Nowadays, we only rely on information that we have obtained ourselves or that has been given to us over the Internet by someone we trust. But that dramatically limits our sources of information because now most information on the Internet is not given to us by someone, but by someThing [1]. The Internet of Things (IoT) offers a huge quantity of data from the real world in real-time, but why do we want so much information if we cannot trust it?, in essence, if we cannot trust the IoT things?

Definitions of things in the IoT vary from research to research. Following the definition Evangelos A. Kosmatos *et al.* [2], supported by S. Madakam *et al.* [1], or the definition of [3], thing is any physical object - living or not-living - with a unique identity that is integrated into a computing network. The most basic "things" are tagged objects with passive RFID tags [4],

e.i., an apple with a smart label. With an RFID reader, it is possible to scan its unique identifier (a bit string) and use it to look in the network for the linked information of this thing, e.i., kind of apple, kind of farming procedure, or expiration date. Other more complex things can be powered devices with a battery and energy harvesting that, once installed, measures temperature and send it to a defined IP through NB-IoT [5], indicating its identity. Mobile-phones, smart cars, tablets, server are also considered things.

These things cooperating together can create “an open and comprehensive network of intelligent objects that have the capacity to auto-organize, share information, data and resources, reacting and acting in face of situations and changes in the environment” [1]. It can be applied in a huge number of scenarios: factories where robots communicate together collaborating in order to become an only entity, monitoring water and air pollution for human and agriculture safety [6], automatizing agriculture with drones and autonomous trucks increasing productivity and reducing prices [7][8], smart house automation [9], smart education, smart cities, patients health tracker, etc. However, all these IoT applications remain largely untapped, as they are often confined to isolated environments, such as IIoT devices that cannot communicate between smart factories to improve the efficiency of collaborative manufacturing. The same occurs to thousands of applications in the State of the Art (SoA) stuck in the realm of theoretical studies, far from the implementation of practical application. All these problems occur because of the insecurity and distrust that so much automation brings to people. And this fear is not unfounded because, indeed, IoT has a big problem: IoT devices and their data are not cybersecure.

1.1. Motivation

The number of IoT devices connected to internet is growing rapidly, from 6.1 billion in 2018 to 14.1 billions in 2023 [10]. However, the security limitations of IoT devices are well known: the majority of IoT devices have vulnerabilities that can be remotely exploited [11, 12]. This explains that 83% of these devices used or manufactured by large companies suffered a cyber attack in 2018 and 79% of the manufacturing companies experienced a cyberattack in their IIoT the same year [13].

However, the risks are not only in the devices themselves but also IoT system architecture and data management. For instance, the data flow represents a critical concern as it can be vulnerable to tampering, enabling attackers to insert malicious exploits into the system or manipulate data, leading to erroneous outcomes and decisions [14]. Additionally, inadequate data flow management can lead to critical confidentiality breaches [14].

In addition, the security of data at rest is a major concern in the context of IoT devices, in addition to the threats associated with data transmission.

Determining a reliable storage mechanism becomes essential since these devices produce enormous volumes of diverse data. Despite being widely utilized, centralized cloud systems like Amazon Web systems, Microsoft Azure, Alibaba Cloud, and Google Cloud have dependability concerns since they are susceptible to errors and data loss, as shown in [15, 16, 17, 18]. On the other hand, local storage is not a practical option because of the dispersion of IoT devices and the sheer volume of data collected, and fog store raises trust difficulties since it could alter data.

Another critical problem with IoT data is its confidentiality, a significant issue with IoT data at rest. Traditional methods encrypt the data using conventional encryption and distribute decryption keys to authorized parties. This approach creates issues, though, because a party with access to one set of data may abuse that access to acquire unlawful access to other data kept in the same repository. Such illegal access is a serious concern that needs careful consideration given the diversity of IoT data.

One of the current solutions to this issue is to cluster IoT devices and their data and isolate them using local networks and firewalls. However, the effectiveness and possibilities of this tactic are extremely limited. In industrial contexts, for instance, internal bureaucratic restrictions may make it challenging for different organizational levels to share IIoT-generated data. Supply chain ecosystems have similar limitations on data sharing with outside partners, which makes it difficult to successfully upgrade and enhance supply chain networks. Moreover, even in the context of smart cities and their IoT devices, the adoption of stringent isolation measures often results in compromising the full potential of these devices. Many functionalities must be curtailed to mitigate the heightened risks limiting the potential benefits they could offer.

To overcome these challenges, innovative and holistic approaches are required, which strike a balance between security and functionality, enabling seamless data sharing, and fostering collaborations across different sectors without compromising on cybersecurity.

1.2. Objectives

This Doctoral thesis aims to achieve the complete reliability of the data generated and processed by IoT devices with all its security guarantees. To do so, the research designs, implements and analyzes secure IoT ecosystems in different scenarios using all available and applicable technologies in each of them. The objective can be split on:

- Finding scenarios where IoT security is a bottleneck for the development.
- Proposing viable solutions to improve the security of the IoT ecosystem in the relevant scenarios.

- Showing the feasibility of the proposed solutions through a technical implementation.
- Ensuring the proposed solutions fulfill the security requirements.

In the following, we describe those that we consider the most important and necessary security requirements for these achievements. All of these requirements were gathered through the meticulous application of the STRIDER methodology [19] and the considerations of Lui *et al.* [20].

- Confidentiality: It is necessary to implement measures in order to avoid an external entity eavesdropping the communication.
- Fine-grained Authorization: The data must also be protected against insider snooper; authorized users shall not abuse their privileges to access non-relevant data for them.
- Integrity: Mechanisms shall be implemented so that the data's final consumer can recognize changes made to those compared to the data produced by the devices.
- Authenticity: The identity of the generator of the data must be checked and probed before accepting any data.
- Trustworthiness: Even if the identity of the data source is confirmed, it is needed to verify that the device was not manipulated or attacked, e.i., it was in a trusted status when sending the data.
- Traceability: It is important that all data stored can be tracked back to identify the origin of fake or incorrect data.
- Freshness: All data generated by an IoT device is relevant in a temporal context. The timestamp of the data must be granted.

Finally, all the solutions to address the security requirements shall be scalable. Due to the nature of the IoT ecosystems, solutions that require constant human supervision or cannot scale up proportionally with the number of IoT devices are not applicable and therefore are not real solutions for the problem presented in section 1.1

1.3. Outline

This is an article-based thesis, which means that the thesis is a compendium of the most relevant publications achieved throughout the course of the doctoral program. The document is divided into three parts:

- Part I, PhD Dissertation: This first part aims to present the problem to be solved and its importance in the section 1.1, the objectives of the research in the section 1.2, the methodology followed during the PhD in the chapter 2 and the results achieved in the chapter 3.
- Part II, Publications: This part groups the publications obtained or submitted during this research. These publications support the achievements presented in Chapter 3.
- Part III, Conclusions: Finally, the third part explains the impact of the achievements in the chapter 8 and how these achievements fulfill the security requirements. In this last chapter, we also venture to predict the future of IoT and its security.

Four are the proposals that set the basis for this research. Three of these proposals have been published in high-impact factor journals, and one of them was sent for publication and is currently under review:

- Ernesto Gómez-Marín, Valerio Senni, Luis Parrilla, Jose L. Tejero López, Encarnación Castillo, and Davide Martintoni. *An innovative strategy based on secure element for cyber-physical authentication in safety-critical manufacturing supply chain*. Applied Sciences, 13(18), 2023.(Q2).
- Ernesto Gómez-Marín, Davide Martintoni, Valerio Senni, Encarnación Castillo, and Luis Parrilla. *Fine-grained access control with user revocation in smart manufacturing*. Electronics, 12(13), 2023. (Q2).
- Ernesto Gómez-Marín, Luis Parrilla, Jose L. Tejero López, Diego P. Morales, Encarnación Castillo. *Towards sensor measurement reliability in Blockchain* UNDER REVIEW by Sensors (Q2).
- Ernesto Gómez-Marín, Luis Parrilla, Gianfranco Mauro, Antonio Escobar-Molero, Diego P. Morales, and Encarnación Castillo. *Resekra: Remote enrollment using sealed keys for remote attestation*. Sensors, 22(13), 2022.(Q2).

Chapter 2

Methodology

*Start by doing what's necessary; then do
what's possible; and suddenly you are
doing the impossible.*

Francis of Assisi

This chapter presents the methodology followed to achieve the objectives introduced in the section 1.2. Additionally, this chapter presents the tools found, studied, and employed as the building blocks to design and construct the planes showcased in this thesis with remarkable success. Noticed that all the steps of the methodology had to be iterative repeated in order to create successful comprehensive solutions. For example, after finding a new tool and its limitations, it can be needed to reach back to the study of scenarios to investigate how this limitation can affect the scenario and thus to the final solution. The flow diagram of the complete methodology used in this thesis can be found in figure 2.1.

2.1. Study of the scenarios

The first step in this long path was performing a comprehensive study of the scenarios that envision using sensors and actuators but are currently blocked because the cybersecurity risks are higher than the benefits. The study was performed thanks to numerous surveys as [21, 22, 23, 24, 25] and also own knowledge about the industrial ecosystem gathered from the involvement of the Ph.D. student in the European project COLLABS [26].

The results of this study found four scenarios where data from the real world must be sent to a second party (centralized or decentralized) that will use them to execute critical actions of utmost responsibility: critical supply chain, industrial data-sharing, sending of measurements for critical actions, and edge computing. Each of them has its own requirements and limitations:

The first scenario is critical supply chain management. Here, critical parts

such as aircraft or luxury products travel between various stakeholders before reaching the end consumer. In this scenario, it is vitally important that stakeholders such as government regulators and end consumers can verify the legitimate origin of the products. This is a big challenge, considering stakeholders themselves are interested in manipulating the system and the products to obtain individual benefits. Also, Third-Trusted Parties (TTP) for data storage cannot be trusted due to the incapability of detecting data tampering. And the system must prevent organizations from accessing data that is not relevant to them. Therefore, the proposed solutions must be decentralized and address the risk that IIoT devices that track products will change their ownership among untrusted stakeholders and offer privacy between organizations.

In the second place, we found the necessity of facilitating data sharing within and without the organization while providing high confidentiality and integrity guarantees. In this scenario, many kinds of data are generated by the IIoT devices, and some of the data can contain proprietary knowledge, which diffusion can negatively impact the manufacturer. At the same time, many entities and stakeholders have a legitimate interest in accessing some of the data. In this scenario, it is essential to keep the principle of least knowledge and avoid entities accessing data that is not relevant to them. Additionally, it is necessary to prevent any data manipulation at storage to increase the IIoT data value.

Other scenarios require high veracity guarantees of measurements generated by IIoT devices belonging to third parties. In these scenarios, it is not enough to protect the IIoT data at rest; it must be protected from the first moment of generation to the data consumption. The end readers must be able to verify the security guarantees along all the data way. Additionally, the proposed solution must take into account the scalability and dynamism of IIoT devices.

Finally, in many cases, data must be processed directly where it is collected to reduce latency and data traffic. This concept is called edge computing in the SoA. At the same time, security guarantees must be provided. In this scenario, it is not enough to verify the correct data gathering of the devices; it is also necessary to remotely verify the correct data processing. The guarantee of this verification must be sharable between entities and allow a large number of IoT with discontinuous activity. At the same time, it must not affect device performance and avoid dependence on centralized servers in the cloud.

All these scenarios are very challenging and relevant in industry and other fields such as healthcare. The SoA has sought solutions to them, finding some with their advantages and disadvantages, and not always applicable. To find new solutions and stand out from the SoA, it is necessary to resort to all available old and especially new technologies.

2.2. Related technologies

A deep study of elements and tools that could be useful was performed, with the result of the three technologies that are detailed in the following.:

2.2.1. Cryptography

This study found cryptography a strong ally for the security of IoT ecosystem. A deep research was performed to find the advantages and limitations of each cryptographic algorithm, such as the extension attack in standard hash algorithms such as SHA-2 [27], or the possibility of using BLS signatures to generate a random value [28].

Particularly, this thesis research used hashes to send fast and effective proof of the integrity of a message. The algorithm SHA-2 is very widely used and improves the tools' compatibility. However, a very extended technology, Ethereum [29], uses a different hash algorithm, Keccak [30]. Therefore, the two algorithms have been used depending on the convenience of the adjacent tools and scenarios. We also used asymmetric cryptography for authentication through digital signatures. In this sense, the cryptography algorithm Elliptic Curve Digital Signature Algorithm (ECDSA) [31] has the advantage over the algorithm RSA due to the difference in the key sizes. The first one requires private keys of 256 bits, while the second one requires private keys of 2048 bits. This is a remarkable difference for memory-contained devices as most of the IoT devices. AES was the algorithm chosen for symmetric encryption. It is standardized by NIST, widely used, fast, and efficient without strong alternatives nowadays. We also used the Attributes-based Encryption from Waters [32]. It is a cryptographic algorithm published more than ten years ago. It offers plenty of possibilities, such as encrypting data without knowing its future reader but limiting their attributes, which suits perfectly in fine-grained authorization. However, it is criticized for the absence of revocation of users.

Finally, we found that the use of cryptography highly increases its value with the use of special dedicated hardware to secretly produce and store the secret credentials, the Hardware Security Modules (HSM).

2.2.2. Hardware Security Modules

Hardware Security Modules (HSMs) are devices secure by design that generate and store secure credentials to use later as inputs in cryptographic operations. The sensitive data as the private keys of asymmetric algorithms, are secretly stored there without a way of extracting it. Among their features, they even include ambient light sensors to detect physical hacking and deleting the data. Also, they hide their functionality with a wire mesh aimed to disable the chip and will also remove all sensible data if any of its electric circuits are

disturbed [33]. HSMs can have as complex functionality as those following the TPM 2.0 standard [34] that can perform remote attestation or can seal the use of the private keys to the fulfillment of some policies or combination of policies, which remarkable increments the versatility of the tool. Also, others more simple as smart-carts or model OPTIGA™ TRUST M [35] can be found in the market, which can merely perform the more basic cryptographic operations. Additionally, the study revealed the existence of novel HSM personalized for particular tasks as a solution by Dominic *et al.* [36] that was found as a perfect tool to make hardware oracles in [37].

HSMs were found as an excellent solution for remotely trusting IoT devices as they present tamper-resistance needed to avoid physical attacks on the devices; they include insulated environments that protect against various attacks, prevent unauthorized access and key compromise; and they offer software attestation solutions.

2.2.3. Blockchain

In every scenario, various stakeholders need to access data gathered or computed by IoT devices because of their location to perform their tasks. However, the stakeholders do not own these devices or have physical access. In some scenarios, they cannot even send messages to them. This fact presents a trust issue from the data consumers not only to the devices but to the owner's devices. In this context, blockchain has proved to be a very powerful tool to provide a root of trust in the cloud, e.i., trusting the information in the blockchain just because it was at some point accepted by the blockchain.

Blockchain is a distributed database formed by a set of records named "blocks." The information in there is protected via cryptography and the correct operation of most of the participants. Also, all new records must be authorized by all the participants who have an interest in collaborating correctly to the system. In public blockchains, the participants are called miners and get paid or punished so their correct behavior is the most profitable. It is the case of blockchains like Bitcoin [38] or Ethereum [29]. Instead, the consortium blockchains have much fewer participants; they need authorization to participate, and they do not get paid for behaving correctly. They act correctly because the correct functionality of the blockchain is indeed beneficial for them in some way. Hyperledger [39], an umbrella project and its blockchains stand out among the consortium blockchains.

An intensive study of blockchain solutions was performed in this thesis. It showed that while blockchain has robust features such as transparency, non-repudiability and excellent availability, it has the same time, some limitations: lack of confidentiality, non-deletable data and low efficiency. After acquiring a good understanding of the technology, new research solutions were developed by taking advantage of its attributes and solving the limitations with other technologies.

Additionally, this study revealed that not all the works in the state of the art properly attend to its limitations, leading to severe security risks as explained in [40].

2.3. Methodology for the achievement of the objectives

In order to achieve the different objectives enumerated previously, we have used a specific methodology adapted to the characteristics of our study. It can be seen in Figure 2.1, where the different steps and considerations are shown graphically. In the following sections, these steps are detailed.

2.3.1. Development of the Schemes

To develop a successful solution, it is needed to start with a theoretical presentation with the form of a general scheme. This research intensively used the Unified Modeling Language [41], particularly sequence and class diagrams. The development of the schemes poses limitations or requirements in the functionality that affect the final solution, e.g., defining beforehand a correct software status [42] or requiring constant communication of the data consumer with the key provider [40]. If these limitations are incompatible with the scenario, the scheme must be discarded partially or completely. Sometimes, it is necessary to go back to the study of the scenario in order to understand better the impact of the limitations.

2.3.2. Security analysis

This is a critical step in cybersecurity research, given that no experiments or quantitative results can prove the solution's security. Once the scheme was finalized, we performed a security analysis, ensuring that the theoretical solution guarantees all the required security features along with the full system functionality. Secondly, we systematically evaluate the resilience of the system by simulating cyber-attacks analogous to those observed in state-of-the-art scenarios, substantiating its robustness. After finding a security risk, it is necessary to go back to the scheme's development to solve the issue.

2.3.3. Prototype development

When the theoretical solution fits the scenario correctly in terms of functionality and security, the next step is the implementation of the solution. The implementation of the prototype was always realized using the real tools required in the scheme, e.g., commercial HSMs or decentralized blockchain networks. After the implementation of the prototype, we realized performance

analysis to find overall scalability limitations that could affect the implementation of our solution in a big IoT net. An unsatisfactory analysis leads to a new implementation to improve the prototype performance or redesign the scheme.

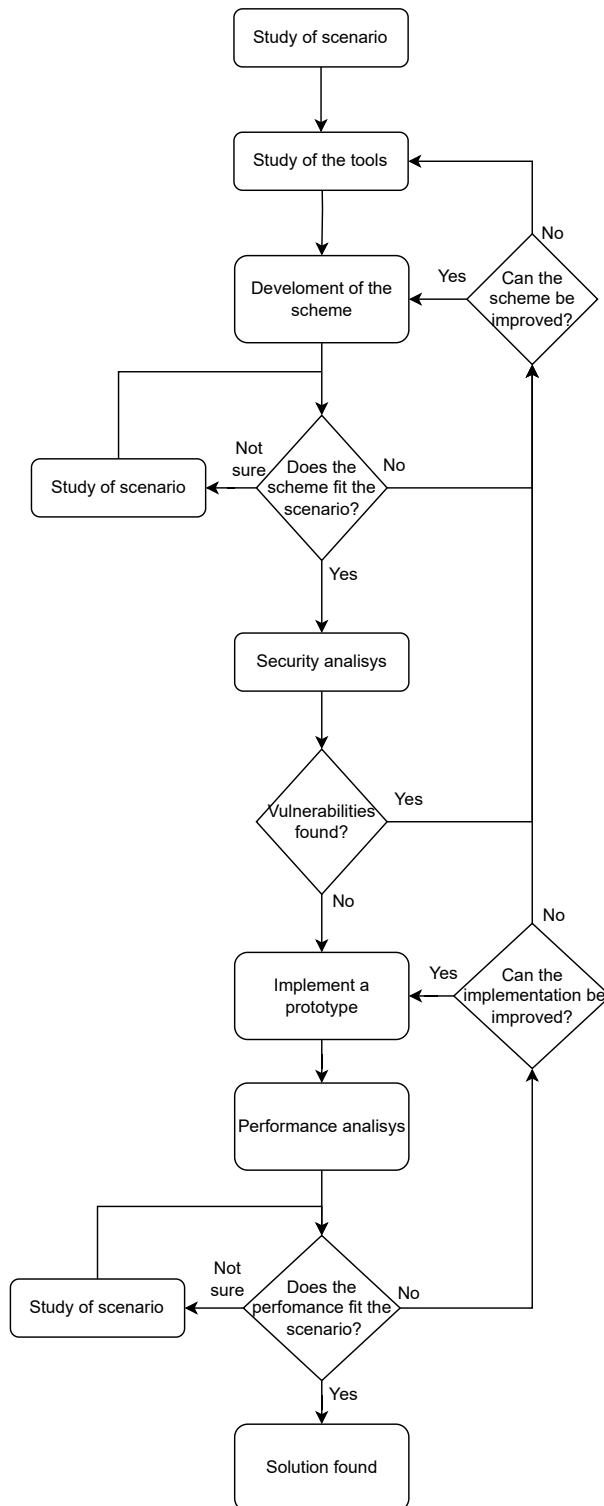


Figure 2.1: Flow diagram of the used methodology.

Chapter 3

Achievements

*Creativity is seeing what others see and
thinking what no one else ever thought.*

Albert Einstein

This chapter explains the results obtained by applying the methodology detailed above to achieve the thesis objectives. The results have given rise to three scientific publications and one publication that is currently under review. These publications are presented below, showing how to solve IoT node threats in different scenarios.

It is important to acknowledge the importance of the founding of the European Union's Horizon 2020 Research Innovation program to achieve these successful results. The help came under the grant agreement No. 871518, a project named, "A COMprehensive cyber-intelligence framework for resilient coLLABorative manufacturing Systems", COLLABS [26]. This project brought together relevant partners from the global industry, Renault [43], Philips [44] and Collins [45], in order to gather the current cybersecurity concerns of their factories and the desired operation scenarios they would like to protect. Subsequently, a larger consortium of equally relevant companies in the sector, such as Infineon [46] or Thales [47], took on the task of developing solutions to the challenges provided. This project created the perfect environment to design and develop the results shown below.

3.1. Research on IoT security in supply-chain

Firstly, the research took direction to address one of the fields with more requirements and risks in IoT security, the industry. Industry is heading towards a new industrial revolution, Industry 4.0, where a modern and efficient supply chain becomes a fundamental element for the company [48].

The supply chain is a complex field due to the need for companies to

collaborate with partners that are not completely reliable. When manufacturing products that require high quality, companies are concerned about the genuineness and, thus, the quality of the supplies. On the other hand, the confidentiality of their dealings and stakeholders is as high as in any other business. To offer a solution to these problems, we propose the use of blockchain to avoid the necessity of trusting in a TTP to manage the cloud operation. We chose a particular kind of blockchain widely used in the SoA Hyperledger Fabric (HF) because it provides tools to preserve confidentiality, which are explained in depth below. Additionally, we propose the use of an IoT device with an HSM as a fundamental part of our self-designed new cryptographic protocol to reliably and irrefutably declare the origin and ownership of the supplies.

We use HSMs with NFC communication as IoT device, called Smart-Tag (ST) hereafter, which resembles a smart card used in credit cards. STs are implemented on goods that are physically moved in the supply chain to identify the parties. We created a novel protocol so that the blockchain and all the nodes that are part of it, and the stakeholders involved in the supplychain can remotely verify the genuineness of the product remotely at each of the part's ownership transferences. To achieve this, the protocol uses a first digital signature made by the ST to propose in the blockchain the new owner of the supply, which is verified by the blockchain before proceeding. Then, the new owner makes a second digital signature to accept to be the new owner of the supply. The procedure can be observed in Figure 3.1. With this double-signed transfer, the authentication of the packet and acceptance of the new owner are forced in order to transfer a packet. The protocol includes the use of two nonces — *Challenge* and *Secret* in Figure 3.1 — to avoid some cyberattacks, which is explained in more detail in [49]. Thanks to this protocol, when the packet arrives at the final destination, the consumer can verify the genuineness of the part and check the entire path it has taken to reach him. This system can also demonstrate to a third party the reliable origin of the supplies and their quality.

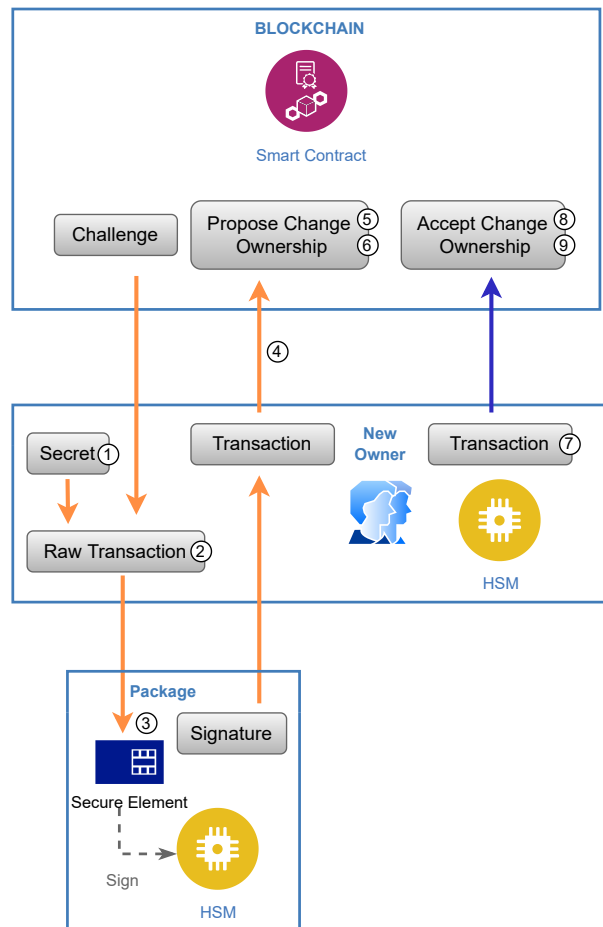


Figure 3.1: Two-step transaction infrastructure used to change the ownership of a package in the smart contract in the blockchain. [49]

However, the problem of applying a common blockchain tool in the industrial field is the lack of confidentiality that historically accompanies blockchains. In this scenario, this affects data such as the price, the manufacturer of the supplies or the requirements of the products, data that should not be made public to all stakeholders. To solve this problem and to grant a certain degree of confidentiality to the data depending on the organizations and entities, we use a type of databases offered by HF called Private Data Collection [50]. The private collections allow to store certain data from the databases only on selected nodes of the blockchain. Moreover, the hash of the private collections is stored in the ledger, i.e., in all the nodes of the blockchain; therefore, their integrity is still protected by all the system's nodes.

This solution is the first to deliver such a high level of trust in the product's path in a supply chain and simultaneously is the only one that proposes

using private collections to maintain confidentiality between partners while protecting the integrity of the data with blockchain. However, it has certain limitations. The use of Private Data Collections does not offer a perfect fine-grained authorization since the data access authorizations remain at a very high level, i.e., at the level of organizations, not at the level of individuals within the organizations considering their roles and attributes. On the other hand, despite being a consortium blockchain, and therefore cheaper and more efficient than a public blockchain, HF is not scalable enough to store a huge amount of data, such as that generated by the thousands of IoT devices in a factory. Finally, the IoT used, despite solving such an important use case as track in the supply chain, is far from being applicable to any scenario due to the limitations of a passive IoT, i.e., without a battery.

3.2. Research on Fine-Grained Authorization in IoT

The above solution presents a mechanism to offer a certain degree of authorization of access to grained data, but Private Data Collections can only allow or authorize access to data for entire organizations. This is a problem when different authorization levels are required within the organization itself, where, depending on roles and attributes, the people or individuals, readers hereafter, can access data. The complexity of this scenario can be observed in Figure 3.2, where many different internals and external entities having different access levels are requesting data stored in the same server. In fact, the handling of large quantities of sensitive data among a large number of people with heterogeneous privileges is a current state-of-the-art challenge. The use of a server to distribute the data according to entities and permissions does not give enough confidence to the data owners since any server administrator can access these data, while the servers can fail. On the other hand, encrypting the data with a key and giving the keys to the readers involves a complex management of accurate access control rules that easily leads to systems where everyone has access to more than they are entitled to or to much less than they need.

In the SoA it is common to try to solve this problem by applying a peculiar encryption algorithm called Attribute Based Encryption (ABE) or its later version Ciphertext-Policy ABE (CP-ABE). CP-ABE allows the definition of a combination of attributes joined by *AND/OR* logic functions called policy when encrypting a message, e.g., "(Admin and CEO) or HumanResources". Once the data is encrypted under this policy, only those users with decryption keys (*DK*) containing the necessary attributes can decrypt the data. The key provider generates and distributes these keys. This is a suitable algorithm for fine-grained authorization as it allows to encrypt one's own data by defining a policy and trust that only people with the right attributes will be able to read the data, even without knowing the identity of these people while

encrypting. However, this algorithm has certain problems that prevent it from being applicable in the real world. One of them, and possibly the most negative, is the absence of a revocation mechanism. Once the key provider no longer trusts a user with a decryption key, they have no mechanism through which they can revoke the user's key to prevent them from accessing data anymore.

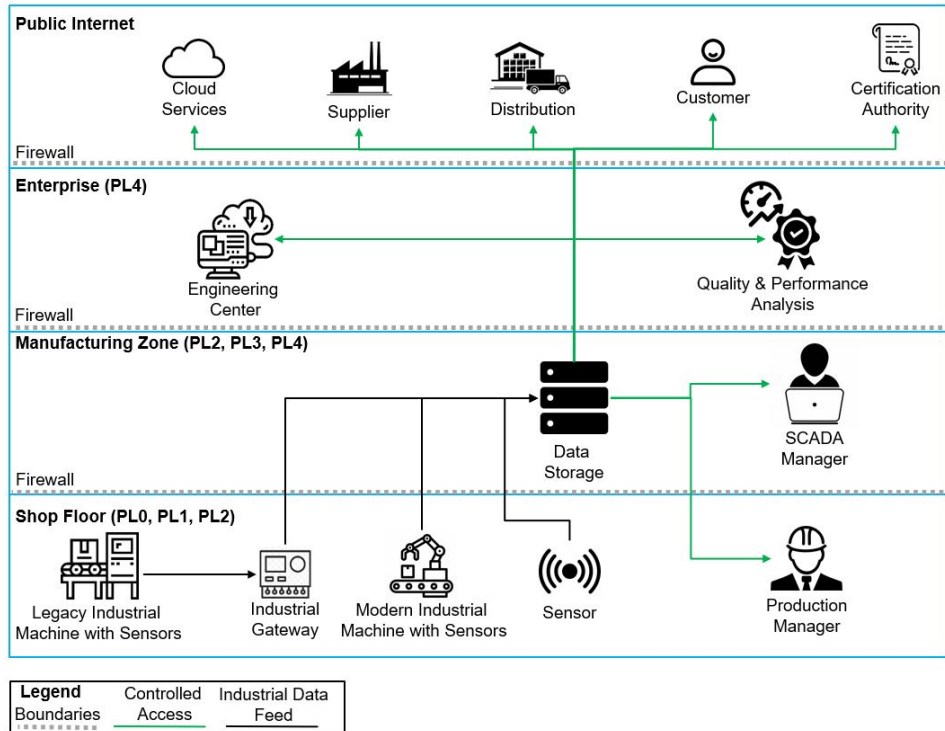


Figure 3.2: High-level scheme of factory scenario [40].

Apart from this important problem, the lack of reliability, traceability, and availability in data storage services results in the loss of interest in an encryption mechanism as complex as CP-ABE. To increase the trust in data encrypted with CP-ABE, in the absence of a TTP for data storage, [51] proposes the use of the InterPlanetary File System (IPFS) [52] to store the encrypted data, called ciphertext. IPFS is a decentralized storage capable of storing huge data volumes in a scalable way with high availability, achieving data integrity by using the data hash as its identifier. The problem with this service is the absence of a mechanism to store the hashes and identify the desired data to be downloaded. To solve it [51] proposes the use of blockchain to store the hashes, the metadata of each data and the identification of the author. This solution is an excellent proposal, but it does not have any revocation mechanism and does not allow multiple writers, which is essential to be applied to IoT. Our proposed approach builds on [51], and improves it

by providing revocation with a self-designed mechanism, strong authenticity guarantees for multiple data senders, and a dynamic reader registry, standing out significantly over the SoA.

In our self-designed revocation method, we combine direct revocation and indirect revocation with the goal of the maximum number of readers to serve. Indirect revocation consists of updating the *DK* of all non-revoked users with each revocation. The problem with this mechanism is that generating decryption keys for all readers is a computationally expensive process. So when the number of readers is too high, they have to wait a long time to receive their new *DK* with each revocation, making this mechanism not scalable. We mitigate this problem using what we call *backup decryption keys* (*bDK*). All the users already have their new decryption keys encrypted with a unique symmetric key. So, when a new revocation occurs, the key provider only has to publish a list with the symmetric keys needed to decrypt the *bDK* of all the non-revoked users. Then, the Key provider will need some time to provide the new *bDK* after a revoke, but it will not produce a service interruption because the non-revoked users already have their new decryption keys.

As we indicated, we combine it with direct revocation. In direct revocation, there is a list that is updated every time there are revocation and the encryptors need to be aware of this list when encrypting to exclude the revoked users. The problem with this solution is that the list ends up becoming excessively large with the number of users, and the encryption is complicated proportionally to this number. To solve this problem, we create a set of groups and subgroups to distribute the users by assigning particular attributes to them, *revocation attributes*. These are ABE attributes with particular properties to join and separate users between groups and subgroups. Then, this list is proportional to the number of groups solving the problem of the very long lists. Also, the solution does not revoke users but entire subgroups formed of many users. However, this does not impact the performance of our solution thanks to the use of *bDK* explained above.

Additionally, we proposed and implemented OpenID [53] in the system to provide a dynamic enrolment of readers in the system. Finally, we provide strong authenticity mechanisms for the income data through the use of HSM in the IoT devices to sign the encrypted data. All the public keys of the IoT devices are registered in an available list in the blockchain called *IoT list* that is queried to verify the incoming data.

Thus, this system can manage 864,000 users with the computational power of a laptop and manages to solve the problems encountered in the SoA for this cybersecurity challenge. This work is an important step forward in the field of information security in IoT, as it demonstrates that it is possible to share data from many IoT nodes to many clients while maintaining all the security guarantees. However, the automatic and secure enrollment of

IoT devices in the system, even if necessary, is beyond the scope of this work. Additionally, the trustworthiness of the IoT devices is not considered.

3.3. Research on PKI in IoT and Sensors

As explained in the previous section, the autonomous enrollment of IoT devices in the system is not included in the solution because it is out of the scope. However, it is essential to get a scalable solution with the lowest human intervention, which is one of the objectives of this thesis. Also, the trustworthiness and freshness of the data are vital to use this data as the base for critical operations or actions.

Registering a vulnerable or unauthorized device is a threat to the network. To avoid this, the registries collect authorization from a trusted part of the system, which mitigates this risk. However, blockchain is used when no trusted party exists for all partners. At the same time, registration needs to be convenient and scalable when dealing with IoT devices. For these reasons, registering an IoT device in a decentralized network is a challenge. In fact, the works of the SoA [54, 55, 56] consider that registration must be done in a trusted environment, offline phase or in the presence of all partners, which is not applicable.

To solve this complex issue in our work [37], we defined a new concept called smart certificate authority. This mechanism aims to allow a smart contract to verify whether a device is trustworthy in the enrollment phase and store its identification in case of success. For this, we first define all the requirements for an IoT to be trusted and belong in our decentralized system, e.g., device manufacturer, device model, life years, ownership, etc. Then, all the verification of the defined requirements is implemented in a smart contract. Once defined, the manufacturer's digital certificates are mapped to the device identifier, i.e., its public key. Also, the device's owner can claim ownership of his device through a digital signature. In this way, the IoT device itself can query the smartcontract being identified and validated for the registration. Then, all blockchain nodes verify through the manufacturer's digital certificate and the owner's signature the veracity of the device through the defined requirements. Finally, the verified device identifier is stored in the blockchain itself to avoid this verification being performed every time the device wants to communicate with the blockchain. Notice that for a successful secure registration using smart certificate authority, a HSM in the IoT device is needed to protect its credentials.

This solution has great relevance since it allows us to verify the hardware and the reliable manufacturing of an IoT device in a decentralized and remote way. Thus, we use this mechanism to verify that a device is a sensor with a novel secure hardware architecture that can prove the trustworthiness and integrity of the gathered data. This is a particular sensor proposed by Dominic

et al. in [36], hereafter called Secure Sensor (SS). This hardware architecture can create and securely store a special sealed private key. This sealed key can only be used to sign measurements the sensor generates. The combination of SS with smart certificate authorities is a great achievement for the objectives of this thesis since, given a measurement and a signature, it is possible to check the trustworthiness and integrity of the data even without first-hand knowledge of the IoT device.

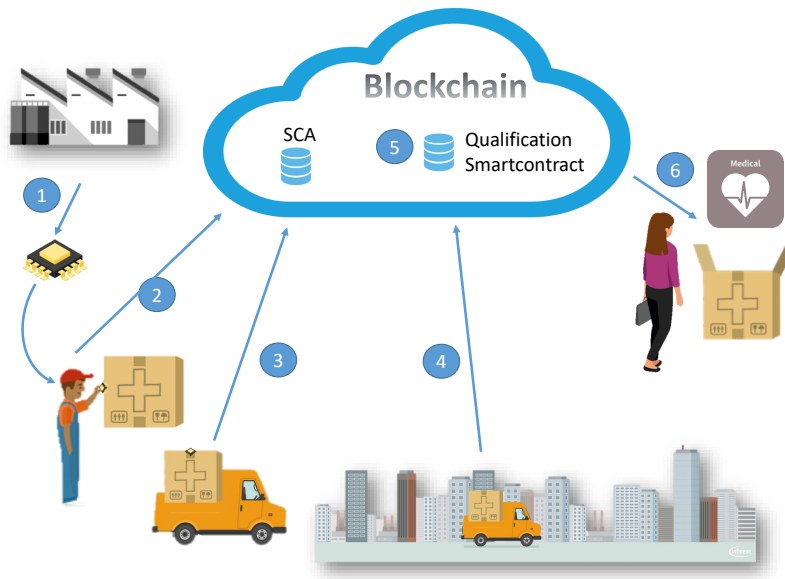


Figure 3.3: High-level scheme of a SS installed in a package of a supply chain sending data to the Blockchain [37].

However, SS cannot ensure the freshness of the measurement. Even knowing that a reliable device generated the measurement and was not manipulated, SS cannot ensure that the measurement is not old or from a particular time interval. To solve this, we use a nonce in the signature. During the signing by SS of the measurement using the sealed key, SS allows the inclusion of a nonce in the data to be signed of a nonce. Using a nonce for data freshness in cryptography is common in one-to-one communication. However, if a single entity generates the random nonce, only this entity will be able to validate the freshness of the data, and the other entities will have to rely on the first one. On the other hand, using time as a nonce to indicate when data was generated and signed is unreliable because absolute trusted clocks in HSMs are a challenge. In our work, we solve the problem by using as nonce the blockhash of Ethereum. This is a questionable solution because this number is not completely random and can be manipulated to a certain degree, reason why it is not used in the SoA. However, we did a statistical

analysis of the predictability of this number and we proved that this solution is reliable and secure as freshness evidence of the measurement by delivering a confidence interval of 72 seconds. Figure 3.3 shows how this solution can be used to prove the correct temperature of a product like medical material or luxury food along all the cold-chain.

This work has great relevance as it is the first SoA implementation of an IoT device that delivers data to the blockchain with such reliability: trustworthiness, freshness and authentication. The implementation is performed on Ethereum, the most important blockchain that allows synchronous execution of formal logic by all nodes. However, the problem with this solution is that it only allows the delivery of data without preprocessing, which somewhat limits its possible implementation scenarios.

3.4. Research on remote attestation in IoT

The above solutions achieve great results in terms of information security guarantees for transmitting data from IoT devices to the cloud. However, as soon as IoT devices process or manipulate the data as part of the system, these security guarantees are lost. Therefore, the data has to be transmitted raw in order to maintain trust in the origin of the data. This can be a drawback in many scenarios, as raw data can be much heavier than net data, with negative repercussions on the network. On the other hand, the use of blockchain requires communicating with distant servers, which results in increased latency even if the IoT node and the end user are in the same vicinity. Precisely, edge computing is a new technology field that seeks to solve these problems [57].

Edge computing proposes to process data close to where it is generated and where it is consumed because of its advantages in terms of data traffic and latency reduction. However, from a cybersecurity point of view, edge computing opens up many new challenges. As in cloud computing, different layers of technology must be protected (from cloud to IoT), but edge computing also has to provide a distributed global connectivity of heterogeneous devices between the different layers. Rodrigo Roman *et al.* [58] define it as "the worst-of-all-worlds". Such is the complexity of cybersecurity in edge computing and the negative impacts of a cyber attack that the advantages of edge computing can be undermined by the losses derived from excessive trust in the technology [58].

In order to ensure the reliability of the data sent and processed in this scenario, remote attestation is a very valuable tool. Remote attestation allows an entity (Verifier) to check the correct state of the software of a remote device (Attester) remotely. However, none of remote attestation protocols in the SoA is suitable for edge computing. This is due to the need for a protocol with several unique properties in order to provide remote attestation

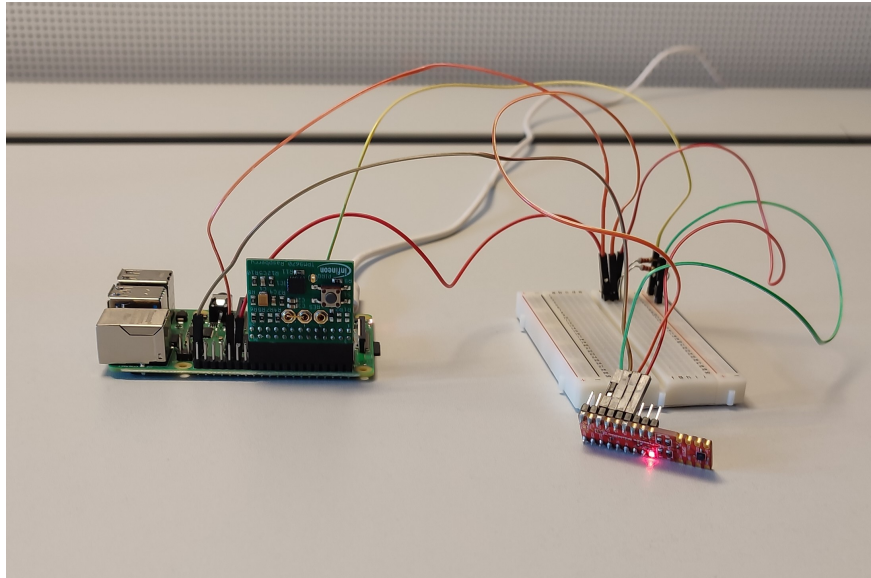


Figure 3.4: Experimental setup. Raspberry Pi 4 with TPM (green hardware) and pressure sensor (red hardware) [42].

in this complicated ecosystem: dynamicity to quickly include and exclude devices; high scalability due to the large number of devices to be attested; decentralization since the readers and devices belong to different entities or organizations; and correct analysis of devices with intermittent activity, because IoT devices are not always online.

To solve this problem, we consider an IoT device with standard secure hardware: a Core Root of Trust of Measurement (CRTM) [59] and a Trusted Platform Module (TPM) [34]. Next, we created a Public Key Infrastructure (PKI) using the appropriate cryptography and security guarantees of these standard hardware so that the Verifier and the Attester do not have to share any shared secret to perform the remote attestation. Also, thanks to this PKI, the Attester can create asymmetric keys and prove that it is sealed in such a way that it can only be used if the IoT device has a correct software status. To prove the validity of the proposed solution, we built a demonstration using a real commercial TPM from Infineon Technologies, TPM IRIDIUM9670 TPM2.0 [60], deployed over a Raspberry Pi 4B with a pressure sensor DPS310 Pressure Shield2Go [61]. The set-up can be observed in figure 3.4

With these two technical achievements, we show that with proper implementation, our RESEKRA protocol achieves the desired decentralized, remote attestation. It allows the end user to rely on the correct status of any IoT device in the center without requiring any trusted third party to verify it, just by knowing the device's name. In this way, we got to integrate remote attestation into edge computing by complying with the necessary features.

References

- [1] Somayya Madakam, Vihar Lake, Vihar Lake, Vihar Lake, et al. Internet of Things (IoT): A literature review. *Journal of Computer and Communications*, 3(05):164, 2015.
- [2] Kosmatos Evangelos A, Tselikas Nikolaos D, and Boucouvalas Anthony C. Integrating rfids and smart objects into a unifiedinternet of things architecture. *Advances in Internet of Things*, 2011, 2011.
- [3] Renu Aggarwal and Manik Lal Das. RFID security in the context of "Internet of Things". In *Proceedings of the First International Conference on Security of Internet of Things*, pages 51–56, 2012.
- [4] K.V.S. Rao, P.V. Nikitin, and S.F. Lam. Antenna design for UHF RFID tags: a review and a practical application. *IEEE Transactions on Antennas and Propagation*, 53(12):3870–3876, 2005.
- [5] Rapeepat Ratasuk, Benny Vejlgaard, Nitin Mangalvedhe, and Amitava Ghosh. Nb-iot system for m2m communication. In *2016 IEEE Wireless Communications and Networking Conference Workshops (WCNCW)*, pages 428–432, 2016.
- [6] Silvia Liberata Ullo and G. R. Sinha. Advances in smart environment monitoring systems using iot and sensors. *Sensors*, 20(11), 2020.
- [7] Krishna Keshob Paul, Jishnu Dev Roy, Sourav Sarkar, Sena Kumar Barai, Abu Sufian, Sachin Kumar Gupta, and Ashutosh Srivastava. Smart agriculture using uav and deep learning: A systematic review. *Internet of Things*, pages 1–16, 2022.
- [8] Robert Sparrow and Mark Howard. Robots in agriculture: prospects, impacts, ethics, and policy. *precision agriculture*, 22:818–833, 2021.
- [9] Kumar Mandula, Ramu Parupalli, CH.A.S. Murty, E. Magesh, and Rutul Lunagariya. Mobile based home automation using internet of things(iot). In *2015 International Conference on Control, Instrumentation, Communication and Computational Technologies (ICCICCT)*, pages 340–343, 2015.

-
- [10] U Cisco. Cisco annual internet report (2018–2023) white paper. *Cisco: San Jose, CA, USA*, 2020.
 - [11] Omnia Abu Waraga, Meriem Bettayeb, Qassim Nasir, and Manar Abu Talib. Design and implementation of automated iot security testbed. *Computers & security*, 88:101648, 2020.
 - [12] Iqbal H Sarker, Asif Irshad Khan, Yoosef B Abushark, and Fawaz Alsolami. Internet of things (iot) security intelligence: a comprehensive overview, machine learning solutions and research directions. *Mobile Networks and Applications*, pages 1–17, 2022.
 - [13] Irdeto. Global connected industries cybersecurity survey, 2019.
 - [14] Oludare Isaac Abiodun, Esther Omolara Abiodun, Moatsum Alawida, Rami S Alkhawaldeh, and Humaira Arshad. A review on the security of the internet of things: challenges and solutions. *Wireless Personal Communications*, 119:2603–2637, 2021.
 - [15] Amazon Web Service. Summary of the amazon s3 service disruption in the northern virginia (us-east-1) region, 2017.
 - [16] Sebastian Moss. Microsoft azure suffers outage after cooling issue, 2018.
 - [17] Yingwei Fu. Alibaba cloud reports io hang error in north china, 2019.
 - [18] Peter Judge and Dan Swinhoe. Cooling failure brings down google cloud data center in london on uk’s hottest day, 2022.
 - [19] Michael Howard and Steve Lipner. *The security development lifecycle*, volume 8. Microsoft Press Redmond, 2006.
 - [20] Dan Liu, Zheng Yan, Wenxiu Ding, and Mohammed Atiquzzaman. A survey on secure data analytics in edge computing. *IEEE Internet of Things Journal*, 6(3):4946–4967, 2019.
 - [21] Moreno Ambrosin, Mauro Conti, Riccardo Lazzeretti, Md Masoom Rabhani, and Silvio Ranise. Collective remote attestation at the internet of things scale: State-of-the-art and future challenges. *IEEE Communications Surveys & Tutorials*, 22(4):2447–2461, 2020.
 - [22] Shuai Wang, Hao Lu, Xingkai Sun, Yong Yuan, and Fei-Yue Wang. A novel blockchain oracle implementation scheme based on application specific knowledge engines. In *2019 IEEE International Conference on Service Operations and Logistics, and Informatics (SOLI)*, pages 258–262, 2019.

-
- [23] Hamda Al-Breiki, Muhammad Habib Ur Rehman, Khaled Salah, and Davor Svetinovic. Trustworthy blockchain oracles: Review, comparison, and open research challenges. *IEEE Access*, 8:85675–85685, 2020.
- [24] Abdeljalil Beniiche. A study of blockchain oracles. *arXiv preprint arXiv:2004.07140*, 2020.
- [25] Olivér Hornyák and George Farid Alkhoury. Smart contracts in the automotive industry. In Károly Jármai and Katalin Voith, editors, *Vehicle and Automotive Engineering 3*, pages 148–157, Singapore, 2021. Springer Singapore.
- [26] Collabs-871518 project website. <https://www.collabs-project.eu/>. Accessed: 2022-08-05.
- [27] Danilo Gligoroski. Length extension attack on narrow-pipe sha-3 candidates. In Marjan Gusev and Pece Mitrevski, editors, *ICT Innovations 2010*, pages 5–10, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.
- [28] Ben Edgington. Upgrading ethereum a technical handbook on ethereum’s move to proof of stake and beyond, part 2: Technical overview, the building blocks, randomness. https://eth2book.info/capella/part2/building_blocks/randomness/, Accessed: 2023-10-08.
- [29] Ethereum Foundation. Ethereum white paper. <https://ethereum.org/en/whitepaper/>, Last accessed on 2022-08-18.
- [30] National Institute of Standards and Technology. Nist selects winner of secure hash algorithm (sha-3) competition. <https://www.nist.gov/news-events/news/2012/10/nist-selects-winner-secure-hash-algorithm-sha-3-competition>, Last accessed on 2023-11-13.
- [31] Don Johnson, Alfred Menezes, and Scott Vanstone. The elliptic curve digital signature algorithm (ecdsa). *International journal of information security*, 1:36–63, 2001.
- [32] Brent Waters. Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization. In *Public Key Cryptography–PKC 2011: 14th International Conference on Practice and Theory in Public Key Cryptography, Taormina, Italy, March 6-9, 2011. Proceedings 14*, pages 53–70. Springer, 2011.
- [33] Dan Goodin. Ex-army man cracks popular security chip. https://www.theregister.com/2010/02/17/infineon_tpm_crack/, Last accessed on 2023-11-13.
- [34] Trusted Computing Group. Tpm 2.0 library, Mar 2021.

- [35] Optiga™ trust m sls32aia. Accessed: 2023-10-08.
- [36] Dominic Pirker, Thomas Fischer, Harald Witschnig, Rainer Matischek, and Christian Steger. Trustful remote-sensing architectures based on hardware-security. In *2021 IEEE 12th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON)*, pages 0256–0263. IEEE, 2021.
- [37] Ernesto Gómez-Marín, Luis Parrilla, José L. Tejero López, Diego P. Morales, and Encarnación. Castillo. Towards sensor measurement reliability in blockchain. *UNDER REVIEW by Sensors*, 2023.
- [38] Satoshi Nakamoto. Bitcoin : A Peer-to-Peer Electronic Cash System. *White Paper*, pages 1–9, 2008.
- [39] TheLinuxFoundation. Hyperledger foundation. <https://www.hyperledger.org/>, Last accessed on 2023-11-13.
- [40] Ernesto Gómez-Marín, Davide Martintoni, Valerio Senni, Encarnación Castillo, and Luis Parrilla. Fine-grained access control with user revocation in smart manufacturing. *Electronics*, 12(13), 2023.
- [41] Grady Booch. *The unified modeling language user guide*. Pearson Education India, 2005.
- [42] Ernesto Gómez-Marín, Luis Parrilla, Gianfranco Mauro, Antonio Escobar-Molero, Diego P. Morales, and Encarnación Castillo. Resekra: Remote enrollment using sealed keys for remote attestation. *Sensors*, 22(13), 2022.
- [43] Renault group, constructeur automobile. <https://www.renaultgroup.com/>, Last accessed on 27-10-2023.
- [44] Koninklijke philips electronics n.v. <https://www.philips.de/>, Last accessed on 27-10-2023.
- [45] Collins aerospace. <https://www.collinsaerospace.com/>, Last accessed on 27-10-2023.
- [46] Infineon technologies ag. <https://www.infineon.com/>, Last accessed on 27-10-2023.
- [47] Thales. <https://www.thalesgroup.com/>, Last accessed on 27-10-2023.
- [48] Maryam Abdirad and Krishna Krishnan. Industry 4.0 in logistics and supply chain management: A systematic literature review. *Engineering Management Journal*, 33(3):187–201, 2021.

-
- [49] Ernesto Gómez-Marín, Valerio Senni, Luis Parrilla, Jose L. Tejero López, Encarnación Castillo, and Davide Martintoni. An innovative strategy based on secure element for cyber-physical authentication in safety-critical manufacturing supply chain. *Applied Sciences*, 13(18), 2023.
- [50] Linux Foundation. Hlf private collection, 2022. <https://hyperledger-fabric.readthedocs.io/en/release-2.2/private-data/private-data.html>, Last accessed on 29-06-2023.
- [51] Shangping Wang, Yinglong Zhang, and Yaling Zhang. A blockchain-based framework for data sharing with fine-grained access control in decentralized storage systems. *IEEE Access*, 6:38437–38450, 2018.
- [52] Juan Benet. Ipfs-content addressed, versioned, p2p file system. *arXiv preprint arXiv:1407.3561*, 2014.
- [53] OpenID Foundation. Openid. <https://openid.net/>, Last accessed on 2023-06-19.
- [54] Miguel Calvo and Marta Beltrán. Remote attestation as a service for edge-enabled iot. In *2021 IEEE International Conference on Services Computing (SCC)*, pages 329–339, 2021.
- [55] Hailun Tan, Gene Tsudik, and Sanjay Jha. Mtra: Multi-tier randomized remote attestation in iot networks. *Computers & Security*, 81:78–93, 2019.
- [56] Jonathan Heiss, Anselm Busse, and Stefan Tai. Trustworthy pre-processing of sensor data in data on-chaining workflows for blockchain-based iot applications. In *International Conference on Service-Oriented Computing*, pages 133–149. Springer, 2021.
- [57] Yinhao Xiao, Yizhen Jia, Chunchi Liu, Xiuzhen Cheng, Jiguo Yu, and Weifeng Lv. Edge computing security: State of the art and challenges. *Proceedings of the IEEE*, 107(8):1608–1631, 2019.
- [58] Rodrigo Roman, Javier Lopez, and Masahiro Mambo. Mobile edge computing, fog et al.: A survey and analysis of security threats and challenges. *Future Generation Computer Systems*, 78:680–698, 2018.
- [59] David Cooper, William Polk, Andrew Regenscheid, and Murugiah Sompaya. Bios protection guidelines - nist, Apr 2011. Accessed: 2023-10-08.
- [60] Infineon Technologies. Iridium9670 tpm2.0 linux. <https://www.infineon.com/cms/en/product/evaluation-boards/iridium9670-tpm2.0-linux/>, Last accessed on 2022-05-23.

- [61] Infineon Technologies. Dps310 pressure shield2go. <https://www.infineon.com/cms/en/product/evaluation-boards/s2go-pressure-dps310/>, Last accessed on 2022-06-21.

Part II

Publications

Chapter 4

An Innovative Strategy Based on Secure Element for Cyber–Physical Authentication in Safety-Critical Manufacturing Supply Chain

Ernesto Gómez-Marín ^{1,*}, Valerio Senni ², Luis Parrilla ³, Jose L. Tejero López ¹, Encarnación Castillo ³ and Davide Martintoni ²

1. Infineon Technologies AG, Am Campeon 1-15, 85579 Neubiberg, Germany

2. Applied Research & Technology, Collins Aerospace, 00185 Rome, Italy

3. Departamento de Electrónica y Tecnología de Computadores, Universidad de Granada, 18071 Granada, Spain

MDPI Applied Science

- Received: 1 August 2023 , Revised: 5 September 2023, Accepted: 13 September 2023, Published: 19 September 2023
- DOI: 10.3390/app131810477
- Impact factor: 2.7
- JCR Rank: 42/90 in category Engineering, Multidisciplinary (Q2)

Abstract: The accurate tracking of every production step and related outcome in a supply chain is a stringent requirement in safety-critical sectors such as civil aviation. In such a framework, trusted traceability and accountability can be reliably and securely managed by means of blockchain-based solutions. Unfortunately, blockchain cannot guarantee the provenance and accuracy of the stored information. To overcome such a limitation, this paper proposes a secure solution to strongly rely on the tracking information of the physical assets in the supply chain. The proposed solution exploits Hardware Security Modules (HSMs) to provide required cryptographic primitives through a Near-Field Communication (NFC) connection. In our approach, each transfer of the assets is authenticated, verified, and recorded in the blockchain through the HSM. Transaction entries are signed, thus providing a guarantee of ownership and authenticity. The proposed infrastructure has been subject of an exhaustive security analysis and proved resilient against counterfeiting attempts, stakeholder repudiations, and misleading information.

Keywords: Supply chain; Industry 4.0; blockchain; Hardware Security Module (HSM); Near-Field Communication (NFC); information tracking.

4.1. Introduction

In the era of Industry 4.0, a modern and efficient supply chain is key for all company success [1]. A close integration of stakeholders, processes, and resources enables interconnected, automated, and correct decision making, which is vital to maximizing efficiency and boosting overall productivity [2]. With this objective in mind, Supply Chain Management (SCM) integrates the multiple organizational entities and coordinates material, information, and financial flows across the supply chain [3].

However, supply chains are inherently complex, spanning multiple organizations and different physical locations and systems, thereby presenting significant challenges in terms of transparency, traceability, and trust. According to the Organization for Economic Cooperation and Development (OECD), it is estimated that, in 2019, there were USD 19 billion worth of counterfeits in the EU alone, 5.8% of their total imports [4]. Thus, modernizing SCM toward Industry 4.0 is an important and complex research field [5, 6, 7].

To approach this challenge, numerous studies strongly advocate avoiding centralized solutions and promoting the use of blockchain in order to provide a trusted and transparent recording of transactions and events of the collaboration between multiple stakeholders [8, 9, 10]. Blockchain, a distributed and immutable ledger, offers a decentralized and transparent platform to record and verify transactions. It eliminates the need for a trusted central authority, enhances data security, and enables real-time access to a shared ledger, fostering trust and collaboration among supply chain participants.

However, its effectiveness is limited when it comes to bridging the gap between the digital realm and the physical world, i.e., consistently reflecting events in the real world [11]. To address this challenge, a trusted entity, known as an “oracle”, is required to record data in the real world and to store them into the blockchain system [12]. Nonetheless, ensuring the reliability and accuracy of real-world data coming from oracles to maintain the trust of recorded information in blockchain poses a formidable challenge, commonly referred to as the “oracle problem” [12]. Historically, in a supply chain context, the provenance of physical assets does not have guarantees, thereby limiting the blockchain-based SCM tracking capabilities. Here, the Internet of Things (IoT) emerges as a prominent solution due to its ability to provide real-time information from diverse sources [13].

Multiple works suggest the use of RFID tags as IoT devices to identify the product and combine it with Blockchain to store the legitimate data of the product’s provenance. However, due to the technology limitation of RFID tag, almost all these solutions require a centralized infrastructure for a secure implementation [14], which makes them unsuitable for integration with a Blockchain-based SCM. On the other hand, those few solutions that are decentralized are extremely complicated to implement securely [15, 16], or they do not consider the security of the IoT devices interacting with the supply chain physical flow [17]. The reliability of the data recorded in the SCM is extremely relevant in supply chains where the value of an asset is closely related to their origin (e.g., medical drugs, luxury products, safety-critical parts). In this context, there would be high economic rewards in tampering with these data source IoTs to insert malicious track information and therefore to enter counterfeits in the supply chain.

In order to solve these challenges, our scheme does not need a third trusted party or a centralized infrastructure, while making the tags unclonable. To achieve this, our work proposes incorporating NFC [18] tags in the assets tracked by the supply chain and using high-performance asymmetric cryptography, an Elliptic Curve Digital Signature Algorithm (EDCSA) [19]. In this way, there is no need to access sensitive information to identify the tags, ensuring the non-duplicable identity. Additionally, we propose a cryptographically secure mechanism to make Blockchain witness each transfer of the asset’s ownership to ensure the reliable provenance and path of the product. With this solution, all stakeholders can easily check the unique identity of any asset and securely rely on the information about its origin and each of its owners during its entire life cycle. Finally, thanks to the secure implementation, the proposed system enforces strict data access control to address one of the biggest technological challenges of the Blockchain-based supply chain: the privacy [20].

The rest of the paper is structured as follows: Section 4.2 highlights the motivations driving this research; Section 4.3 provides an overview of related

works and solutions; Section 4.4 reports the main concepts used for the formulation of the solution; Section 4.5 presents the approach designed by this research. In Section 4.6, a specific prototype of the solution is presented, which is finally used in Section 4.7 to evaluate the overall approach from a security point of view. Section 4.8 concludes the research and analyzes the future steps.

4.2. Motivation and Scenario

The state of the art on supply chain advancements are studied across a wide range of products, industries, and markets, such as health products or food tracking. In order to highlight the issues considered in this research, we will focus on a specific and critical scenario in the context of the aerospace industry. In avionic manufacturing, the airplanes' basic components (i.e., actuation, propulsion, navigation, air-quality, etc.) might be manufactured by multiple different companies in a tiered approach [21], which ultimately converges to create a singular, sophisticated product: an airplane. In this scenario, not only does the plane have to pass through several validations and certification processes, but also, individually, each of the plane's parts is subject to specific regulations with their correlated verification. Passing through the certification process and proving adherence to the quality standards is complex and time-consuming. And still, risks brought by the supply chain are a source of problems more extensive than the design and equipment risks in the aerospace stand-alone devices [22]. For this reason, aerospace assemblers are interested in getting closer and raising the control over the second-tier suppliers to reach better coordination and increase competitiveness [23]. In this context, the proposed approach aims to reduce risks and potential errors, mitigate costs, and increase the trust that due diligence is properly executed at all supply chain stages.

Taking into consideration this context and motivation, in the following, we explain the reference use case of the current work. An authorized entity, e.g., a supply chain manager from an airplane assembler, when receiving a part (e.g., an instrument panel), identifies the part and obtains from a database the information regarding the manufacturer, the certificates, and the historical owners of this part. Then, this information can be used to validate whether the asset can be mounted on a plane. Finally, the manufacturer proves adherence to the quality standards in each of the components to a third-party reviewer, like a governmental administrator. Using the STRIDE methodology [24] and the analysts of Hendrik S.B. and Evi H. [25], we identify the seven threats types of the use case in Table 4.1. These threats are particularly challenging in this scenario due to the nature of the supply chain, where any stakeholder is also considered a possible attacker if it can obtain an economic benefit in some way. For example, an airplane assembler

performs a tamper attack to delete part of the life cycle of an asset to insert a cannibalized asset, saving money and still “proving” adherence to the quality standards, or a distributor may make a spoofing attack to duplicate the identity of an asset and insert a counterfeit in the supply chain.

Table 4.1: High-level threats related to the cyber–physical link.

Threat Type	Identified Threat
Spoofing [24]	An attacker falsifies the identity of an asset. It can be used to insert a counterfeit part in the supply chain linking it to an existing SCM record, or to send illegitimate data or modify data records, respectively.
Tampering [24]	Data records are intentionally inaccurate. An attacker sends false information about the ownership of the asset or its certificates, or because the data records at rest are modified.
Repudiation [24]	An attacker disputes a recorded data’s authorship or a legitimate data modifications’ authorship.
Information disclosure [24]	An external entity accesses data traffic or data records.
Denial of Service [24]	An attacker makes the SCM inoperable temporarily or permanently.
Elevation of Privileges [24]	A stakeholder uses its privileges to access/-modify data that this stakeholder should not be allowed or should be allowed only when it physically owns the asset.
System misuse [25]	Several stakeholders collaborate to ignore the verification of the assets in order to include counterfeits, or the validation of the asset is not performed systematically due to lack of concern of the organizations or the employers.

The threat analysis highlights that the link between physical parts and their cyber representation is a potential attack point that can lead to different threats to the supply chain. This emphasizes the necessity of a secure approach for parts hyperlinking and data storage. To mitigate the risks identified in a safety-critical industrial supply chain, the solution should be formulated in a

way that does not rely exclusively on the trustworthiness of a single supply partner or an external party. Furthermore, the procedure should be designed as an easy-to-operate technology.

4.3. Related Work

Various techniques have been proposed to enhance the tracking and tracing capabilities of parts inside the supply chain. In the food industry, the introduction of IoT technologies is exploited to enhance food safety and traceability: QR code technology is proposed in ref. [26] to trace the logistic steps of vegetables; RFID anchors are instead proposed to enhance the traceability of the fish supply chain in ref. [27]. An RFID-enabled supply chain is also explored in ref. [28], where a theoretical model is proposed and evaluated on an aerospace manufacturing process. Those technologies provide promising results regarding the tracking and traceability requirements of physical parts but have a common weakness regarding the authenticity of the hyperlink: both technologies suffer from a clone vulnerability that makes the introduction of counterfeit parts in the supply chain possible.

RFID tags support hash operations, symmetric encryption, or Physically Unclonable Functions (PUFs) [29]. This is the case of the secure protocol Cipurse [30], which uses symmetric cryptography (often Advanced Encryption Standard, AES [31]) to authenticate a tag. Even if this protocol can be applied to NFC [32], it focuses on the secure use of symmetric cryptography inside the tag to keep the tag's secret protected from the majority of attacks. However, the authenticator needs complete or partial knowledge of this secret to perform the authentication, which gives the authenticator the possibility of creating falsifications. The secure management of this secret is a critical point of the symmetric cryptography, which is out of the scope of Cipurse.

There are many works in the state-of-the-art proposing solutions to this problem. In accordance with the survey in ref. [14], almost all the proposed solutions require relying on a centralized entity to identify the clones. This would allow this entity to record fake data about an authentication (tampering attack) or to clone a tag (spoofing attack). The only author in this survey that supports a decentralized solution is Elkhiyaoui [15]. This system uses read/write tags without hardware protection, i.e., everyone can read/write. Every tag contains the ID of the product and the cryptographically protected path of this ID. This solution cannot protect against cloning but proposes a method to detect it. In this method, all partners in the supply chain check between them that the same ID is not used in the two partners' stores. This solution is difficult to apply because it requires not only the cooperation of all the partners in the supply chain but also in the market, e.g., a counterfeit with a cloned ID could be sold to an assembler, and the partner would never detect it if they do not compare their database with the distributor from

a different supply chain line that has the original product. Also, another problem is that even if the path in the tag cannot be modified, it can be restored to an old version. So, when a part is in a non-auditable state (i.e., in final user control or discontinued), the distributor can clone its ID and its old path information, include it in the tag of a counterfeit, and sell it like a new and legitimate part.

Apart from that, another more recent work from Saikat M. et al. [33] presents a solution with a self-made blockchain to protect product-related information. In this way, the asset's path information cannot be modified. When a stakeholder receives an asset, it reads from the tag the secret "RFID address" and uses it to identify in the blockchain the product-related information. This secret is also used to send valid transactions to the blockchain and includes more product-related information. The problem with the solution is that there are not any mechanisms used to avoid or detect clones of the RFID tags. Therefore, any stakeholder or an attacker who could scan the part can clone the tag and sell a counterfeit.

Michail S. et al. [17] proposes a mechanism where only authorized entities can identify the tags. Once it is identified, a new block is uploaded to the blockchain, the tag's identification is modified, and the hash of the new identification is uploaded to the blockchain. At the end, the final user can identify the tag using the blockchain and trace its origin and path back. However, anyone with access to the tag can extract the secret credentials and clone it, not only the only authorized entities, because the tag answer is only protected with a random bitwise rotation between 0 and 96.

Nevertheless, the problem of Michail S. et al. [17] is resolved in the solution presented by Srinivas J. et al., LBRAPS [16]. Their protocol achieved mutual authentication and also the establishment of a session key between the tags (Ts) and the supply chain node (S). The work is quite novel because the tags themselves can authenticate S and the reader (R). Additionally, the solution is secure against diverse attacks like replay attacks. To achieve this, the tag stores its secret IDt, unique for each tag and the identifications of the reader and of the supply chain node, IDr and IDs, respectively. Also, the tag compares each received message's timestamp with the current time to avoid a replay attack. The solution is presented in the scenario of a single organization (department), a single reader, a single supply chain node, and a single tag. Finally, the authors claim that LBRAPS can also be applied in a distributed system, i.e., for various departments, similar to Michail S. et al.'s protocol [17] without giving further details. However, LBRAPS has key differences with Michail S. et al.'s protocol [17] that make it not applicable in the same way in distributed systems. LBRAPS requires the tags to know the IDs and IDr of all the S and R in the system, which is unfeasible and risky for a reduced memory tag. Also, in LBRAPS, the readers need to know the secret IDt of all tags that they are going to communicate with

beforehand, which requires a delicate system to distribute these sensible data. These two facts make the solution of Srinivas J. [16] hardly applicable to distributed systems.

On the other side, standard QR codes provide virtually no protection against copying. Efforts have been made to enhance these structures with anti-cloning features such as adding digital watermarks in QR code [34] or including copy detection patterns [35], but all these techniques present an accuracy that varies highly based on printing and scanning calibration, which makes them hard to implement in a real-world scenario. Another possible solution to enhance the link between a physical object and its cyber representation is the usage of physical characteristics that can uniquely identify an item. For example, DustIdentity [36], which proposes a Diamond Unclonable Security Tag, is a coating made out of diamond nanocrystals that can be registered as a unique fingerprint and thus used as an item identifier. This type of solution provides strong identification guarantees but lacks embedded computational capabilities to enhance the supply chain security.

Our research presents a different technological approach to this problem, creating an easy-to-operate and easy-to-implement solution that avoids the need to trust stakeholders or external parties for authentication. In our approach, the tags use high-performance asymmetric cryptography ECDSA, which avoids the possibility of cloning the assets. Also, every ownership transference of the asset is not just recorded in the blockchain but also the blockchain itself verifies the package and the new owner before performing the digital change in ownership, which detects any misuse. This provides complete transparency and highly increases trust in blockchain information. Finally, we implement our solution in a real privacy-preserving blockchain that avoids stakeholders accessing information that is not relevant to them.

4.4. Background

In this section, some important terms already mentioned previously are going to be described in detail, with the intention of going deeper into the proposal of this work.

- **Blockchain:** blockchain is a peer-to-peer infrastructure proposed by Satoshi Nakamoto in 2008 with the name Bitcoin [37]. The peers share a common database that can be extended by adding new blocks containing signed information to the chain, making it perfect for use as a ledger. Six years later, a new blockchain technology was released, called Ethereum [38]. This blockchain technology can be used not only to store bank balances but also to host scripts that can be executed and used to manage data in the system. These scripts are known as smart contracts. Blockchain possesses inherent characteristics that render it

exceptionally suitable for certain security applications: All information uploaded—transactions—passes through a consensus mechanism before being accepted in the system; the information, once uploaded, cannot be removed or modified; and all new participants can verify the authenticity of all previously recorded data. This information is publicly available in public blockchains, where everyone can access the infrastructure. However, information confidentiality can be guaranteed in permissioned blockchains where access to data is limited to authorized organizations. For example, HyperledgerFabric [39] is a framework for permissioned blockchain where data confidentiality can be protected through a certificate-based access control system.

- **Smart contract:** the smart contracts can be seen as stored procedures that regulate the operations on a blockchain system. All the transactions addressed to a smart contract must respect the rules embedded in the smart contract scripts. The compliance is validated through a consensus algorithm by the peers of the blockchain infrastructure, ensuring reliability, accountability, and availability [40]. These scripts are stored in the blockchain itself, guaranteeing code immutability. Mahd Miraz defines it like a trust machine [41]. All the functions processed are signed and stored in the blockchain, providing strong traceability.

- **Hardware Security Modules (HSMs):** HSMs are hardware-based solutions with security-by-design that can perform the essential cryptographic operations. These modules can perform the secure generation, secure storage, and secure operation of asymmetric cryptography (e.g., RSA, ECC). By design, there is no practical way to extract its secret crypto material; therefore, they can only be used to perform crypto-operations (e.g., digital signatures) by accessing its secure APIs.

- **Near-Field Communication (NFC):** NFC is a ISO standard [42] that defines a wireless communication technology working in the 13.56 MHz band. RFID and NFC are two different topics confused in the state of the art. Some papers consider NFC as a standard inside RFID [43] while others consider it as a completely independent standard [44, 45]. Nevertheless, the clear advantages of NFC with respect to other wireless technologies is its robustness against eavesdropping due to its very short reading range [45] of up to 10 cm [44] and its popularity, which makes the majority of today's mobile phones support NFC.

4.5. Proposed Solution

This research exploits the capabilities of passive NFC devices with embedded HSMs that are commonly available in the market (e.g., smart cards). Further on, this class of devices will be identified as a Secure Element (SE). The SE is used to assign a robust digital identity to parts tracked in the SCM. The aim is to design the physical integration of an SE into the overall part, effectively transforming it into a “smart-part”.

The SEs are protected with physical security measures that offer tamper protection and detection. For example, the SEs can be protected with a protective mesh and covered with security tape [46]. Moreover, being a secure hardware element designed, produced, and tested by a reliable company following the standard Common Criteria Evaluation Assurance Levels 5+ [47, 48], the risk of software attacks or side-channel attacks is reduced significantly.

The SE contains, in a secure partition, a private key that can be used to perform asymmetric crypto operations such as a digital signature. All the operations are performed through secure-by-design APIs, always keeping the crypto material in a secure state.

Once a part is physically tagged with the hardware-based digital identity, the crypto functionalities of the SE can be used to interact with the blockchain-based SCM system. The smart-part can send valid transactions to the blockchain, signing them with its specific private key. The crypto information required for the signature cannot be extracted from the SE; this means that only someone with physical access to the part can trigger these blockchain functionalities, which proves package ownership.

4.5.1. Integration of SE in the SCM

To demonstrate the integration of the solution, a distributed-ledger infrastructure for supply chain management is properly deployed. The infrastructure used is taken from [49] and the details are out of the scope of this research. In this context, the focus is on a common SCM functionality that provides ownership tracking capabilities of a given part inside the supply chain. From a high-level point of view, the SCM stores the current owner and exposes services to record ownership changes to follow the part inside the manufacturing process from the suppliers to the final customer. As baseline, an SE is provided with strong private keys, the manufacturing certificates, and a certificate showing part validity (which can be used as oracle certificates [50]).

The SCM is coordinated by a smart contract, which is validated by all the stakeholders and stored in a permissioned blockchain.

Figure 4.1 presents a functionality scheme of the interactions between the stakeholders, the SE, and our blockchain SCM system. Notice that the yellow diamonds shall be successful before passing to the next steps, delivering robust

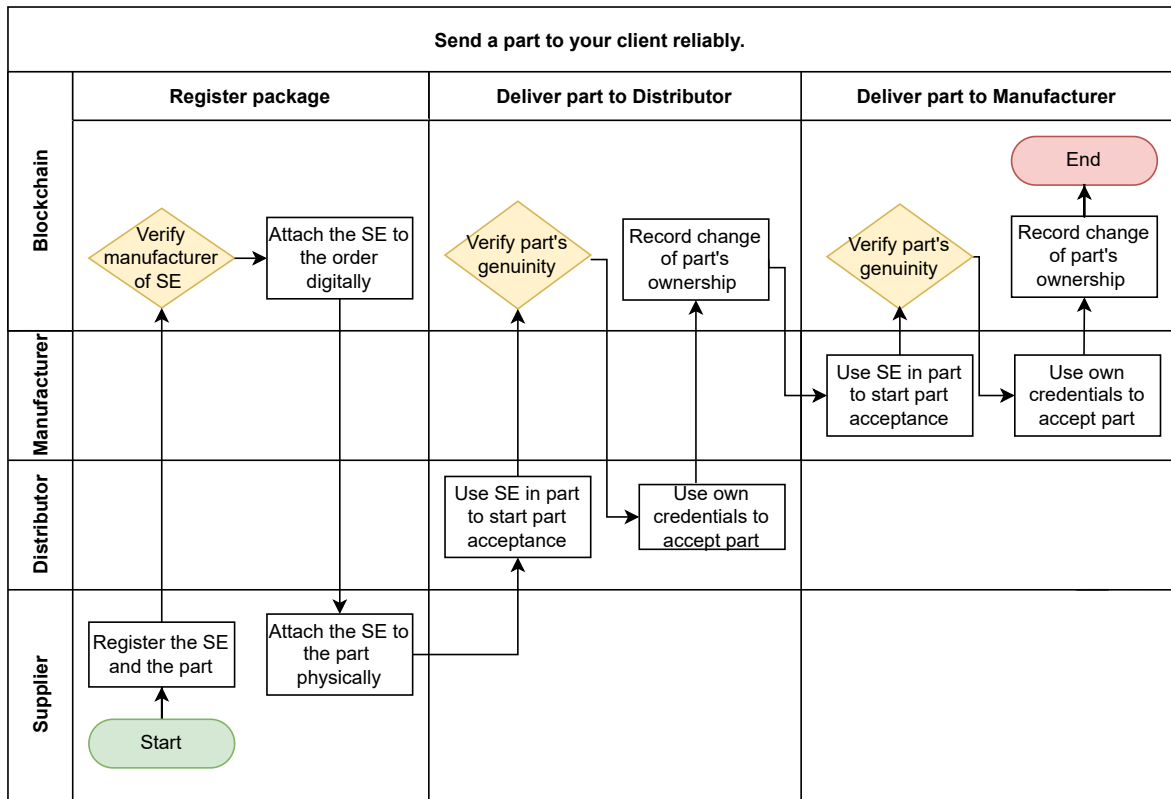


Figure 4.1: Overview of the proposed interactions.

tracking capabilities and consistent information. Firstly, the supplier receives and accepts an order for a specific part. Then, the part manufacturing and shipping starts:

- (A) A supplier physically packages the part to be shipped and prepares it to be traced in the SCM by performing the following:
- The SE is enrolled to the blockchain, registering its identity, the SE's manufacturer certificates, and the physical part that it is linked to.
 - The smart contract verifies the certificates to assert the SE trustworthiness.
 - After verifying the certificates, the SCM approves the use of this SE as a part tag, stores its public key, and attaches it to the order.
 - The SE is finally attached physically to the part, and the tampering security controls are deployed.
- (B) The supplier physically and digitally delivers the part to the next actor in the supply chain, the distributor. The digital change in ownership

requires the use of two random numbers (“Challenge” and “Challenge-proposed”), as explained in detail in Section 4.5.2, to avoid prediction attacks and delay attacks defined in Section 4.7.2.

- The distributor uses the SE attached to the received part to sign a transaction for the blockchain.
 - The blockchain verifies that the SE used to sign the transaction is authentic and, consequently, the delivery part.
 - The distributor uses their own credentials to send a transaction to the blockchain accepting the part.
 - Finally, the smart contract accepts and records the new part owner.
- (C) The change in ownership is repeated successively between the supply chain partners until the smart part reaches the final customer.

The scheme shows that only when the distributed ledger verifies the part can it be accepted, and the change in ownership can be recorded permanently in the blockchain. With this system, all the important steps of the product life cycle are verified by consensus and then recorded. This guarantees a correct verification and of the recorded data and that it has not been tampered with. Thus, this approach enables any stakeholder, like the final customer or any other authorized third party, with read access to the smart contract to accurately monitor in real-time the status and the history of a part in the supply chain.

4.5.2. Change Ownership Detailed

The smart-part has its identity properly enrolled in the blockchain, so its SE can be used to validate transactions in the system issued by anyone with NFC access to the part. For this reason, the physical owner has to confirm all the transactions performed by the smart part with their own identity. The ownership change of a smart part is the following: the receiving organization employs the SE to propose the ownership change that must be confirmed through a second transaction in the SCM. Therefore, an ownership change is considered valid if (1) it is signed with the part secret key, providing proof of the physical presence of the specific part, and (2) it is confirmed by the receiving organization, providing non-repudiation guarantees on the future owner of the part.

Here follows a detailed explanation of the two-step transaction, which is illustrated in Figure 4.2. Each part has the following security variables associated:

- Owner: represents the organization in the supply chain currently holding the part.

- Owner-proposed: represents the partner that has to confirm with a second transaction to become the new owner of the part.
- Challenge: random number.
- Challenge-proposed: next random challenge number.

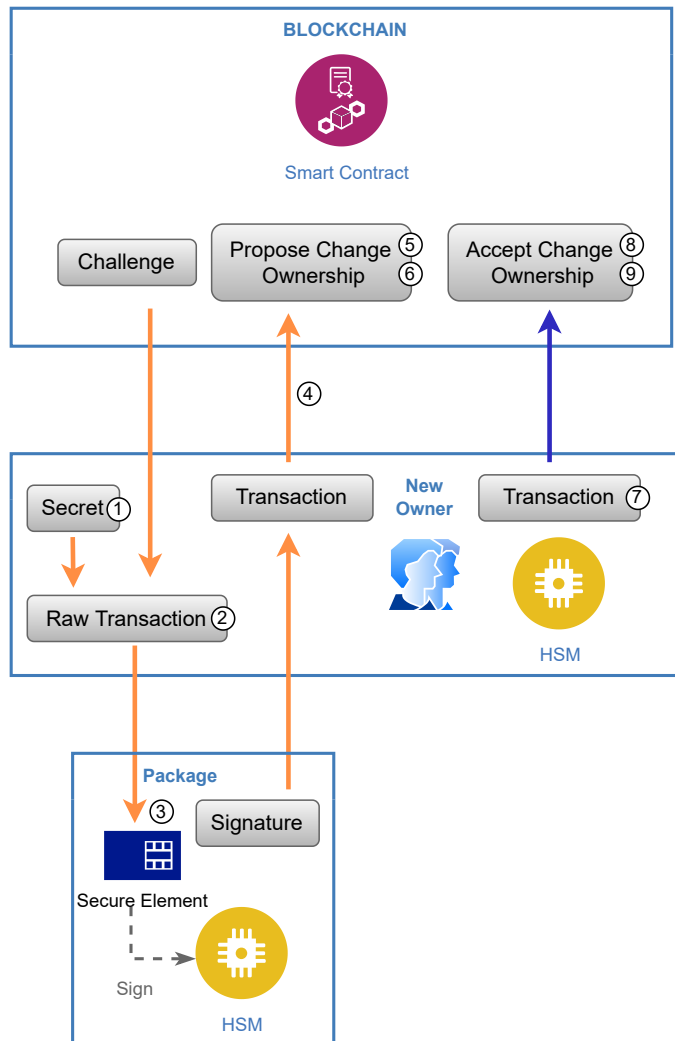


Figure 4.2: Two-step transaction infrastructure used to change the ownership of a package in the smart contract in the blockchain. The numbers in the figure refer to the enumerated steps in the text.

These variables are required to ensure a secure ownership exchange of the part between authorized SCM organizations, avoiding prediction attacks and delay attacks defined in Section 4.7.2. When a new actor wants to

acquire ownership of a part in the supply chain, they proceed by proposing an ownership change as follows (orange arrows in Figure 4.2):

1. The new owner generates a new secret random number (Secret) to be used as “Challenge-proposed”.
2. The actor prepares a non-signed transaction (raw transaction), proposing a change in ownership, ProposedOwnershipChange. The raw transaction contains the part ID, the future owner, the current challenge “Challenge”, and the secret random number as the proposed challenge.
3. The raw transaction is sent to the part through NFC and is signed by the SE, which requires physical access to the smart part.
4. The signed transaction is published on the blockchain. The signature confirms that this action was prompted by the current owner of the part. Another consequence of this is that the proposed challenge is publicly disclosed and is not secret any more.
5. The blockchain system validates the transaction by checking (i) that the current challenge number corresponds to the last challenge stored in the SCM for that specific part, “Challenge”, and (ii) that the part signature is authentic.
6. Once the transaction is validated, the security variables “Owner-proposed” and “Challenge-proposed” are updated.

At this point, any supply chain partner with access to the part could have submitted this transaction. Therefore, the changes produced in the system so far are not binding and must be validated by the receiving organization to have guarantees of robustness and non-repudiation in ownership change. Consequently, the next actor in the supply chain, which is indicated as “owner-proposed” in the security variables, accepts the ownership change with the following steps (blue arrow in Figure 4.2):

7. The new owner creates a blockchain transaction, AcceptOwnershipChange, attaching the part ID and newly published value currently labeled as “Challenge-proposed”.
8. The blockchain verifies (i) that the triggerer is a member of the organization currently stored as “Owner-proposed” and (ii) that the “Challenge” value in the transaction corresponds to what is stored as “Challenge-proposed”.
9. Once the transaction is accepted, the security variables are updated to store the proposed values as Owner and Challenge.

The steps presented to propose and accept an ownership are repeated every time the part is sent/received by a partner in the supply chain. With this chain of events, the system can guarantee proper traceability of the part and non-repudiation of the physical ownership by the supply partners.

4.6. Implementation

The proposed solution was implemented using the permission blockchain framework Hyperledger Fabric (HPL) [39]. Using this framework, we created a blockchain-based SCM system with enhanced access control providing strong confidentiality, data traceability, and non-repudiation guarantees.

In this implementation, we consider four stakeholders: supplier, distribution, manufacturer, and safety certification authority. These entities interact with the SCM using a smart contract with their validated credentials as defined in the permission blockchain HPL. Only using the validated credentials can the information in the blockchain be accessed, or can new information be uploaded, obtaining privacy with respect to the external users.

The smart contract implements the following functions:

- **Orderpart**: the manufacturer executes Orderpart by sending a transaction to the blockchain. It creates the data structures Contract, Objective&Compliance, and PriceAgreement, which contain the information about the requested order, e.g., qualification data or price.
- **AcceptContract**: the supplier accepts the order executing AcceptContract, including in the function the Contract identifier.
- **SendPart**: the supplier executes SendPart to register the SE's public key and link it with the ordered part.
- **AcceptDeliveryContract**: a distributor with access to the blockchain can accept delivering this order by executing the function AcceptDeliveryContract and providing the Contract identifier.
- **ProposedOwnershipChange** and **AcceptOwnershipChange**: these functions are triggered in order to execute a secure ownership change. Their functionality is detailed in Section 4.5.2.

To avoid the elevation of privileges to modify smart contract data, assertions are placed in each smart contract function to confirm that the triggering user is authenticated with the validated credentials and is part of an organization authorized to execute that specific action. For example, in our scenario, only the manufacturer is allowed to execute Orderpart to order a part from the supplier. If any other organization attempts to trigger the

function Orderpart, the validation process of the peers will detect an assertion failure and reject it.

Even if actions identified by the smart contracts are strongly regulated, HLF exposes a query engine, which allows any participant in the blockchain to read the state of the ledger. This action is always permitted and is not regulated by the smart contract. Therefore, in order to enforce the least knowledge principle and to maintain data confidentiality, HLF proposes the Private Data Collections (PDCs) functionality to regulate data read access. Firstly, we identify three data structures with different access rights as shown in Table 4.2. Then, as shown in Figure 4.3, the Contract structure is stored directly in the blockchain, and therefore the four stakeholders can read it. Instead, Objective&Compliance and PriceAgreement structures are shared using Private Data Collections (PDCs), which send data peer-to-peer via gossip protocol to only the partner(s) authorized to access it. This information is stored in a private database on the peers of the authorized organizations, and it can be read only through smart contract functions. A hash value of each PDC entry is written on the ledger state as proof of existence and can be inspected by every participant on the blockchain. The hash serves as evidence of the transaction, is used for state validation, and can be used for audit purposes [51]. Notice in Figure 4.3 that one data structure is openly accessed by every participant in the system (yellow line) while the other data containers have restricted access (orange line).

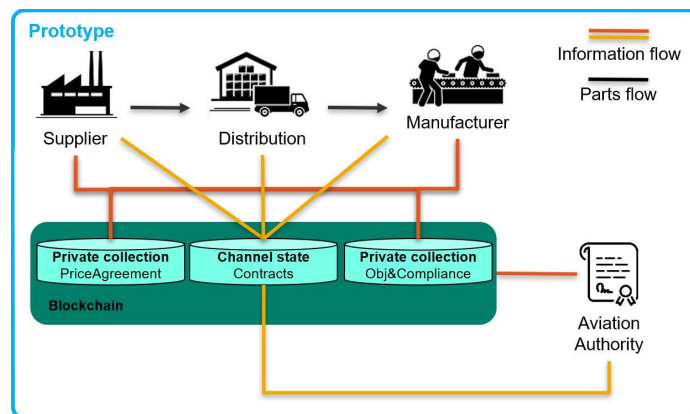


Figure 4.3: Secure architecture of a modern blockchain-based SCM applied to manufacturing use case.

Table 4.2: Data assets and protection policies.

Structure	Description	Access Rights
Contracts	Agreement between the manufacturer and the supplier. It contains common data such as name and number of the components, expected delivery date, and all the logistic details	Supplier(R/W), Distribution(R/W), Manufacturer(R/W), Aviation Authority(R)
Obj&Comp.	Each component is linked to objectives or requirements expressed by the manufacturer. The same data structure contains the compliance measurements that the supplier provides to prove adherence to the requirements. Requirements and quality measurements are considered sensitive data since they contain intellectual property and related information.	Supplier(R/W), Manufacturer(R/W), Aviation Authority(R)
PriceAgr.	Each contract has an economical agreement linked to it.	Supplier(R/W), Manufacturer(R/W)

Finally, to avoid a spoofing attack on the part's identity, it is important to link it to the public key on the SE immovably. In public blockchains, e.g., Bitcoin or Ethereum, the identity of a transaction sender is cryptographically generated by its public key; therefore, each identity is linked with a unique private key, and it is immovable. However, in Hyperledger Fabric, the identity is linked with the public key through a certificate, and, in the case of the part, it is issued by the supplier. Then, the supplier could also issue a second certificate for a part's identity with another SE, which would allow for a package clonation. To avoid this threat, when a part is registered in the SCM, the SE's public key is also stored in the smart contract. Then, in every change in ownership, the identification mechanism of Hyperledger Fabric has to be modified to assert that the public key of the SE is the same as the stored public key as shown in Figure 4.4.

With this solution, we have created a secure and fully functional smart contract that distinguishes between the privileges of users when accepting transactions and delivering information to the stakeholders. Using this same mechanism, other private collections can be created for other uses and more complex data access policies can be defined if necessary. At the same time, this mechanism maintains blockchain's reliability in the private collections since the hash of the private data is stored in the blockchain consortium.

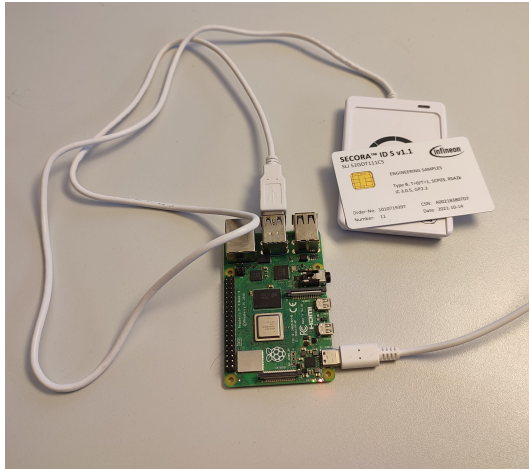


Figure 4.5: Prototype setup for laboratory evaluation.

```

publicKeyDer, _ := x509.MarshalPKIXPublicKey(certificate.PublicKey) //Get public key from smart card certificate
publicKeyDerHex := hex.EncodeToString(publicKeyDer)
publicKeyUncompressedFormat := publicKeyDerHex[52:len(publicKeyDerHex)]
fmt.Println("Public key from transaction signature requestor: %s", publicKeyUncompressedFormat)
if (delivery.PackagePublicKey != publicKeyUncompressedFormat){
    return fmt.Errorf("Package is not original (public key not matching)")
}

```

Figure 4.4: New lines used to identify the parts using their public key instead of their IDs.

To test and validate the capabilities offered by this approach, we used an actual HSM with NFC, and NFC readers, which interact with the smart contract deployed in a privacy-preserving blockchain. To facilitate this process, each participating organization deployed an NFC reader comprising an ACR122U NFC Reader and a Raspberry Pi 4B, as illustrated in the setup in Figure 4.5. To execute the test, we used a configurable smart-card, SecoraTM ID S V1.1 [48], attached to the parts. This smart card possessed the necessary secure functionalities required for our proposed solutions, bringing us closer to market implementation.

Performance Evaluation

The prototype implementation was tested to measure the performances of the SE operations in the proposed architecture. A specific NodeJS [52] client was created to interact with the SE. This client generates the unsigned transaction, hashes it, and sends it to the smart card to be signed. Then, a blockchain verifies the transactions, confirming physical ownership of the part. Due to the internal functionality of Hyperledger Fabric, when sending

a transaction, a second signature is needed. The execution of both actions has been tested and measured from a performance point of view on 20 iterations. Table 4.3 presents the measurements showing the time required for each operation and the overall transaction submission time. Notice that the worst case scenario overhead introduced by adding the two crypto-operations performed by the SE into the Hyperledger client transaction is less than 0.6 seconds (accounting of 17.1 % of the total procedure time).

The security/performance trade-offs introduced by this architecture are considered acceptable for this use case, where security is a big concern and there is no real-time requirement, considering the use case where humans are interacting and managing a physical part. Additionally, this process removes the necessity of the handwritten signatures in the supply chain, which saves time and enhance consistency. Finally, it is important to notice that not only do the two parties agree to the part transference, but all the authorized partners are remotely verifying the process and the part's genuineness.

Table 4.3: HSM performances while interrogated through NodeJS API.

Action	Avg. Time	Min. Time	Max. Time
HSM Sign 1	267.5 ms	264 ms	281 ms
HSM Sign 2	267.7 ms	263 ms	281 ms
Sum of Sign Actions	535.2 ms	527 ms	562 ms
Overall Transaction Time	3290.1 ms	3262 ms	3307 ms
HSM Overhead	535.2 ms	527 ms	562 ms
HSM Overhead (%)	16.3 %	16.0 %	17.1 %

4.7. Security Analysis

This section explains firstly how our solution can securely mitigate the threats exposed in Section 4.2. Secondly, it proves the robustness of the system against the attacks found in the state of the art and some new ones self-developed.

4.7.1. Threat Analysis

The proposed solution has been qualitatively evaluated against the risks identified in Section 4.2. A summary of the mitigation introduced by the solution with respect to the threats is highlighted in Table 4.4.

Table 4.4: Threat analysis.

Threat Type	Mitigation
Spoofting	The use of an HSM to generate and store the private keys of the parts ensures the impossibility of credentials spoofing. Furthermore, identification in the smart contract requires a public key of the SE for the correct functionality. These two facts together ensure the impossibility of cloning the parts.
Tampering	As a result of the proposed scheme, the ownership of a package can only be modified by someone with physical access to the package. This ensures that no false ownership information about the part can be added to the ledger. Also, the data are stored in the blockchain, which prevents any malicious manipulation of the data once uploaded.
Repudiation	All data uploaded to the smart contract are signed, and the decentralized system identifies every contributor before accepting the data. Finally, the peers of the blockchain themselves verify the presence of the real and unique part in the change in ownership process. This means that the information cannot be disputed once it is accepted in the blockchain.
Information disclosure	The solution is implemented over a permissioned blockchain. This means that no entity outside the supply chain can access the data.
Denial of Service	The SCM is based on blockchain with a decentralized peer-to-peer infrastructure. This architecture provides resilience by the design of the system against DoS attacks.
Elevation of Privileges	The solution uses privacy-preserving blockchain, taking advantage of the private collections of Hyperledger Fabric. With this solution, the stakeholders' part of the supply chain cannot freely access all the information, only those that are relevant to them. Also, the ownership change of the asset requires physical access to the part itself, and the information can never be deleted, which avoids any stakeholder using its privileges to insert fake data in the blockchain.
System misuse	In order to take digital ownership of the package, it is required to verify its authenticity. This operation forces the verification of the asset at every ownership change in the supply chain.

4.7.2. Attack Analysis

Given that the system mitigates the common threats identified, here is discussed the resilience provided by the proposed solution against the following known attacks.

4.7.2.1. Key Disclosure

Key Disclosure (KD) is an attack where an internal or outsider attacker can in some way extract the secret variables stored in the tag. The current protocol uses asymmetric cryptography, therefore, the private key is never exposed and always kept protected in the SE.

4.7.2.2. Replay Attack

In a replay attack, an internal or outsider attacker gathers a legitimate message of the system and replays it later to negatively affect the system. In Hyperledger Fabric, all transactions are protected from replay attacks by using a unique nonce [53]. This means that each transaction can be uploaded to the blockchain only once.

4.7.2.3. Man-in-the-Middle Attack

In a Man-In-The-Middle (MITM) attack, an outsider attacker intercepts the communications between two parties and relays and possibly alters the messages. Our protocol, unlike other protocols for supply chain tracking, uses NFC. This highly reduces the possibility of an MITM attack as the attacker should be in the 10 cm distance between the tag and the reader. Furthermore, a very unlikely successful MITM could modify the hash sent to the tag or the signature sent back by the SE. This attack would be detected in the part verification function.

4.7.2.4. Tracking Attack

An outsider attacker can track the part along the supply chain even if they do not belong to the organization based on the responses of the SE. In this proposed solution, the SE performs an ECDSA signature given an external hash. The ECDSA algorithm always gives back a different and unpredictable signature based on a random secret number K internally created by the signer and unique to each message. Therefore, even if an attacker with access to the SE always gives a constant hash to it, the attacker will always retrieve an unpredictable response and will be unable to track the SE.

4.7.2.5. Prediction Attack

In a prediction attack, an insider or outsider attacker completely or partially knows the future answers of the SE without a KD and, therefore, can create a temporal or a permanent clone of the SE. Most blockchain implementations use a predictable nonce in their transactions (e.g., Ethereum [54] and Hyperledger Fabric). This allows the prediction of the hash of the next transaction to be signed by the SE. With this information, an internal attacker

can interrogate the SE to obtain the signature of the next transaction and create a fake part that returns the predicted signature of the next transaction. The next actor in the supply chain would read from the fake part the correct signature and consider it real. This attack is prevented in the proposed protocol by using a secret random number in the “Challenge-proposed” field, which is generated by the next actor, to make the transaction unpredictable and thus the hash be signed by the SE.

4.7.2.6. Delay Attack

An internal attacker can use the SE when it is in its possession to create messages that will be sent later to negatively affect the system without a KD. The attacker could sign, using the SE, a transaction proposing an ownership change, including themselves as the next owner, but they do not publish it on the blockchain. In the future, without physical access to the part, they send the signed transaction and then proceed to confirm its ownership. This attack is impossible to apply in the proposed protocol since the transaction that changes the ownership requires the “Challenge” variable, which is a random, unrepeatable, and unpredictable value. This value is updated from “Challenge-proposed” every time the part’s owner changes, guaranteeing that ownership transactions are ordered and thus cannot be delayed.

4.7.3. Comparison

Table 4.5 compares the previously discussed solutions in the state of the art with the proposed protocol. In the table, X means that the protocol cannot provide protection against the attack or not support the property whereas \checkmark means the opposite. On the other hand, NA indicate that the attack or property is not applicable to the protocol. Notice that, in refs. [15, 33, 16], the internal partners require access to the keys inside the SE to identify it, and therefore are not resistant to KD. That means that, at any later moment, they could insert a clone in the supply chain, and they would very rarely be identified as guilty. The protocol [17] fails the key protection by relying on simple bitwise rotation to guarantee key confidentiality. Therefore, none of them are protected against KD. Additionally, none of the analyzed work presents a solution to avoid the stakeholder taking advantage of their privileges and accessing information that is not relevant to them stored in the distributed system.

4.8. Conclusions and Future Work

This paper presents a completely decentralized architecture for safety-critical industrial supply chain tracking based on a secure element and blockchain. The technology forces the honest behavior of the stakeholders and

Table 4.5: Comparison of decentralized-based SCMs.

Security Protection to:	Elkhiyaoui et al. [15]	Saikat et al. [33]	ULRMAPC [17]	LBRAPS [16]	Own
Key disclosure	X	X	X	X	✓
Replay attack	X	X	✓	✓	✓
MITM attack	✓	X	✓	✓	✓
Tracking attack	✓	X	✓	✓	✓
Prediction attack	X	NA	✓	✓	✓
Delay attack	NA	NA	X	✓	✓
Elevation of privileges	NA	X	X	X	✓
Decentralized	✓	✓	✓	X	✓

provides a strong guarantee of the quality of the product to final users or third-party observers. To achieve this, NFC tags with asymmetric cryptography capabilities are integrated in the parts manufactured and exchanged in the considered supply chain. With a tag, the part receives a digital identity inside the blockchain infrastructure and can trigger transactions. The blockchain-based supply chain manages all the parts information and cryptographically certifies its owner at every moment. Using a novel scheme, the smart contract forces the stakeholders to honestly verify the parts along any asset transference, and the blockchain itself is witness of the transfer process. The solution is implemented over the Hyperledger Fabric framework, and it uses the “private collection” structures to protect the individual privacy of each stakeholder. As next steps, other functionalities are envisioned, such as the ability to provide the part’s physical owner access to certain blockchain privileges or specific data. Additionally, following other tracking solutions, a global navigation satellite system sensor could be integrated in the tag to include information about the position signed by the HSM, though it would present challenges such as the need for an active tag with a power supply.

References

- [1] Abdirad, M.; Krishnan, K. Industry 4.0 in logistics and supply chain management: A systematic literature review. *Eng. Manag. J.* **2021**, *33*, 187–201. [CrossRef]
- [2] Bag, S.; Telukdarie, A.; Pretorius, J.C.; Gupta, S. Industry 4.0 and supply chain sustainability: Framework and future research directions. *Benchmarking Int. J.* **2021**, *28*, 1410–1450. [CrossRef]
- [3] Stadtler, H. Supply chain management: An overview. In *Supply Chain Management and Advanced Planning: Concepts, Models, Software, and Case Studies*; Springer: Berlin/Heidelberg, Germany, 2014; pp. 3–28.

- [4] OECD/EUIPO. Global Trade in Fakes: A Worrying Threat; In OECD-iLibrary. 2021. Available online: <https://www.oecd.org/publications/global-trade-in-fakes-74c81154-en.htm> (accessed on 31 July 2023).
- [5] Tiwari, S. Supply chain integration and Industry 4.0: A systematic literature review. *Benchmarking Int. J.* **2021**, *28*, 990–1030. [CrossRef]
- [6] Queiroz, M.M.; Pereira, S.C.F.; Telles, R.; Machado, M.C. Industry 4.0 and digital supply chain capabilities: A framework for understanding digitalisation challenges and opportunities. *Benchmarking Int. J.* **2021**, *28*, 1761–1782. [CrossRef]
- [7] Fatorachian, H.; Kazemi, H. Impact of Industry 4.0 on supply chain performance. *Prod. Plan. Control* **2021**, *32*, 63–81. [CrossRef]
- [8] Sunny, J.; Undralla, N.; Pillai, V.M. Supply chain transparency through blockchain-based traceability: An overview with demonstration. *Comput. Ind. Eng.* **2020**, *150*, 106895. [CrossRef]
- [9] Chang, S.E.; Chen, Y. When blockchain meets supply chain: A systematic literature review on current development and potential applications. *IEEE Access* **2020**, *8*, 62478–62494. [CrossRef]
- [10] Queiroz, M.M.; Telles, R.; Bonilla, S.H. Blockchain and supply chain management integration: A systematic review of the literature. *Supply Chain. Manag. Int. J.* **2019**, *25*, 241–254. [CrossRef]
- [11] Wüst, K.; Gervais, A. Do you need a blockchain? In Proceedings of the Crypto Valley Conference on Blockchain Technology (CVCBT), Zug, Switzerland, 20–22 June 2018; pp. 45–54.
- [12] Caldarelli, G. Understanding the blockchain oracle problem: A call for action. *Information* **2020**, *11*, 509. [CrossRef]
- [13] Aich, S.; Chakraborty, S.; Sain, M.; Lee, H.I.; Kim, H.C. A review on benefits of IoT integrated blockchain based supply chain management implementations across different sectors with case study. In Proceedings of the 21st International Conference on Advanced Communication Technology (ICACT), PyeongChang, Republic of Korea, 17–20 February 2019; pp. 138–141.
- [14] Bu, K.; Weng, M.; Zheng, Y.; Xiao, B.; Liu, X. You Can Clone But You Cannot Hide: A Survey of Clone Prevention and Detection for RFID. *IEEE Commun. Surv. Tutorials* **2017**, *19*, 1682–1700. [CrossRef]
- [15] Elkhyaoui, K.; Blass, E.O.; Molva, R. CHECKER: On-site checking in RFID-based supply chains. In Proceedings of the 5th ACM Conference

- on Security and Privacy in Wireless and Mobile Networks, Tucson, AZ, USA, 16–18 April 2012; pp. 173–184.
- [16] Jangirala, S.; Das, A.K.; Vasilakos, A.V. Designing secure lightweight blockchain-enabled RFID-based authentication protocol for supply chains in 5G mobile edge computing environment. *IEEE Trans. Ind. Inform.* **2019**, *16*, 7081–7093. [CrossRef]
- [17] Sidorov, M.; Ong, M.T.; Sridharan, R.V.; Nakamura, J.; Ohmura, R.; Khor, J.H. Ultralightweight Mutual Authentication RFID Protocol for Blockchain Enabled Supply Chains. *IEEE Access* **2019**, *7*, 7273–7285. [CrossRef]
- [18] Coskun, V.; Ozdenizci, B.; Ok, K. A survey on near field communication (NFC) technology. *Wirel. Pers. Commun.* **2013**, *71*, 2259–2294. [CrossRef]
- [19] Johnson, D.; Menezes, A.; Vanstone, S. The elliptic curve digital signature algorithm (ECDSA). *Int. J. Inf. Secur.* **2001**, *1*, 36–63. [CrossRef]
- [20] Dutta, P.; Choi, T.M.; Somani, S.; Butala, R. Blockchain technology in supply chain operations: Applications, challenges and research opportunities. *Transp. Res. Part E Logist. Transp. Rev.* **2020**, *142*, 102067. [CrossRef]
- [21] Meixell, M.J.; Gargeya, V.B. Global supply chain design: A literature review and critique. *Transp. Res. Part E Logist. Transp. Rev.* **2005**, *41*, 531–550. [CrossRef]
- [22] Hui, X.; Li, K.; Wang, C.; Zhang, C.; Gu, Z. Risk Management of Aerospace Stand-Alone Device Supply Chain. In Proceedings of the International Conference on Industrial IoT, Big Data and Supply Chain (IIoTBDSC), Beijing, China, 23–25 September 2022; pp. 272–277. [CrossRef]
- [23] Lanotte, H.; Ferreira, A.; Brisset, P. Lean supply chain and designing a customer-oriented dashboard: The case of an aerospace company. In Proceedings of the IEEE 13th International Colloquium of Logistics and Supply Chain Management (LOGISTIQUA), Fez, Morocco, 2–4 December 2020; pp. 1–7. [CrossRef]
- [24] Howard, M.; Lipner, S. *The Security Development Lifecycle*; Microsoft Press: Redmond, WA, USA, 2006; Volume 8.
- [25] Birkel, H.S.; Hartmann, E. Impact of IoT challenges and risks for SCM. *Supply Chain. Manag. Int. J.* **2019**, *24*, 39–61. [CrossRef]

- [26] Gao, H. Study on the Application of the QRcode Technology in the Farm Product Supply Chain Traceability System. *Appl. Mech. Mater.* **2013**, 321–324, 3056–3060. [CrossRef]
- [27] Hsu, Y.C.; Chen, A.P.; Wang, C.H. A RFID-enabled traceability system for the supply chain of live fish. In Proceedings of the IEEE International Conference on Automation and Logistics, Qingdao, China, 1–3 September 2008; pp. 81–86. [CrossRef]
- [28] Harun, K.; Cheng, K.; Wibbelmann, M. RFID-enabled aerospace manufacturing: Theoretical models, simulation and implementation issues. In Proceedings of the IEEE International Conference on Industrial Engineering and Engineering Management, Singapore, 8–11 December 2008; pp. 1824–1829. [CrossRef]
- [29] Gassend, B.; Clarke, D.; Van Dijk, M.; Devadas, S. Silicon physical random functions. In Proceedings of the 9th ACM Conference on Computer and Communications Security, Washington, DC, USA, 18–22 November 2002; pp. 148–160.
- [30] OSTP Alliance™ Cipurse™ v2 Cryptographic Protocol Revision 2.0. Available online: https://www.cardlogix.com/downloads/support/CIPURSE_V2_Revision_2_0_Document_Overview.pdf (accessed on 31 July 2023).
- [31] Dworkin, M.; Barker, E.; Nechvatal, J.; Foti, J.; Bassham, L.; Roback, E.; Dray, J. Advanced Encryption Standard (AES). 2001. Available online: <https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.197.pdf> (accessed on 31 July 2023).
- [32] Madlmayr, G.; Langer, J.; Kantner, C.; Scharinger, J. NFC Devices: Security and Privacy. In Proceedings of the 3rd International Conference on Availability, Reliability and Security, Washington, DC, USA, 4–7 March 2008; pp. 642–647. [CrossRef]
- [33] Mondal, S.; Wijewardena, K.P.; Karuppuswami, S.; Kriti, N.; Kumar, D.; Chahal, P. Blockchain Inspired RFID-Based Information Architecture for Food Supply Chain. *IEEE Internet Things J.* **2019**, 6, 5803–5813. [CrossRef]
- [34] Biro, A.; Kristo, G.; Remenyi, P. Security Element and Method to Inspect Authenticity of a Print. European Patent EP2815567B1, 14 March 2017.
- [35] Picard, J.; Landry, P.; Bolay, M. Counterfeit Detection with QR Codes. In Proceedings of the 21st ACM Symposium on Document Engineering, Limerick, Ireland, 24–27 August 2021.

- [36] Dust Identity. Available online: <https://dustidentity.com/products> (accessed on 25 July 2022).
- [37] Nakamoto, S. Bitcoin: A Peer-to-Peer Electronic Cash System. 2008. Available online: <https://bitcoin.org/bitcoin.pdf> (accessed on 31 July 2023).
- [38] Ethereum, W. Ethereum Whitepaper. Ethereum. 2014. Available online: <https://ethereum.org> (accessed on 7 July 2020).
- [39] Androulaki, E.; Barger, A.; Bortnikov, V.; Cachin, C.; Christidis, K.; De Caro, A.; Enyeart, D.; Ferris, C.; Laventman, G.; Manevich, Y.; et al. Hyperledger fabric: A distributed operating system for permissioned blockchains. In Proceedings of the 13th EuroSys Conference, Porto, Portugal, 23–26 April 2018; pp. 1–15.
- [40] Gupta, R.; Tanwar, S.; Kumar, N.; Tyagi, S. Blockchain-based security attack resilience schemes for autonomous vehicles in industry 4.0: A systematic review. *Comput. Electr. Eng.* **2020**, *86*, 106717. [CrossRef]
- [41] Miraz, M.H. Blockchain of things (BCoT): The fusion of blockchain and IoT technologies. In *Advanced Applications of Blockchain Technology*; Springer: Berlin/Heidelberg, Germany, 2020; pp. 141–159.
- [42] *ISO/IEC 18092:2013*; ISO/IEC JTC 1/SC 6 Telecommunications and Information Exchange between Systems. ISO: Geneva, Switzerland, 2013.
- [43] Juels, A. RFID security and privacy: A research survey. *IEEE J. Sel. Areas Commun.* **2006**, *24*, 381–394. [CrossRef]
- [44] Vazquez-Briseno, M.; Hirata, F.I.; Sanchez-Lopez, J.; Jimenez-Garcia, E.; Navarro-Cota, C.; Nieto-Hipolito, J.I. Using RFID/NFC and QR-code in mobile phones to link the physical and the digital world. *Interact. Multimed.* **2012**, *12*, 219–242.
- [45] Lahtela, A.; Hassinen, M.; Jylha, V. RFID and NFC in healthcare: Safety of hospitals medication care. In Proceedings of the 2nd International Conference on Pervasive Computing Technologies for Healthcare, Tampere, Finland, 30 January–1 February 2008; pp. 241–244. [CrossRef]
- [46] Anderson, R. *Security Engineering: A Guide to Building Dependable Distributed Systems*; John Wiley & Sons: Hoboken, NJ, USA, 2020; Chapter 16.
- [47] *ISO/IEC 15408-5:2022*; Information Security, Cybersecurity and Privacy Protection—Evaluation Criteria for IT Security—Part 5: Pre-Defined Packages of Security Requirements. ISO: Geneva, Switzerland, 2022.

- [48] Technologies, I. SECORA™ ID Security Solutions. Available online: <https://www.infineon.com/cms/en/product/security-smart-card-solutions/secora-security-solutions/secora-id-security-solutions/> (accessed on 18 August 2022).
- [49] Martintoni, D.; Senni, V.; Gomez Marin, E.; Cabrera Gutierrez, A.J. Sensitive information protection in blockchain-based supply-chain management for aerospace. In Proceedings of the IEEE International Conference on Omni-Layer Intelligent Systems (COINS), Barcelona, Spain, 1–3 August 2022; pp. 1–8.
- [50] Miličević, K.; Omrčen, L.; Kohler, M.; Lukić, I. Trust model concept for IoT blockchain applications as part of the digital transformation of metrology. *Sensors* **2022**, *22*, 4708. [CrossRef] [PubMed]
- [51] Foundation, L. HLF Private Collection. 2022. Available online: <https://hyperledger-fabric.readthedocs.io/en/release-2.2/private-data/private-data.html> (accessed on 29 June 2023).
- [52] Node.js®. Available online: <https://nodejs.org/en/> (accessed on 11 August 2022).
- [53] Foundation, H. Hyperledger Protocol Specification. Available online: <https://hlf.readthedocs.io/en/v0.6/protocol-spec.html> (accessed on 31 July 2023).
- [54] Foundation, E. Transactions. Available online: <https://ethereum.org/en/developers/docs/transactions/> (accessed on 31 July 2023).
- [55] COLLABS-871518 Project Website. Available online: <https://www.collabs-project.eu/> (accessed on 31 July 2023).

Chapter 5

Fine-Grained Access Control with User Revocation in Smart Manufacturing

Ernesto Gómez-Marín ¹, Davide Martintoni ², Valerio Senni ²,
Encarnación Castillo ³ and Luis Parrilla ³

1. Infineon Technologies AG, Am Campeon 1-15, 85579 Neubiberg,
Germany

2. Applied Research & Technology, Collins Aerospace, 00185 Rome,
Italy

3. Departamento de Electrónica y Tecnología de Computadores,
Universidad de Granada, 18071 Granada, Spain

Electronics

- Received: 19 May 2023, Revised: 20 June 2023, Accepted: 23 June 2023,
Published: 27 June 2023
- 10.3390/electronics12132843
- Impact factor: 2.9
- JCR Rank: 131/275 in category Engineering, Electrical & Electronic
(Q2)

Abstract: Collaborative manufacturing is a key enabler of Industry 4.0 that requires secure data sharing among multiple parties. However, intercompany data-sharing raises important privacy and security concerns, particularly given intellectual property and business-sensitive information collected by many devices. In this paper, we propose a solution that combines four technologies to address these challenges: Attribute-Based Encryption for data access control, blockchain for data integrity and non-repudiation, Hardware Security Modules for authenticity, and the Interplanetary File System for data scalability. We also use OpenID for dynamic client identification and propose a new method for user revocation in Attribute-Based Encryption. Our evaluation shows that the solution can scale up to 2,000,000 clients while maintaining all security guarantees.

Keywords: Industrial Internet of Things; access control; blockchain; attribute-based encryption; revocation; data-sharing; Industry 4.0.

5.1. Introduction

The third industrial revolution was a paradigm shift due to the inclusion of computers and programmable elements in the industry. However, the fourth industrial revolution brings a new manufacturing change—this time not by the inclusion of new machines, but by the advancement of information and communication of the machines and the autonomous manufacturing [1, 2]. One of the fundamental elements of this revolution is precisely the machine-to-machine interaction, or rather, thing-to-thing through the internet, with the aim of sharing information and self-organizing to face the changes in the environment. This is called the Internet of Things (IoT) [3].

IoT is currently being used in the industry for a large variety of use cases. For this reason it has received its own name—Industrial IoT (IIoT). Some of the applications are: Predictive maintenance by gathering information from the machinery, such as temperature or vibration, optimizing the supply chain with asset tracking by sending the asset location in real-time, inventory management by monitoring the inventory in real-time, quality control by detecting defects in the product or failures in the manufacturing process, etc. [4]. Furthermore, IIoT is growing rapidly. A recent study by The Business Research Company claims a growth from USD 209 billion in 2022 to USD 252 billion in 2023, and they also forecast that IoT in the manufacturing market will grow to more than USD 400 billion in 2027 [5].

For all those use-cases, data has to be shared not only with machines in the same factory level but with individuals from different levels and even with other stakeholders, through vertical and horizontal data flow. The potential value of sharing data between different partners is estimated to be more than USD 100 billion [6]. This Collaborative Manufacturing (CM) brings the ability to react to the customers' needs and requirements and provide a

significant advantage [2]. Thereby, data sharing between different stakeholders of the product life cycle, regardless of location, is a must for Industry 4.0 [7, 8]. Here IIoT plays a vital role as data producers for production flow, quality control, product traceability, supply chain, and enterprise decision management [7]. However, IIoT inherited from IoT poses a big challenge: overcoming the security concerns raised by the need to actually connect IIoT the Internet, in order to make it part of a larger and distributed network. This is due to the great security risk involved in this leap, i.e., 83% of the IoT devices used or manufactured by large companies experienced a cyber attack in 2018 and 79% of the manufacturing organizations admitted to being victims of a cyberattack in their IIoT the same year [9].

IoT nodes can be remotely attacked by exploiting their vulnerabilities [10, 11] and on the other hand, outgoing messages could be modified to attack the infrastructure or the messages could simply be read in the flow [12]. However, this is only part of the problem, as data normally needs to be stored to be eventually shared. However, data storage for IIoT presents two big challenges. The reliability of cloud storage is not trusted (1), it can fail and lose data accidentally, e.g., such as the failures of Amazon Web Services, Microsoft Azure, Alibaba Cloud and Google Cloud [13, 14, 15, 16], fog storage can intentionally modify the data to affect the user operation, and local storage is not an option due to the huge quantity of data generated by IIoT [7]. The other big challenge is ensuring the data confidentiality (2) [7, 12]. A common method to protect data confidentiality is by encryption in the form of cipher-text, and then forwarding the decryption key to the data readers. The problem with this mechanism is the complexity of managing accurate access control rules in order to provide least-privilege access to sensitive data and proprietary knowledge. Moreover, agents requiring data access may belong to different entities, have different levels of authorization, and have complex enroll/leave dynamics. Therefore data sharing with fine-grained authorization is a relevant challenge in IIoT [7].

There are many works in the state-of-the-art trying to solve this problem. Still, none of them can provide a completely secure solution, e.g., Ref. [17] ensured fine-grained authorization, but did not provide a method to protect the integrity of the data. On the other hand, Ref. [18] did protect the integrity of the data but did not propose any mechanisms to revoke the users. Therefore, they can ultimately not be applied in their current status.

This paper proposes a complete secure mechanism to collect and share data from IIoT devices with multiple readers with fine-grained access authorization. The proposed solution is tested and evaluated on a smart manufacturing use-case scenario identified in the context of the H2020 COLLABS project [19]. The contributions of the current paper can be summarized as follows:

- Maximization of data availability by using decentralized databases based on InterPlanetary File System (IPFS) [20].
- Data integrity through blockchain-based mechanisms.
- Strong identity and authenticity of IIoT through hardware-based encryption.
- Confidentiality in-transit and at-rest through end-to-end encryption.
- Fine-grained data access control by defining specific attributes and policies per datum through the use of Attribute-Based Encryption (ABE).
- Dynamic enrollment based on the OpenID standard [21].
- A novel reader revocation mechanism, beyond the capabilities of current research on fine-grained data access control.

The paper is structured as follows: Section 5.2 presents the industrial use case and its requirements; Section 5.3 reviews the related work; Section 5.4 presents the core technical concepts used in the proposed solution; Section 5.5 introduces the proposed solution from a high-level point of view; Section 5.6 explains in detail the design of the technical innovation achieved by this paper; Section 5.7 provides insights on the prototype implementation, discusses the evaluation results, and how the solution fits the use-case; Section 5.8 explains the findings of our work by comparing it with the state-of-the-art, highlights the limitations and discusses how they could be addressed as future work; Finally, Section 5.9 presents the conclusions.

5.2. Use Case and Requirements

The scenario described in this paper is a real-world scenario provided by Collins, one of the contributing organizations of this paper. In the context of safety-critical products, such as in aerospace systems, collecting and sharing data throughout the manufacturing lifecycle is of the utmost importance: the information produced is not only used for coordination of the manufacturing supply chain, but it is also used for quality controls or to prove standard adherence providing the final item with a safety certification that is strictly required to its usage. Of course, in the same context, the information required for a proficient CM might include important data containing sensitive information or intellectual property that must be protected.

Figure 5.1 represents a high-level architecture of the data flows. Notice that the architecture is divided into four different zones, each of which is mapped to the corresponding Purdue Enterprise Reference Architecture level, also known as Purdue Level (PL) [22]. These levels are used to relate a

specific architecture to key enterprise layers. Specifically, PL0 represents the physical process; PL1 includes devices that are directly interacting with the physical processes (i.e., sensing); PL2 describes the control systems that are monitoring and controlling the physical processes (i.e., SCADA); PL3 represents the manufacturing operations that manage production flows (i.e., manufacturing execution/operations); PL4, on the other hand, consists of IT networks, where the enterprise runs its business functions (i.e., engineering functions).

The following presents an introduction of the zones depicted in Figure 5.1:

- Shop Floor: this layer represents the inner-most layer of the enterprise, where the actual manufacturing activities are performed by industrial machines. This corresponds to PL0, PL1, and PL2.
- Manufacturing zone: this layer hosts all the industrial functionalities that directly manage the shop floor such as SCADA servers and other industrial controls. This corresponds to PL2, PL3, and PL4.
- Enterprise: this zone includes all the high level services of the company that are used to coordinate, manage, and operate the manufacturing operations such as the engineering center and the IT infrastructure. This corresponds to PL4.
- Public Internet: this zone represents all the services collaborating with the production of the final product that are not inside the company itself, for example, cloud services and supply chain partners. This layer is not mapped to any PL since it is physically and logically outside of the company.

The data flows shown in the architecture can be divided into two types: the black arrows identify data flows produced by equipment deployed on the shop floor, and the green arrows identify the data flow consumed by entities localized in any of the four layers. Notice that different types of devices are involved in the data collection phase, for example, modern industrial machines and sensors, which can be categorized as IIoT. On top of that, a typical shop floor usually hosts legacy industrial devices that might not be able to interface directly with the data-sharing system and therefore can use an industrial gateway as a trusted proxy to send their data to the data storage. All these devices gathering data during the manufacturing phases (for example manufacturing parameters such as production temperature, time, or precision) send this data to a central location (here identified as data storage) to be shared with the desired data consumers (hereafter called clients) afterward. Clients can analyze this information for performance improvement or quality assurance. On the other hand, the green lines identify data consumers: these

data flows identify a set of possible clients that need access to the data collected throughout the manufacturing lifecycle. The clients can be found at all levels of the architecture and might include employees of the factory (i.e., production manager, SCADA manager or engineers), automated data analysis services (i.e., quality and performance analysis tools or computing cloud services) or external partners (i.e., suppliers, customers, or certification authority). The shared data contains critical proprietary knowledge of the manufacturers and other stakeholders that shall be shared only with the specified client. Therefore, all the aforementioned data consumers must get access only to the minimum amount of data required to perform their duty (least knowledge principle [23]). However, obtaining fine-grained authorization in this scenario is complicated because managing diverse readers with different level of authorization and requirements can become highly intricate. In addition, thousands of IIoT devices generating data require ensuring that they are correctly configured with their policies and authorizations, which further complicates the authorization process. Additionally, clients and IIoT devices will dynamically join and leave the network requiring revocations, enrollments, and authorization updates, making the scenario a challenging undertaking.

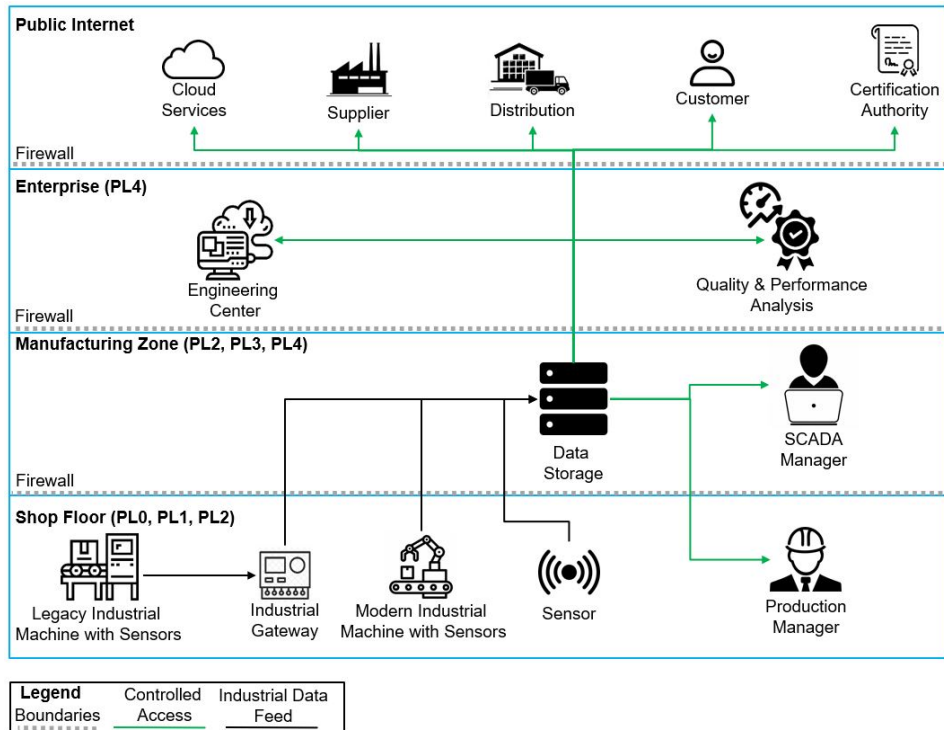


Figure 5.1: High-level scheme of factory scenario.

With the collaboration of a forum of industry experts involved in the COLLABS project [19], a list of high-level requirements for the CM dataflows identified [24] in the reference architecture has been redacted:

- RQ1: data confidentiality shall be protected while in transit and while at rest (end-to-end encryption).
- RQ2: clients shall have access only to the minimum amount of information required to perform their duty (least knowledge principle).
- RQ3: clients shall have access to the information only within a specific time-frame (time limited access).
- RQ4: information managed by the system shall have integrity guarantees to be used as proof in a certification process.
- RQ5: the solution architecture should support scalability and multi-site integration to be applicable in manufacturing scenarios with globally distributed manufacturing sites.
- RQ6: data in the system shall guarantee authenticity to provide confidence in its origin.
- RQ7: data in the system shall have non-repudiation guarantees to establish trust and accountability between the parties involved.

5.3. Related Work

This section presents the solutions in the state-of-the-art to provide confidentiality to large quantities of data with fine-grained authorization while ensuring the integrity of the data from generation to consumption.

Some solutions for access control tend to rely on the cloud services, which may compromise data security and privacy due to the reliance on a single cloud service provider with unrestricted access to all data. Moreover, the recent failures of major cloud providers such as Amazon Web Services, Microsoft Azure, Alibaba Cloud, and Google Cloud [13, 14, 15, 16] have highlighted the risks involved in relying solely on cloud services. To address this issue, classic cryptography techniques (such as symmetric and asymmetric cryptography) can reduce the required trust in cloud service providers. However, providing fine-grained access using these techniques remains a challenge.

A possible approach is to define a user list per file [25], but this requires the encrypter to know the data consumers beforehand and perform a specific encryption per data consumer (readers), which is not applicable to a distributed IoT-based environment with a high number of readers. A solution to this problem is Attribute-Based Encryption (ABE) [26], and its later variation Ciphertext-policy ABE (CP-ABE) [27] where a central entity

provides attributes to the readers, and the encryptor can easily define the combinations of attributes needed to decrypt its data. However, this solution does not offer efficient user revocation [28] or does not provide data integrity and authenticity guarantees.

There are multiple solutions trying to address the revocation problem. Historically, the proposed solutions were theoretical works proposing new algorithms or modifications of the original, such as Ref. [29, 30], which allow a dynamic membership. However, the first one requires trustful servers to avoid collaboration of servers and revoked users, which could lead to a collision attack. In addition, the server could modify the files during the re-encryption without being noticed. In the second work, after every revocation and key update, all the old messages are not reachable anymore with the new key, so a newly joined user cannot decrypt the previously published ciphertexts, i.e., they do not provide forward security [31]. Other theoretical solutions to this problem were presented in Refs. [32, 33], but they suffered from revocation collision attacks, (i.e., a revoked user can collaborate with a non-revoked user to recover the decryption capability of the revoked user accessing data that they could not get separately).

Revocation in ABE is still a challenging issue. On top of that, the lack of reliability, traceability, and availability in data storage services has also caught the attention of researchers as it affects the overall security of the infrastructure. To avoid relying on a third party to store the immutable proofs, decentralization stands out as a possible solution. The authors of Ref. [18] propose to store encrypted data (ciphertext) in the InterPlanetary File System (IPFS) [20]. IPFS is a decentralized data storage service that can handle large volumes of data, and it uses the ciphertext's hash as a locator, similar to a Uniform Resource Locator (URL) on the web. In addition, they store the ciphertext's hash on a blockchain. This research achieves reliability and transparency using blockchain, while benefiting from the availability and scalability of IPFS, and finally taking advantage of the fine-grained authorization of ABE. The approach proposed in Ref. [18] is similar to the scheme proposed in this paper but does not provide revocation nor allow multiple writers. In Ref. [34], blockchain is used to deliver attributes to consumers transparently and decentralized. In addition, the blockchain performs a pre-decryption of the ciphertext to relieve IoT devices of part of the decryption process while keeping confidentiality. To avoid users abusing the decryption services, they implement a user credibility incentive scheme. In Ref. [35], the authors create a Privacy-preserving Blockchain Energy Trading Scheme to manage energy production and sales transparently using blockchain and keeping the user privacy through using ABE. However, none of these three papers propose any revocation mechanisms for their systems.

Indeed, some works exploit the synergies of ABE and blockchain while claiming to have revocation mechanisms. In Ref. [36], an infrastructure is

created to share data from IoT nodes with fine-grain authorization. They propose to store the data encrypted with a symmetric encryption key in the cloud. The corresponding symmetric encryption key is encrypted with CP-ABE, which is called a header, and is stored in a smart contract. A client has to satisfy the policy defined in the smart contract in order to download the header. In this system, by modifying the smart contract, it is possible to modify the data access policy and thus to revoke users.

Another approach, described in Ref. [37], is an infrastructure that manages patient data shared among various hospitals and medical entities. This work aims to give patients the control over the access to their data. In this system, patients publish their data access attributes or policies in a permissionless blockchain. The patient or hospital then publishes a header in a permissioned blockchain, which only accepts storing the header if it is encrypted according to the guidelines defined in the permissionless blockchain. Using a “transaction consumption” mechanism, the patient can modify access to the header stored in the permissioned blockchain, and data consumers will not receive the data from the permission blockchain despite having an authorized access structure in the ABE encryption.

In Ref. [38], data is encrypted using a secret key K , which is generated through the secret keys K_1 and K_2 . All consumers have access to K_1 , but K_2 is encrypted with CP-ABE on the blockchain. All the nodes in the blockchain have the master key of CP-ABE and the attributes of all the consumers but lack K_1 , so they cannot access data even if they have K_2 . Eventually, when the consumer requests access to K_2 in the blockchain, the blockchain nodes verify their attributes and, if valid, generates their decryption key to decrypt K_2 and return it to the consumer. The consumer with K_1 and K_2 can then generate K and decrypt the message.

In Ref. [17] the authors use a Java-based blockchain to perform all the vital functions of ABE: generation of master key and the public key of ABE, and generation of the secret keys of each user. They obtain revocation by re-encrypting the ciphertexts using a group-based attribute access policy instead of the whole access policy.

The two last papers [38, 17] fail in a common element; they rely on the correct behavior of the individual blockchain nodes. A permissionless or permissioned blockchain is reliable for data integrity because the failure or dishonesty of one node is corrected by the others, so it can only be attacked with a “51 % attack”, i.e., more than half of blockchains miners agree to attack the system, which is very unlikely. However, a blockchain is not that reliable for confidentiality and privacy [39] since the failure or the dishonesty of only one of the blockchain nodes can lead to a sensitive data leak (e.g., a individual blockchain node can provide unauthorized data from the ledger to the user). This single-node misbehavior could lead to data leaks in both works. Additionally, Refs. [38, 17, 36] do not have any mechanism to verify the

integrity of the data received by the consumer. Finally, Refs. [36, 37, 38] do not have complete revocation mechanisms because they base their revocation on a policy update without giving further details, which means modifying the attributes and their relation in the crypto data. However, revocation is produced by an unexpected event that affects a user, not attributes, and therefore, it must be addressed to individual users [40, 41]. A policy update of the crypto data will very likely affect non-revoked users, therefore, it is needed to explain how to revoke only to the specified users without affecting non-revoked users.

Finally, Guangsheng Yu et al. [42] proposes a solution that addresses various problems related to storing encrypted data in blockchain. To ensure data integrity, they store the encrypted data directly in blockchain, using scalability solutions such as OmniLedger. They also introduce a new type of blockchain, called the Redactable Key Chain, which is based on Chameleon Hash and allows modification of hashes and blocks where they store the headers of the encrypted messages. This allows the Redactable Key Chain to modify blocks through a consensus mechanism, and then modifying the headers of the encrypted data. Their revocation mechanism is indirect revocation, which means updating the keys of all users except those of the revoked users. To perform the update of all keys in a scalable way, they use a concept called “updating factor”. When a non-revoked user receives an “updating factor”, they can update their keys. In addition, they provide another “updating factor” to the servers so they can update the headers without access to them. This solution has the following problems: it requires updating all the headers every time there is a revocation to guarantee forward security, i.e., new users have access to old data if they satisfy the policies [31]; it has no revocation collision resistance, i.e., non-revoked user Alice can share the Updating factor with revoked user Bob to gain access to data that only Bob could access if he were not revoked. Additionally, the concept of the “updating factor” is not well explained or supported by existing research.

As can be observed, many works are seeking a secure implementation of ABE and have found blockchain to provide a great synergy. However it is a big challenge to achieve revocation with collision resistance revocation, blockchain-based integrity protection, and without storing sensitive information in a blockchain. Nevertheless, these four features are necessary to apply ABE to real-world scenarios. This paper presents a probable secure solution for all these challenges. The proposed approach builds on Ref. [18], and improves it by providing a self-designed revocation mechanism, a dynamic user registry, and strong authenticity guarantees for multiples data senders, representing a significant step forward in the application of ABE in real-world scenarios. The reader can find a comparison of the shortcomings of the state-of-the-art and our work in Section 5.8.

5.4. Background

Our work relies on three existing technologies to achieve the security features that we claim: CP-ABE, blockchain, and Hardware Security Modules.

5.4.1. CP-ABE Functions

This work uses the CP-ABE solution proposed by Brent Waters in Ref. [43], Appendix A. In this construction, the set of attributes U is unlimited and the public parameters are constant. In CP-ABE, the readers have decryption keys with a set of attributes S , and the writers can encrypt a raw message (M) with public keys, defining a list of attributes combined with *and/or* logic functions called policy P , e.g., *Area51 and worker or manager*.

The attribute set has to satisfy the policy in order to decrypt the ciphertext (CT):

- $\text{Setup}()$: The setup function is executed by the trusted entity. The output is the Master Secret Key (MSK) and the ABE Public Key (ABE-PK). Once executed, these two elements remain constant.
- $\text{KeyGen}(\text{MSK}, S)$: The key generation function is also executed by the trusted entity. It takes the MSK and S . The output is an ABE Decryption Key DK with attributes S .
- $\text{Encrypt}(\text{ABE-PK}, P, M)$: The encryption function can be executed by any entity. It takes as input ABE-PK, and M . The output is CT .
- $Y(P, S)$ The satisfaction function $Y()$, checks if the attributes in the set S satisfy the policy P ; if yes return 1, if not, return 0.
- $\text{Decrypt}(CT, DK)$: This function takes an input a CT and a DK . The output is M if and only if $Y(P, S)$, where P is the policy defined during the encryption of CT and S is the set of attributes established in the generation of DK . In the rest of the cases, the output is \perp .

Different policies can be easily merged with *and* and *or* gates.

5.4.2. Blockchain

The blockchain as a distributed ledger was first proposed by Satoshi Nakamoto in 2008 [44]. Since then, this technology has gained significant attention and undergone numerous design variations. In 2014, Ethereum [45] introduced a new concept to blockchain with the creation of public and transparent script—the smart contracts. Nowadays the term blockchain identifies a set of distributed ledgers technologies where data can change only through a specific process. Data stored in blockchains is guaranteed to be

immutable, meaning it can never be altered or deleted. This is accomplished through the use of cryptographic hashes, digital signatures, and distributed consensus protocols. For example, adding information in a blockchain requires a consensus among the network's nodes. This consensus is achieved if the network participants verify the new information and agree to add them to the blockchain. In addition, to avoid the use of certificates in some blockchains, the use of "addresses" instead IDs is widespread. The address is a unique bit array computed from the public key with a collision resistance function. In this way, given a public key, it is possible to bind it with its assigned address without using a certificate. Blockchains can be public ledgers that everyone can join, or they can enforce authentication and authorization systems to enable participation as peers or clients. The latter type of blockchain are referred to as permissioned blockchains. The present work is developed using the permissioned blockchain Hyperledger Fabric [46].

5.4.3. Hardware Security Modules

Hardware Security Modules (HSM) are dedicated hardware implemented according to security-by-design principles and offering core security services and cryptographic operations. The elemental functions of an HSM are true random number generator used to generate private keys, secure data storage for the private credentials, and secure implementation of the standard cryptographic operations such as AES, ECC, or RSA. The HSMs are prepared to resist hardware-based attacks and Side-channel attacks [47]. Therefore, verifying the signature performed by a private key generated by an HSM ensures that the signature was also signed using this same HSM. These are only the more basic functionalities; however, more sophisticated HSMs such as those following the Trusted Platform Module 2.0 standard (TPM2.0) [48] offer more complex tools so that they can be used to remotely attest the software of an IoT device (Remote Attestation) or to specify policies before using the private keys, e.g., delivering a pin or proving adherence to a software status. Thereby, HSM can offer big reliability over the IoT devices and their operations, and are a perfect fit to answer IoT security requirements [49].

5.5. Proposed Scheme

This section presents the main functionalities of the proposed system from a high-level point of view. Here follows a list of the entities involved in the system:

- *Sensor*: this label represents all the devices in the Shop Floor that collect essential data and send them to the data storage.
- *Sensor Owner (SO)*: is the entity in charge of the correct operation of the sensors.

- *Client*: as explained in Section 5.2, a client is a data consumer that needs access to specific sensor data. They can be located in any of the four layers.
- *ID Provider (IDP)*: the entity that identifies the clients and their data requirements. Through the use of Open ID this entity can identify the *Client* and its attributes to *SO*.
- *IPFS message broker*: it is a decentralized message broker that can handle high volumes of data. Because the hash of the data is used as data locator, the integrity of the data is as trusted as the integrity of the hash.

Using the entities from the list, a simple sequence of actions is presented to showcase the system's capability. In the following example, a set of sensors is initiated with crypto material (Private/Public Keys, certificates, identifiers, etc.) and the related access policy. In this context, two users (Alice and Bob) will be used to demonstrate the possible operations of the systems, including encryption, decryption, and user revocation. The following lists explain the corresponding step number in Figure 5.2.

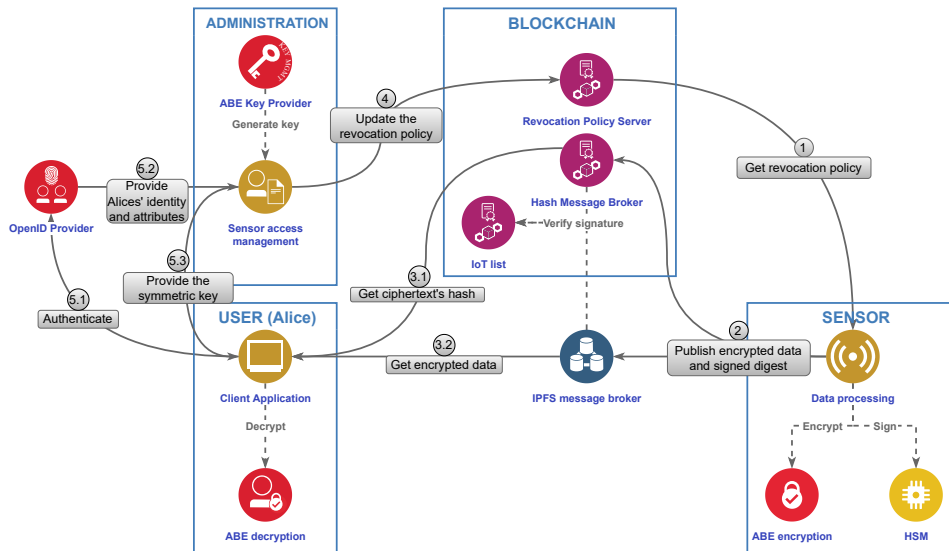


Figure 5.2: High-level scheme of the proposed solution.

0. **Setup phase:** *SO* generates the master key (private data) and the encryption key (public data). The encryption key is distributed to the sensors. Every sensor s gets an access policy Ap_s depending on the kind of data it generates. Finally, *SO* registers the sensors in a transparent list in the blockchain—the *IoT list*. After the setup phase, the operation of the system starts.

1. **Get the revocation policy, generate, encrypt, and sign data:** The sensor s gets a public policy called Revocation Policy, (Rp_i) from the blockchain, generates data, and encrypts them using Rp_i , its policy Ap_s , and a policy that changes with the time. The Slot Attribute (SA_t) gets the ciphertext. Then, it signs the ciphertext with its HSM.
2. **Publishing encrypted data and signed digest:** Next, the sensor sends the ciphertext to IPFS and sends the ciphertext's hash to the smart-contract *Hash Message Broker*. The smart-contract identifies the sensor before accepting the data through the *IoT list*. The IPFS nodes shall do the same identification to avoid a denial-of-service attack.
3. **Data retrieval:** Knowing the address of the sensor that generates their required data (public information), a *Client*, e.g. Alice, **gets the ciphertext's hash** from *Hash Message Broker* (3.1). Then, using the hash of the ciphertext, Alice **gets the encrypted data** from IPFS (3.2) and checks the integrity of the received ciphertext using the same hash. Next, Alice decrypts the data using her DK . For a successful decryption, Alice's attributes have to satisfy Ap_s , and the Rp_i used when encrypting the message. If the message is old, Alice's attribute must satisfy Ap_s and contain the correct *slot attribute* SA_t .
4. **User revocation:** When a user is revoked, e.g., Bob, *SO* **updates the Revocation Policy** with Rp_{i+1} , removing the decryption capabilities to Bob and all the users in his group. All the non-revoked users in the group, the *affected users*, will require a new DK , updating some of their attributes, the *revocation attributes*, while the remaining ones are unchanged, the *access attributes*.
5. **Decryption key update:** When Alice, the *affected user*, needs a DK update, she **authenticates** to *IDP* (5.1), which will **provide Alice's identity and attributes** to the *SO* (5.2). The latter needs to provide Alice with a new DK with her *access attributes* and the new *revocation attributes*. Generating a new DK to all *affected users* can take too long if they are a high number. Therefore, to avoid service interruption for the users, the new DKs are already generated, encrypted by the *SO* and stored by the users receiving the name of *backup decryption key*, bDK . The bDK of each user is encrypted with symmetric encryption using a *symmetric key* that is different for each user. Therefore, *SO* will only have to regenerate it and **provide the symmetric key** to Alice (5.3), which is much faster than generating a decryption key. Next, she can decrypt her bDK , which now will replace her DK . Then, while Alice already has her decryption capabilities recovered, *SO* generates a new bDK for Alice, which will take a longer time (around 3 minutes) but will not produce service interruption.

5.6. Details of the Proposed Scheme

This section will first describe the tools developed for the solution. Then, it will explain the phases in detail, referring to those tools.

5.6.1. Attributes

An innovative contribution proposed by this research is to take advantage of the infinite attribute universe U offered by the chosen ABE scheme. The attributes can constantly change without changing the master key, encryption, or decryption keys. We define the sets Q , W , and E from the universe U . We assume that Q , W , and E are disjoint subsets of U , i.e., no element in U belongs to more than one of these sets. Then, a client n has a set of attributes $A_n \subseteq Q \cup W \cup E$ with: *access attributes* set $AA = \{A_n \cap Q\}$, *slot attributes* $SA = \{A_n \cap W\}$ and *revocation attributes* $RA = \{A_n \cap E\}$:

- The *access attributes*, AA_n , are those related to the client n position, department, or responsibilities n (e.g. steel_quality_supervisor, efficiency, chemistry_reliability) and can be modified only with *SO* authorization.
- The *slots attributes*, SA_n , are used to define the periods of the past when the client n had read access. Given a time t_l in Coordinated Universal Time (UTC), its associated *slot*, $slot_l$, can be easily derived $slot_l = t/SlotPeriod$ where *SlotPeriod* is the duration of every *slot*. Then, from every $slot_l$, a unique attribute $sa_l \in W$ is associated, so $f_{asso} : UTC \rightarrow W$. When a sensor encrypts data at time t_l , it includes the attribute $sa_l = f_{asso}(t_l)$ in the encryption policy so that decryption keys with the attribute sa_l can decrypt the message. SA_n never contains the attribute of current *slot* nor the future *slots*. Therefore, it can only be used to access old encrypted data.
- Each user n is allocated to a group g_i and assigned to either subgroup $subg_{i,A}$ or $subg_{i,B}$. All the users in a subgroup share the same set of *revocation attributes* RA in their corresponding *DK*, namely $RA_{i,A}$ and $RA_{i,B}$ respectively. However, in their *bDK* they share a completely different set, $bRA_{i,A}$ and $bRA_{i,B}$. Notice that $RA_{i,A} \cap RA_{i,B} = \{A(g_i)\}$ where $A(g_i)$ is called the *Official revocation attribute* of g_i , $RA_{i,A} \cap bRA_{i,B} = \{ra_j\}$, $RA_{i,B} \cap bRA_{i,A} = \{ra_k\}$ and $bRA_{i,A} \cap bRA_{i,B} = \{ra_l\}$ where $ra_j \neq ra_k \neq ra_l \neq A(g_i)$. This distribution of the *revocation attributes* is used to revoke all the users of a subgroup without affecting their sibling subgroup. It is explained in more detail in the next subsection.

5.6.2. Revocation

The objective of the revocation is to remove the client's access to any data in the system. However, a malicious client could have downloaded, decrypted, and stored all the messages when he/she was still enrolled in the system. Therefore, perfect revocation is not possible in most cases, so the main purpose of the proposed scheme is to prevent the revoked client from accessing new data.

The user revocation in our system is performed by mixing indirect revocation and direct revocation [50]. In direct revocation, there is a list of revoked users, and the writers need to be aware of this list while encrypting to exclude the revoked users. In indirect revocation, all the users but the revoked user (all users are *affected users*) need to contact the key authority to get the update of their decryption keys. In our solution, we use both indirect and direct revocation.

To apply indirect revocation, we propose modifying one attribute, the *official revocation attribute*, which is present and required in all the policies of the ABE encryption. Then all the users need a new decryption key with the new attribute. With this system, there is no need to perform a new setup and change the *PK* for all the writers which can be annoying if there are too many IoT devices. However, indirect revocation still can be too computationally expensive for *SO* if the number of users that need a key update (*affected users*) is high. To avoid revoking all the users with every revocation, which would require many key updates, we apply direct revocation by dividing the users into groups, e.g., eight groups, where each group has an *official revocation attribute*. Then, during the encryption, the *official revocation attributes* of all the groups are included and combined with *or* gates so all the groups can decrypt the message. It still takes long to provide new keys to all the users in a group when there is a revocation. This problem would lead to a service interruption for all the *affected users*. To solve this problem, we exploited the concept of *backup decryption key*, *bDK*. All the users already have the new decryption key with their group's next *official revocation attribute*, the *bDK*, but it is encrypted using symmetric encryption. So, only providing the affected users with the symmetric key is enough to recover their decryption capabilities. All the *bDK* are encrypted using a different symmetric keys, so the revoked users cannot use the symmetric key of any other *affected user*, avoiding a collusion revocation attack. Since every symmetric key is only useful for the recipient users, the list with the symmetric keys for the *affected users* can be shared with gossip protocol [51] to avoid too many petitions to the *SO* and its size will be around 16 MB for 1,000,000 *affected users*.

Still, with this mechanism, the revocation policy has the size of the number of groups. Thus, this system uses what here is called subgroups, to reduce the revocation policy size to half of the number of subgroups. Notice that under each group, there are two sibling subgroups that are linked—

subgroup A and subgroup B ($sg_{i,A}$ and $sg_{i,B}$). Given the distributions of the RA explained in the last subsection, there is always a shared ra in the DKs of the subgroups $sg_{i,A}$ and $sg_{i,B}$, the $A(g_i)$, which is the one included in Rp to cover both subgroups. For example, when the user Bob from $sg_{i,B}$ is revoked, all users in $sg_{i,B}$ but Bob will get access to their bDK with $bRA_{i,b}$, then the new $A(g_i)$ will pass to be ra_j where $ra_j = RA_{i,A} \cap bRA_{i,B}$, and Rp will be modified to replace the old $A(g_i)$ with the new $A(g_i)$. So Alice from $sg_{i,A}$ with $RA_{i,A}$ will be able to satisfy the new Rp without the need to update her DK . All users in $sg_{i,b}$ will get a new bDK with a new $bRA_{i,b}$. The new $bRA_{i,b}$ will share a new ra with $RA_{i,A}$ and another with $bRA_{i,A}$, which requires ra , which will not have been shared previously from $RA_{i,A}$ and another in $bRA_{i,A}$. This mechanism has a limitation, as every time a user from a subgroup, e.g. $sg_{i,B}$, is revoked, the $RA_{i,B}$ is revoked, so its attributes become useless, and the ra shared with $bRA_{i,A}$ becomes useless too. Therefore, being $bRA_{i,A} = \{ra_1, ra_2 \dots ra_x\}$, it will become completely useless after x revocations of $RA_{i,B}$ if $bRA_{i,A}$ is not revoked earlier, i.e., there are not two revocations in $sg_{i,A}$. Equation (5.1) defines the probability, denoted as P_r , of experiencing x revocations in one subgroup before encountering two revocations in the sibling subgroup. We recommend distributing 10 ra in each RA , highly reducing the probability of this event to happen as defined in Equation (5.1). Still, this event would simply require updating also the bDK of the clients in $sg_{i,A}$ when there is a revocation in $sg_{i,B}$.

$$P_r = \frac{2(x+1)+2}{2^{(x+1)}} \quad (5.1)$$

With this solution, we mix both indirect and direct revocation to reduce the revocation list to a list of eight *revocation attributes* and to reduce the *affected users* to a 16th of the total users.

Notice that this mechanism is only needed for non-planned revocation. Nevertheless, usually, the revocations are planned, e.g., the contract of a worker or company finishes. Planned revocation does not require a short time slot, i.e., they can be revoked within the day. Therefore, all the planned revocations of a subgroup in the day are performed at once, highly reducing the impact on the SO . Moreover, they are preferably performed when there is a non-planned revocation in the subgroup within the day, which would avoid including any overhead in the SO .

Proof of Revocation

Revocation means that an adversary Adv_A revoked at time t will not be able to decrypt any new message published after time t . This property requires that even if the adversary has the AA that satisfy the Access policy of a message, such a message cannot be decrypted. To prove that our solution for revocation securely revokes Adv_A , we also consider an adversary Adv_B .

Adv_A and Adv_B are trying to distinguish two messages given the cipher text using a set of decryption keys that does not satisfy the policies. The objective of this proof is to show that Adv_A has the same advantage to break the solution as Adv_B to break the ABE construction of Waters [43].

- Theorem: Suppose that the assumptions of the ABE construction holds. Then no adversary with a revoked key at time t can distinguish a ciphertext from one of two messages encrypted after time t with non-negligible advantage.
- Setup: Challenger runs the Setup() function, and share ABE-PK with Adv_A and Adv_B . Challenger defines a published set of attributes $ARp \subset E$ and a policy RP such that RP is the *or* combination of all the elements of ARp . In addition, Challenger defines two sets of attributes $RA_{1,A} \subset E$ and $bRA_{1,A} \subset E$ such that $RA_{1,A} \cap ARp = \emptyset$ and $|bRA_{1,A} \cap ARp| = 1$ as defined in Section 5.6.1. Notice that with this given configuration $Y(RP, RA_{1,A}) = 0$ but $Y(RP, bRA_{1,A}) = 1$.
- Query phase: Adv_A request to Challenger j different DK_s , $\{DK_j\}_{j \in \mathbb{N}}$ associated j set of Access Attributes $\{AA_j \subseteq Q\}$ and j sets of Slot Attributes $\{SA_j = \{w \in W | w < sa_c\}\}$. Notice that Adv_A cannot include in its query attributes from the subset E and therefore any of these DKs can satisfy RP. Adv_B request to Challenger i different DKs, $\{DK_i\}_{i \in \mathbb{N}}$ associated with i sets of attributes $\{S_i\}$.
- Response phase: Challenger provides to Adv_A the j different DKs with the associated attributes $\{AA_j \cup SA_j \cup RA_{1,A}\}$. Challenger provides to Adv_B the i different DKs.
- Challenge: Adv_A prepares two equal length messages M_{A1} and M_{A2} . Then it gives the two messages and Access Policy Ap to Challenger. Challenger flips a coin c and encrypts M_{Ac} with policy $P = (Ap) \wedge (sa_c \vee Rp)$. The resulting ciphertext CT_A is given to Adv_A . Notice that $Y(P, AA_j \cup SA_j \cup RA_{1,A}) = 0 \forall j$. Adv_B prepares two equal length messages M_{B1} and M_{B2} . Then it gives the two messages and a policy $P : Y(P, S_k) = 0 \forall k \leq i$. Challenger flips a coin c and encrypts the M_{Bc} with the policy P and returns the cipher text to Adv_B .
- Guess: The Adv_A uses a function f_A that, given $\{DK_j\} \forall j$ and the encrypted message CT_A , makes a guess of M' with a non-negligible advantage (ϵ_A) , as shown in Equation (5.2):

$$Pr[f_A(\{DK_j\} \forall j, CT_A) = M_{ac}] = \frac{1}{2} + \epsilon_A \quad (5.2)$$

The Adv_B then uses a function f_B that given $DK_i \forall i$ and the encrypted message CT_B makes a guess of M' a non-negligible advantage (ϵ_B) , as shown in Equation (5.3):

$$Pr[f_B\{Dk_i\}\forall i, CT_B = Mb_c] = \frac{1}{2} + \epsilon_B \quad (5.3)$$

However, Adv_A cannot satisfy the policy P because $(AA_j \cap E = \emptyset) \wedge (SA_j \cap E = \emptyset) \wedge (sa_c \notin SA_j) \wedge (RA_{1,A} \cap ARp = \emptyset) \Rightarrow Y((sa_c \vee Rp), (AA_j \cap SA_j \cap RA_{1,A})) = 0$. Therefore, Adv_A and Adv_B are in the same scenario. If f_a exists, it could be used as a substitute of f_b by Adv_B to get a non-negligible advantage. However, the ABE construction used proves that f_b does not exist, and therefore neither f_a . In this way, we prove that Adv_A cannot get access to new messages after RP is updated at time t without first accessing $bRA_{1,A}$.

5.6.3. Smart Contracts

This section defines the smart contract interfaces and the algorithm logic. There are three smart contracts: Revocation Policy Smart Contract, IoT List, and Hash Message Broker.

- Revocation Policy Smart Contract (RPSM): a smart contract that holds the Revocation policy. All the participants in the blockchain can read this policy; however, only the SO can modify the policy. Section 5.6.2 provides more details about the revocation policy and its management.
- IoT list: a smart contract that stores the addresses of all the sensors authorized by the SO to send information to the system. Only the SO can modify this list, but all the members in the blockchain can read it. A transparent list in the blockchain is essential for the chain of trust to provide strong authentication guarantees of the sensors to the clients. More details are provided in Section 5.7.2.
- Hash Message Broker (HMB): this smart contract links every sensor's address with the hash of its messages and its metadata. When receiving a transaction from a sensor to store a hash, the smart contract queries the *IoT list* for the address of the sender. If the sender of the transaction is in the list, the hash and its metadata are stored linked to the address of the sender.

5.6.4. Phases

The proposed scheme is logically divided into four phases: set up, sending data, reading data, and updating the decryption keys.

5.6.4.1. Set Up ($SO \rightarrow Sensor, IoT List, RPSM$)

This phase includes all the processes that should be executed one time per system or per sensor. Firstly, from a random seed, SO generates the MSK

and ABE-PK. The random seed is deleted, and the MSK is kept secret. In addition, the *slot* duration, *SlotPeriod*, is defined (usually 24 h). When a new sensor s is to be included in the system, SO registers the s 's address in the smart contract *IoT list*, provides it with the ABE-PK, and defines the access policy Ap_s for the encrypted data of this *Sensor*. Finally, the SO uploads a Rp_i to RPSM. The Rp_i is a policy formed by the *Official revocation attributes* of all the groups united by or .

5.6.4.2. Sending Data ($Sensor \rightarrow IPFS, HMB$)

A sensor s generates a message M_i at a particular time t_i . Then from t_i the sensor gets the $sa_i \in W$, from the Revocation policy Smart Contract it gets the Revocation policy Rp_i , and performs the encryption as shown in Equation (5.4):

$$EM_i = \text{Encrypt}(ABE-PK, (Ap_s) \wedge (sa_i \vee Rp_i), M_i) \quad (5.4)$$

Then the encrypted message, EM_i , is signed and sent to IPFS. The last one can check the sender's validity in the *IoT list*. Meanwhile, the s uses the same credentials to send a transaction to the smart contract *Hash message broker* with the hash of the EM_i , $Hash_i$. If and only if the address of the IoT device is in the *IoT list*, $Hash_i$ and the M_i metadata are accepted, and then they are linked to the s 'address, including the encrypted message in the set of messages from s .

5.6.4.3. Reading Data ($HMB, IPFS \rightarrow Alice$)

The client Alice (a) needs data from a *Sensor* s . Knowing the address of the device, she can query the metadata of their messages and use it to identify the required message M_r . Alice tries to decrypt it using her decryption key DK , ($\text{Decrypt}(EM_r, DK)$). If M_r is a message from the current slot sa_c , the decryption will be successful if and only if, $Y(Rp_r, RA_a) \wedge Y(Ap_s, AA_a)$. However, when decrypting a non-current message from the slot sa_r , the decryption will be successful if and only if $sa_r \in SA_a \wedge Y(Ap_s, AA_a)$.

5.6.4.4. User Revocation: ($SO \rightarrow RPSM$)

When a user Bob from the group i and subgroup $subg_{i,B}$ is revoked, SO has to update the Rp in the RPSM. Particularly SO has to update the $A(g_i)$. The new $A(g_i)$ will be the only ra that is in the RA of the bDK of all the users of $subg_{i,B}$ and that it is at the same time in all the RA of the DK of all the users of $subg_{i,A}$, as shown in Equation (5.5):

$$A(g_i) = bRA_{i,B} \cap RA_{i,A} \quad (5.5)$$

This way, Bob and all the users in $subg_{i,B}$ will not be able to decrypt any new data. All the affected users in $subg_{i,B}$ will receive access to their bDK —apart from Bob.

5.6.4.5. Updating Decryption Key ($SO \rightarrow Alice$)

When a user is revoked, the Rp is updated, changing the *Official revocation attribute* of the group of the revoked user, which may make Alice lose her capability to decrypt new data. To avoid this, Alice will quickly get a symmetric key from SO or another client through a gossip protocol [51]. With this symmetric key, she will be able to decrypt her bDK , which has the new $A(g_i)$. In this moment, Alice starts to authenticate herself to $IDprovider$ in order to connect with SO and request a new bDK , which will be encrypted with a new symmetric key.

5.7. Evaluation

The prototype was deployed in a relevant environment for evaluation purposes. This test environment is a flexible cyber-physical system composed of specific hardware (i.e., industrial gateways and Raspberry PIs) and is extensible with virtual devices hosted on a server with 28 cores and 256 Gb of RAM. The laboratory is configured using VLANs to represent all the different layers in an Industry 4.0 architecture, as presented in Figure 5.1.

The prototype services are distributed on different devices (whether physical or virtual): The sensors were implemented using Raspberry Pi 4B [52] equipped with a real HSM, Iridium 9670-TPM2.0 chip [53] to perform secure crypto-operations. On the other hand, services which in a real scenario are hosted on IT network are deployed on virtual machines: the IPFS servers were distributed on four separate virtual machines, and blockchain nodes were implemented with Hyperledger Fabric 1.4 [46] in three other virtual machines, all of them with Ubuntu 20.04 [54] and 4 Gb of RAM.

5.7.1. Performance Evaluation

The proposed scheme requires a continuous update of the decryption keys, which can lead to a bottleneck in the decryption key provider, the SO . In this section, an analysis of the scalability of the proposed system is presented, considering the number of users as a scale-up factor. As explained in Section 5.6.2, the revocations are commonly planned and they do not affect the performance. However, in non-planned revocations, a group of non-revoked users lose access to data, requiring a key update, which can lead to a bottleneck in the SO . In the following we present the analysis results conducted by utilizing standard hardware to verify that no bottleneck is measured in practice.

Let us define Nu as the total number of users, Nsg as the number of subgroups and, NuG as the number of users per subgroup. In addition, consider Pv as the average time a user needs a non-planned revocation, and V_G as the number of DK that SO can generate every second.

To avoid SO overhead, the time of generating the bDK of a subgroup shall be less than the average time there is a revocation in the system. Equations (5.6)–(5.8) show the estimation of the maximum number of users in the proposed solution.

$$\frac{NuG}{V_G} \leq \frac{Pv}{Nu} \quad (5.6)$$

$$\frac{Nu}{Nsg * V_G} \leq \frac{Pv}{Nu} \quad (5.7)$$

$$Nu \leq \sqrt{Pv * Nsg * V_G} \quad (5.8)$$

Using the library OpenABE [55] in a laptop Thinkpad E495 with 16 GB RAM, AMD Ryzen 7 3700 and Radeon Vega Mobile GFX 8 [56], $Pv \geq 600$ DK/s, with 30 attributes in each DK , are observed. Using a Rp of 8 *official revocation attribute* united with *or* gates, it results in $Nsg = 16$. Pv can vary from scenario to scenario; nevertheless, assuming a similar behavior of X.509 certificates estimated by NIST [57], a 10% of key revocations are considered. This means that if a key has a lifetime of 3 months, every user needs a key revocation every 30 months on average. As a result, this system could manage 864,000 users with the computational power of a laptop. In addition, based on the estimation in Equation (5.8), by using a server 7 times as powerful as the test laptop the system could reach 2,000,000 users.

The analysis performed highlights that the solution has a limited scalability, as the number of maximum clients is proportional to the square root of the computational power. Therefore it could not be deployed to applications with massive user numbers such as Instagram or Telegram (which have more than 500 million users each). However, it can be considered applicable for most professional manufacturing environments (i.e., Infineon Technologies) given that only one laptop could manage the credentials of the workers of an entire company (50,000+ workers).

5.7.2. Security Evaluation

This section evaluates a set of security properties that the proposed solution is guaranteeing. In the first place, it presents the common security features guaranteed in most ABE-based infrastructures:

- Collision resistance: the most basic requirement for fine-grained authorization. All the sensitive information is encrypted with ABE mechanism, which provides collision resistance.
- End-to-end Encryption: the data is encrypted by the sensors using ABE and it is only decrypted by the clients with the correct DK 's client. Additionally, unlike in Ref. [38, 17], the proposed system does not trust the blockchain or IPFS with any sensitive information that could lead to a data leak.
- Forward secrecy: "The newly joined user can also decrypt the previously published ciphertexts if it has sufficient attributes" [31]. All the messages are encrypted including in the policy a *slot attribute*. Then, in the future, when a user (Alice) wants to access some old data from a particular $slot_i$, she simply asks the SO for a DK with her access attributes and sa_i .

In the following, we outline the essential security properties that are not guaranteed by all of the analyzed works in the state-of-the-art, while these properties are integral to our proposed solution:

- User Revocation (UR): the revocation of a user is performed by modifying a policy published in a smart contract on the blockchain. Once it gets published, it is applied to all the encryption processes, revoking a portion of the users. To avoid this portion of the users in the system losing their decryption capabilities when they should not be revoked *backup decryption keys*, the affected users get access to a symmetric key to decrypt their *backup decryption keys*, thus avoiding service interruption.
- Collision Resistance Revocation (CRR): as explained in Section 5.3, a revoked user could collaborate with a non-revoked user to get access to data that shall be inaccessible for the collaborators individually. In the proposed system, every *backup decryption key* is encrypted using a unique symmetric key. Therefore, the information needed to recover an affected user's data access (the symmetric key) is useless for any other user. Thus, a revoked user cannot collaborate with other users.
- Data Integrity (DI): defined here as the capacity of the client to verify the non-modification of the data. In the proposed system, the client does not need to verify any signature (apart from those needing to connect with the blockchain) in order to verify the data integrity. All data hashes are stored in the immutable blockchain; therefore, even if the data is stored in a non-trusted service such as IPFS, this data is as immutable as the blocks in the blockchain.

- No Sensitive Data in the Blockchain (NSDB): as explained in Section 5.3, sensitive data shall never be exposed to the blockchain. In our solution, no private keys or partial decryption keys are stored in the blockchain. The only public information is the identities of the IoT devices, in the *IoT list*, and the metadata needed to identify the required messages.

While the data authentication feature may be a straightforward task in ABE solutions involving human or organizational writers, the authentication of data in ABE applied to IoT devices requires further evaluation:

- Data authenticity: the capacity to securely identify the data generator entity. This means identifying that specific data in the system comes exactly from a particular IoT node. To do so, the smart contract *Hash message broker* only accepts data hashes sent in a signed transaction. From the signed transaction, the smart contract gets the sender's address and stores the hash together with the address if the address is approved in the smart contract *IoT list*. An address that is approved on the *IoT list* means that the public key utilized to generate the address has been verified to be associated with the Hardware Security Module of a specific IoT device. Although this paper does not cover this verification process, it can be accomplished using Platform Certificates part of the standard *Trusted Platform Module* [48] or using IDevID certificates of the standard *IEEE Std 802.1AR: Secure Device Identity* [58]. With this chain of trust, the data origin is as reliable and secure as the *Hardware Security Module* of the IoT device.

5.7.3. Requirements Evaluation

Table 5.1 explains how the proposed solution covers the necessities of secure data sharing identified in the industrial use case based on the technology performance and security evaluation.

Table 5.1: Evaluation of the requirements.

Requirement	Evaluation
RQ1	The system provides <i>end-to-end encryption</i> , which guarantees data confidentiality protection both in transit and at rest.
RQ2	Using ABE, the data access of the client can be defined beforehand through a combination of attributes and the data can be encrypted by defining a policy that can decrypt it. This way, it is possible to define fine-grained data access for each user. Additionally, given the <i>collision resistance</i> , different users cannot collaborate to access more data.
RQ3	There are two mechanisms used to limit the time access to users, non-planned revocation and planned revocation, which are both explained in Section 5.6.2. This way, the reading privileges of a user can be removed or modified whenever required.
RQ4	Because of the <i>data integrity</i> of the system based on blockchain, the information provided in the system by the IoT nodes has integrity guarantees to be used as proof of certification.
RQ5	As explained in Section 5.7.2, this system has limited scalability with the number of users; however, with basic resources, it can be applied to 800,000 users and could easily be increased to 2,000,000, which covers the majority of industrial scenarios.
RQ6	The proposed solution has a chain of trust ending in the reliability of the HSM in order to provide authenticity guarantees to the client. This is explained in detail in Section 5.7.2.
RQ7	The hash of all data and its metadata are stored in the blockchain in an immutable fashion, so even if critical data with high impact is intentionally deleted from the IPFS, any party that accessed the data can claim its veracity.

5.8. Discussion and Future Work

In this paper we have proposed a robust mechanism for many-to-many data sharing with security-by-design. In this solution, we have considered and covered the six well-known security threats of the STRIDE methodology: Spoofing, Tampering, Repudiation, Information disclosure, Denial of service, and Elevation of privilege [59]. To achieve this result, many different technologies had to be combined (Blockchain, ABE, HSM, IPFS, OpenID), and some even created (Revocation). In the end, we ended up with a complex system that delivers the necessary security properties for data sharing with fine-grained authorization: Authenticity, Integrity, Non-repudiation, Confidentiality, Availability, and Authorization.

Despite our achievement, there are other results in the state-of-the-art that offer solutions to many of the mentioned security threats, but as shown in Table 5.2, none of them fulfill them completely. It is important to consider that information security solutions do not follow a linear progression, in which each solution is incrementally stronger than the previous one. Instead, they resemble a chain, where the overall security is only as strong as its weakest link, as stated by Thomas Finne [60]. For this reason, we consider the current work to be a great leap forward from previous results as it finally offers a complete security solution for data sharing with fine-grained authorization supporting revocation, which is the long-standing problem of ABE. Such a result enables many new functionalities requiring data sharing that were previously impossible to achieve due to security reasons or because they were too tedious. For example, it greatly enhances the implementation of machine learning in smart factories, because one of the most challenging tasks in this technology is data collection [61], and this task can be even more complicated in the industry field due to the sensitivity of the information.

Table 5.2: Comparison of blockchain-based ABE protocols.

ABE and Blockchain	Security Properties			
	UR	CRR	DI	NSDB
[18]	X	X	✓	✓
[34]	X	X	✓	✓
[35]	X	X	✓	✓
[36]	X	X	X	✓
[37]	X	X	NA	✓
[38]	X	X	X	X
[17]	NA	NA	X	X
[42]	✓	X	✓	✓
Proposed solution	✓	✓	✓	✓

Even if our own work fits into most of the Industrial scenarios, it is far from perfectly applicable to overall scenarios. As explained in Section 5.7.1, it is necessary to prepare a server acting as a key provider (*SO*) with a capability proportional to the square of the clients. So, while it is easily applicable to more than a million clients, it would be almost impossible to apply this solution to a billion clients, such as those scenarios where IoT nodes also read data from the share data-based or in a global social network.

In addition, there are many additional ABE features, which although not necessary for a fully secure architecture and for our scenario, may provide more flexibility and applicability to the solution. From a Survey of attribute-based encryption, [62], we can discover the following ABE features that our solution does not have: Decentralized Authority, Asynchronous Authority, Key Delegation, and Privacy Preservation.

However, these missing features are not intrinsically unfeasible in our architecture—they were simply out of scope. For example, OpenID offers the possibility to identify the attributes of a user without revealing the user's identity for Privacy Preservation. It also provides the possibility to identify a client through several ID providers decentralizing attribute issuers. On the other hand, our solution is designed on the Waters construct, but there are many other ABE constructs with different features that could be applied as long as they allow infinite attributes. Additionally, Key Delegation could be performed in our solution by making the blockchain witness the operation to solve the system's accountability problem defined in Ref. [62]. Finally, the trustworthiness of the IoT devices is not covered in this work, nor is data re-encryption, hence, technologies such as remote attestation and Chameleon Hash-based blockchains are valid future works for this article. Therefore, there are many options for further research and adaptation of the solution to other scenarios with different requirements.

5.9. Conclusions

In this article, we present a real scenario where data generated from multiple IoT devices can be consumed by different stakeholders in a collaborative ecosystem motivated by the transition to Industry 4.0. We highlight the necessity of data authentication, integrity, non-repudiation, and confidentiality with fine-grained access control. Firstly, IPFS is used to provide data availability and scalability. We use blockchain to protect the metadata and data hashes, which is essential for the integrity protection of the data in IPFS. Then, the HSMs offer strong guarantees of the authenticity of the data generators and we create a smart contract IoT list to assure the correct identity of each HSM transparently and we implement OpenID to improve the dynamism of client enrollment in the system. Finally, we propose a novel solution to revoke users of ABE, using revocation attributes to reduce the

impact of indirect revocation and backup decryption keys to delete any possible side effect of the revocation on the legitimate users. In a performance evaluation, we prove that the solution can be easily applied to 2,000,000 users, which satisfies any industrial scenario. With our work, we demonstrate that controlled data sharing with all the security guarantees from many IoT devices to many clients is possible. This solution has the potential to make a significant impact in collaborative manufacturing and thus in the Industry 4.0. Our hope is that this investigation enables further research and solutions in the field of secure data sharing and ultimately forms part of the realization of revolution in the manufacturing process.

Abbreviations

The following abbreviations are used in this manuscript:

IoT	Internet of Things
IIoT	Industrial Internet of Things
CM	Collaborative Manufacturing
IPFS	InterPlanetary File System
ABE	Attribute-Based Encryption
PL	Purdue Level
CP-ABE	Ciphertext-Policy Attribute-Based Encryption
URL	Uniform Resource Locator
HSM	Hardware Security Modules
CT	CipherText
TPM2.0	Trusted Platform Module 2.0
UTC	Coordinated Universal Time
UR	User Revocation
Q	Set of attributes that contains all the <i>access attributes</i>
W	Set of attributes that contains all the <i>slots attributes</i>
E	Set of attributes that contains all the <i>revocation attributes</i>
AA_n	Set of attributes that contains all the <i>access attributes</i> of client n
SA_n	Set of attributes that contains all the <i>slots attributes</i> of client n
DK	Decryption key of ABE, it has attributes assigned and it is unique per client
bDK	Backup of a DK , it is encrypted with an unique symmetric key
g_i	Group i
$A(g_i)$	<i>Official revocation attribute</i> of g_i
$subg_{i,A}$	First subgroup of group i
$subg_{i,B}$	Second subgroup of group i
$RA_{i,A}$	Set of <i>revocation attributes</i> of clients' DK in $subg_{i,A}$
$bRA_{i,A}$	Set of <i>revocation attributes</i> of clients' bDK in $subg_{i,A}$
t_n	Particular time n

sa_i	Slot attribute used in encryption at t_i
$ABE-PK$	Public key needed to perform ABE encryption
MSK	Master Secret Key of ABE, needed to create DK and PK
SO	Sensor Owner, responsible for the management of ABE and the only entity with knowledge of MSK
IDP	ID provider, identifies the client and their attributes to SO using OpenID
HMB	Hash Message Broker smart contract
R_p	Revocation policy, combination of all the $A(g)$ united with <i>or</i> logic functions
$RPSM$	Revocation Policy smart contract
Adv	Adversary
Nu	Number of clients in the system
NuG	Number of clients per group
Pv	Average time a client needs a non-planned revocation
V_G	Number of DK that SO can generate every second
CRR	Collision Resistance Revocation
DI	Data Integrity
NSDB	No Sensitive Data in the Blockchain

References

- [1] Elijah, O.; Ling, P.A.; Abdul Rahim, S.K.; Geok, T.K.; Arsad, A.; Kadir, E.A.; Abdurrahman, M.; Junin, R.; Agi, A.; Abdulfatah, M.Y. A Survey on Industry 4.0 for the Oil and Gas Industry: Upstream Sector. *IEEE Access* **2021**, *9*, 144438–144468. <https://doi.org/10.1109/ACCESS.2021.3121302>.
- [2] Lasi, H.; Fettke, P.; Kemper, H.G.; Feld, T.; Hoffmann, M. Industry 4.0. *Bus. Inf. Syst. Eng.* **2014**, *6*, 239–242.
- [3] Kim, J.H. A review of cyber-physical system research relevant to the emerging IT trends: industry 4.0, IoT, big data, and cloud computing. *J. Ind. Integr. Manag.* **2017**, *2*, 1750011.
- [4] Soori, M.; Arezoo, B.; Dastres, R. Internet of things for smart factories in industry 4.0, a review. *Internet Things Cyber-Phys.Syst.* **2023**.
- [5] Company, T.B.R. *IoT in Manufacturing Global Market Report 2023*; ResearchAndMarket, U.S. 2023.
- [6] Betti, F.; Bezamat, F.; Fendri, M.; Fernandez, B.; Küpper, D.; Okur, A. Share to Gain: Unlocking Data Value in Manufacturing. World Economic Forum, 2020. Available online: https://www3.weforum.org/docs/WEF_Share_to_Gain_Report.pdf (accessed on 20 June 2023).

-
- [7] Yu, Y.; Chen, R.; Li, H.; Li, Y.; Tian, A. Toward Data Security in Edge Intelligent IIoT. *IEEE Netw.* **2019**, *33*, 20–26. <https://doi.org/10.1109/MNET.001.1800507>.
- [8] Müller, J.M.; Veile, J.W.; Voigt, K.I. Prerequisites and incentives for digital information sharing in Industry 4.0—An international comparison across data types. *Comput. Ind. Eng.* **2020**, *148*, 106733.
- [9] Irdeto. *Global Connected Industries Cybersecurity Survey*; Irdeto 2019.
- [10] Waraga, O.A.; Bettayeb, M.; Nasir, Q.; Talib, M.A. Design and implementation of automated IoT security testbed. *Comput. Secur.* **2020**, *88*, 101648.
- [11] Sarker, I.H.; Khan, A.I.; Abushark, Y.B.; Alsolami, F. Internet of things (IoT) security intelligence: a comprehensive overview, machine learning solutions and research directions. *Mob. Netw. Appl.* **2022**, <https://doi.org/10.1007/s11036-022-01937-3>
- [12] Abiodun, O.I.; Abiodun, E.O.; Alawida, M.; Alkhaldeh, R.S.; Arshad, H. A review on the security of the internet of things: challenges and solutions. *Wirel. Pers. Commun.* **2021**, *119*, 2603–2637.
- [13] Service, A.W. *Summary of the Amazon S3 Service Disruption in the Northern Virginia (US-EAST-1) Region*; Amazon Web Services, Inc. 2017. Available online: <https://aws.amazon.com/message/41926/> (accessed on 26 June 2023).
- [14] Moss, S. *Microsoft Azure Suffers Outage after Cooling Issue*; DCD 2018. Available online: <https://www.datacenterdynamics.com/en/news/microsoft-azure-suffers-outage-after-cooling-issue/> (accessed on 26 June 2023).
- [15] Fu, Y. *Alibaba Cloud Reports IO Hang Error in North China*; Shanghai EqualOcean Technology Co., Ltd. 2019. Available online: <https://equalocean.com/news/201903031507> (accessed on 26 June 2023).
- [16] Judge, P.; Swinhoe, D. *Cooling Failure Brings Down Google Cloud Data Center in London on UK's Hottest Day*; DCD 2022. <https://www.datacenterdynamics.com/en/news/cooling-failure-brings-down-google-cloud-data-center-in-london-on-uks-hottest-day/> (accessed on 26 June 2023).
- [17] Sharma, P.; Jindal, R.; Borah, M.D. Blockchain-based cloud storage system with CP-ABE-based access control and revocation process. *J. Supercomput.* **2022**, *78*, 7700–7728

-
- [18] Wang, S.; Zhang, Y.; Zhang, Y. A blockchain-based framework for data sharing with fine-grained access control in decentralized storage systems. *IEEE Access* **2018**, *6*, 38437–38450.
- [19] COLLABS-871518 Project Website. Available online: <https://www.collabs-project.eu/> (accessed on 26 June 2023).
- [20] Benet, J. Ipfs-content addressed, versioned, p2p file system. *arXiv* **2014**, arXiv:1407.3561.
- [21] Foundation, O. OpenID. Available online: <https://openid.net/> (accessed on 19 June 2023).
- [22] Williams, T.J. The Purdue enterprise reference architecture. *Comput. Ind.* **1994**, *24*, 141–158.
- [23] Lienberherr, K. Formulations and Benefits of the Law of Demeter. *ACM SIGPLAN Not.* **1989**, *24*, 67–78.
- [24] Consortium, C. *D1.2 COLLABS System Architecture Definition*; COLLABS EU-Project, Europe 2022. Available online: <https://www.collabs-project.eu/wp-content/uploads/2022/07/D1.2.pdf> (accessed on 26 June 2023).
- [25] Hwang, Y.H.; Lee, P.J. Public key encryption with conjunctive keyword search and its extension to a multi-user system. In Proceedings of the Pairing-Based Cryptography–Pairing 2007: First International Conference, Tokyo, Japan, 2–4 July 2007; Springer: Berlin, Heidelberg, 2007; pp. 2–22.
- [26] Sahai, A.; Waters, B. Fuzzy identity-based encryption. In Proceedings of the Advances in Cryptology–EUROCRYPT 2005: 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Aarhus, Denmark, 22–26 May 2005; Springer: Berlin, Heidelberg 2005; pp. 457–473.
- [27] Bethencourt, J.; Sahai, A.; Waters, B. Ciphertext-Policy Attribute-Based Encryption. In Proceedings of the 2007 IEEE Symposium on Security and Privacy (SP '07), Oakland, California, 20–23 May 2007; pp. 321–334. <https://doi.org/10.1109/SP.2007.11>.
- [28] Albulayhi, K.; Abuhussein, A.; Alsubaei, F.; Sheldon, F.T. Fine-grained access control in the era of cloud computing: An analytical review. In Proceedings of the 2020 10th Annual Computing and Communication Workshop and Conference (CCWC), Las Vegas, NV, USA, 6–8 January 2020; pp. 748–755.

- [29] Li, M.; Yu, S.; Zheng, Y.; Ren, K.; Lou, W. Scalable and secure sharing of personal health records in cloud computing using attribute-based encryption. *IEEE Trans. Parallel Distrib. Syst.* **2012**, *24*, 131–143.
- [30] Yu, S.; Ren, K.; Lou, W. FDAC: Toward Fine-Grained Distributed Data Access Control in Wireless Sensor Networks. *IEEE Trans. Parallel Distrib. Syst.* **2011**, *22*, 673–686. <https://doi.org/10.1109/TPDS.2010.130>.
- [31] Yang, K.; Jia, X. Expressive, Efficient, and Revocable Data Access Control for Multi-Authority Cloud Storage. *IEEE Trans. Parallel Distrib. Syst.* **2014**, *25*, 1735–1744. <https://doi.org/10.1109/TPDS.2013.253>.
- [32] Tysowski, P.K.; Hasan, M.A. Hybrid Attribute- and Re-Encryption-Based Key Management for Secure and Scalable Mobile Applications in Clouds. *IEEE Trans. Cloud Comput.* **2013**, *1*, 172–186. <https://doi.org/10.1109/TCC.2013.11>.
- [33] Hur, J.; Noh, D.K. Attribute-based access control with efficient revocation in data outsourcing systems. *IEEE Trans. Parallel Distrib. Syst.* **2010**, *22*, 1214–1221.
- [34] Qin, X.; Huang, Y.; Yang, Z.; Li, X. LBAC: A lightweight blockchain-based access control scheme for the internet of things. *Inf. Sci.* **2021**, *554*, 222–235.
- [35] Guan, Z.; Lu, X.; Yang, W.; Wu, L.; Wang, N.; Zhang, Z. Achieving efficient and Privacy-preserving energy trading based on blockchain and ABE in smart grid. *J. Parallel Distrib. Comput.* **2021**, *147*, 34–45.
- [36] Zhang, Y.; He, D.; Choo, K.K.R. BaDS: Blockchain-based architecture for data sharing with ABS and CP-ABE in IoT. *Wirel. Commun. Mob. Comput.* **2018**, *2018*, 1–9.
- [37] Pournaghi, S.M.; Bayat, M.; Farjami, Y. MedSBA: a novel and secure scheme to share medical data based on blockchain technology and attribute-based encryption. *J. Ambient Intell. Humaniz. Comput.* **2020**, *11*, 4613–4641.
- [38] Jemel, M.; Serhrouchni, A. Decentralized access control mechanism with temporal dimension based on blockchain. In Proceedings of the 2017 IEEE 14th International Conference on e-business Engineering (ICEBE), Shanghai, China, 4–6 November 2017; pp. 177–182.
- [39] Zhang, R.; Xue, R.; Liu, L. Security and privacy on blockchain. *ACM Comput. Surv. (CSUR)* **2019**, *52*, 1–34.

- [40] Horváth, M. Attribute-based encryption optimized for cloud computing. In Proceedings of the SOFSEM 2015: Theory and Practice of Computer Science: 41st International Conference on Current Trends in Theory and Practice of Computer Science, Pec pod Sněžkou, Czech Republic, 24–29 January 2015; Springer: Berlin, Heidelberg 2015; pp. 566–577.
- [41] Liang, X.; Li, X.; Lu, R.; Lin, X.; Shen, X. An efficient and secure user revocation scheme in mobile social networks. In Proceedings of the 2011 IEEE Global Telecommunications Conference-GLOBECOM 2011, Houston, TX, USA, 5–9 December 2011; pp. 1–5.
- [42] Yu, G.; Zha, X.; Wang, X.; Ni, W.; Yu, K.; Yu, P.; Zhang, J.A.; Liu, R.P.; Guo, Y.J. Enabling attribute revocation for fine-grained access control in blockchain-IoT systems. *IEEE Trans. Eng. Manag.* **2020**, *67*, 1213–1230.
- [43] Waters, B. Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization. In Proceedings of the Public Key Cryptography–PKC 2011: 14th International Conference on Practice and Theory in Public Key Cryptography, Taormina, Italy, 6–9 March 2011; Springer: Berlin, Heidelberg 2011; pp. 53–70.
- [44] Nakamoto, S. *Bitcoin: A Peer-to-Peer Electronic Cash System*; Bitcoin.org 2008. Available online: <https://bitcoin.org/bitcoin.pdf> (accessed on 26 June 2023).
- [45] Foundation, E. Ethereum White Paper. Available online: <https://ethereum.org/en/whitepaper/> (accessed on 18 August 2022).
- [46] Androulaki, E.; Barger, A.; Bortnikov, V.; Cachin, C.; Christidis, K.; De Caro, A.; Enyeart, D.; Ferris, C.; Laventman, G.; Manevich, Y.; et al. Hyperledger fabric: A distributed operating system for permissioned blockchains. In Proceedings of the Thirteenth EuroSys conference, Porto, Portugal, 23–26 April 2018; pp. 1–15.
- [47] Lou, X.; Zhang, T.; Jiang, J.; Zhang, Y. A survey of microarchitectural side-channel vulnerabilities, attacks, and defenses in cryptography. *ACM Comput. Surv. (CSUR)* **2021**, *54*, 1–37.
- [48] Group, T.C. *Trusted Platform Module Library Part 1: Architecture*; TCG, 2019. Available online: <https://trustedcomputinggroup.org/wp-content/uploads/TPM-Rev-2.0-Part-1-Architecture-01.16.pdf> (accessed on 26 June 2023).
- [49] Xu, T.; Wendt, J.B.; Potkonjak, M. Security of IoT systems: Design challenges and opportunities. In Proceedings of the 2014 IEEE/ACM International Conference on Computer-Aided Design (ICCAD), San

- Jose, CA, USA, 3–6 November 2014; pp. 417–423. <https://doi.org/10.1109/ICCAD.2014.7001385>.
- [50] Attrapadung, N.; Imai, H. Attribute-based encryption supporting direct/indirect revocation modes. In *Proceedings of the IMA International Conference on Cryptography and Coding*; Cirencester, UK; 15-17 December 2009, Springer: Berlin, Heidelberg, 2011. pp. 278–300.
- [51] Allavena, A.; Demers, A.; Hopcroft, J.E. Correctness of a gossip based membership protocol. In *Proceedings of the Twenty-Fourth Annual ACM Symposium on Principles of Distributed Computing*, Las Vegas, NV, USA, 17–20 July 2005; pp. 292–301.
- [52] Raspberry Pi Foundation. *Raspberry Pi 4B*; Cambridge, UK. Available online: <https://www.raspberrypi.com/products/raspberry-pi-4-model-b/> (Access on 26 June 2023).
- [53] Infineon Technologies AG. *IRIDIUM SLI 9670 TPM2.0*; Munich, Germany. Available online: <https://www.infineon.com/cms/en/product/evaluation-boards/iridium-sli-9670-tpm2.0/> (Access on 26 June 2023).
- [54] Canonical Ltd.; Open source; *Ubuntu 20.04*; Available online: <https://old-releases.ubuntu.com/releases/20.04/> (Access on 26 June 2023).
- [55] Zeutro, L. *OpenABE*; GitHub, Inc. San Francisco, California, 2018. Available online: <https://github.com/zeutro/openabe> (Access on 26 June 2023).
- [56] Lenovo Group Limited. *ThinkPad E495*; Hong Kong, China. Available online: <https://www.lenovo.com/de/de/p/laptops/thinkpad/thinkpade/e495/22tp2tee495> (Access on 26 June 2023).
- [57] Berkovits, S.; Chokhani, S.; Furlong, J.A.; Geiter, J.A.; Guild, J.C. *Public Key Infrastructure Study*; Technical Report; National Inst of Standards and Technology: Gaithersburg, MD, USA, 1994.
- [58] *IEEE Standard for Local and Metropolitan Area Networks - Secure Device Identity* in IEEE Std 802.1AR-2018 (Revision of IEEE Std 802.1AR-2009) New York, USA. 2018; pp. 1–73. <https://doi.org/10.1109/IEEESTD.2018.8423794>.
- [59] Howard, M.; Lipner, S. *The Security Development Lifecycle*; Microsoft Press: Redmond, WA, USA, 2006; Volume 8.
- [60] Finne, T. The information security chain in a company. *Comput. Secur.* **1996**, *15*, 297–316.

-
- [61] Roh, Y.; Heo, G.; Whang, S.E. A Survey on Data Collection for Machine Learning: A Big Data - AI Integration Perspective. *IEEE Trans. Knowl. Data Eng.* **2021**, *33*, 1328–1347. <https://doi.org/10.1109/TKDE.2019.2946162>.
- [62] Qiao, Z.; Liang, S.; Davis, S.; Jiang, H. Survey of attribute based encryption. In Proceedings of the 15th IEEE/ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD), Las Vegas, NV, USA, 30 June–2 July 2014; pp. 1–6. <https://doi.org/10.1109/SNPD.2014.6888687>.

Chapter 6

Towards sensor measurement reliability in Blockchain

Ernesto Gómez-Marín ^{1,2}, Luis Parrilla ², Jose L. Tejero López ²,
Diego P. Morales² and Encarnación Castillo ²

1. Infineon Technologies AG, Am Campeon 1-15, 85579 Neubiberg,
Germany

2. Departamento de Electrónica y Tecnología de Computadores,
Universidad de Granada, 18071 Granada, Spain

MDPI Sensors

- Received November 2023, Under review
- N/A
- Impact factor: 3.847
- JCR Rank: 19/64 in category Instruments and instrumentation (Q2)

Abstract: This document presents a secure architecture for sending data from an Internet of Things (IoT) sensor to Blockchain. Blockchain can process critical information with very high reliability, but it frequently requires information from the real world, where this information may be erroneous, intentionally or unintentionally. The entities that provide this information are called Oracles. Oracles are troublesome because the veracity of their provided information affects any Blockchain-based application. This is called "the Oracle Problem". Different solutions to approach this problem have been carried out in the literature, implementing secure Oracles. However, there is not a single solution for achieving a secure implementation of an IoT device acting as an Oracle, and thus, to securely send sensor data to Blockchain. In order to fill this gap, this paper presents a holistic solution which enables Blockchain to verify the security requirements, which are authenticity, traceability, integrity, device trustworthiness and data freshness, before accepting incoming data from an IoT node. The proposed solution uses Hardware Security Modules (HSMs) and a novel Public Key Infrastructure (PKI) based on Blockchain, implemented on Ethereum. An evaluation of the performance and a detailed security analysis prove the validity and the improvements provided by our proposal when compared with the State of the Art.

Internet of Things (IoT), Blockchain, Smart Contract, Hardware Oracle, Public Key Infrastructure (PKI), Trustworthiness.

6.1. Introduction

Blockchain is a peer-to-peer infrastructure where multiple nodes perform a synchronized computation and decide among them what is the correct result through a consensus algorithm to either updating of the database (ledger) or the executing programs (smart contracts). This system was proposed by Satoshi Nakamoto in Bitcoin [1] where multiple nodes maintain a common database to store transactions. It was later updated in Ethereum [2] in 2014 to include into the nodes a virtual machine (Ethereum Virtual Machine) that could deterministically execute the scripts stored in the Blockchain (smart contracts). As a result of several computers executing and verifying the same operations, Blockchains are quite inefficient and not suitable for complex computation tasks, but because its fundamental features (immutability, reliability, accountability, availability) [3], it is considered a Trust Machine [4] resulting in a new paradigm for secure computing.

The Trust Machine can run scripts that receives the name of smart contracts. They are executed by the decentralized infrastructure, ensuring that the assets managed in the script are handled as intended. There are many current applications that execute part of their logic on the Blockchain to benefit from its secure features. These decentralized applications are called

Dapps [5].

When a smart contract receives an input, it verifies the digital signature, processes this input and moves the digital assets as stipulated in the script, leaving the entire infrastructure aware of the transaction and saving the result in a list of records defined as blocks. The existence of Dapps revolves around the proper handling of the assets on the Blockchain and the high reliability of this process. Smart contracts are designed to support high liability, but as mentioned above, the entire execution of smart contracts depends on the inputs received. The entities in charge of uploading these inputs to the Blockchain, data on which the correct execution of smart contracts depends on, are called Oracles. There is always a risk that these Oracles could provide corrupt, malicious, or inaccurate data [6]. Thus, these Oracles must be able to deliver the trust that smart contracts require, have a high availability and demonstrate the use of reliable sources [7]. This is called the "oracle problem" [8].

On the other hand, the widespread usage of Blockchain in the Internet of Things (IoT), will largely increase the number of use cases in which this information is generated and imported to the Blockchain by IoT nodes [9] receiving the name of hardware Oracles [9, 10, 11, 12]. For example, providing information about a product temperature along a cold chain [13, 11, 14], ensuring the correct manufacturing of parts that have to follow high safety standards or penalizing the organizer of an event if the amount of CO₂ in the room rises above certain thresholds. The synergy between IoT and Blockchain has proven to be extremely advantageous in numerous surveys in the field [7, 15, 16]. However, IoT systems naturally lack trusted relationships which makes it very difficult to ensure their credibility, the no tampering of the data flow [17, 12] and their correct software status since these devices are insecure [18]. Therefore, it is unfeasible at this time to develop real applications subordinated to information coming from hardware Oracles.

Any non-hardware Oracle (e.g. software Oracles, human Oracles) that uploads information to the Blockchain from IoT nodes ultimately relies on the veracity of their data from the IoT nodes themselves. Even if the Oracle is reliable as in the case of DiOr-SGX [19] and J. Heiss' work [20], if they get incorrect information from the IoT nodes, whether on purpose or accidentally, this misleading information will spread to the Blockchain and go undetected. Therefore, the creation of a hardware Oracle giving smart contracts the capacity to independently check the accuracy of data from the real world is crucial for the continued growth of IoT applications based on Blockchain and, by extension, the Blockchain itself.

Thus, this work is the first to propose the use of secure hardware that signs the data even before being gathered by the IoT nodes and sends the real-world information to the Blockchain where the smart contracts itself verify the signatures, achieving an end-to-end data signature. To achieve

this, we rely on trustful remote-sensing architectures, with measurements protected by hardware [21] and we develop a novel Blockchain-based Public Key Infrastructure (PKI) that allows smart contracts to identify the IoT nodes, their origins, their security features and the data timestamp. Our work has a high synergy with current server Oracles that gather data from IoT nodes, as it enables them to provide also references of the origin and reliability of their gathered data. In a nutshell, our objective is to attain a stage where we can confidently rely on the data generated by an IoT device to execute critical actions of utmost responsibility.

The remaining of the paper is structured as follows: Section 6.2 defines the methodology to analyse the reliability of incoming data and explains the related work in the State of the Art (SoA) and Section 6.3 gives needed information to understand the proposed work. Then, Section 6.5 explains firstly how the system works and secondly explains the details of the design, Section 6.6 describes our real implementation in a controlled environment, Section 6.7 analyses our proposal using a detailed methodology and compares the results with the SoA, and finally, Section 6.8 closes the paper summarizing the results, contributions and future work.

6.2. Related work

The creation of trusted hardware Oracle requires a comprehensive approach. Every step in the data flow must be secured in order to trust the final data [14].

Firstly we will define the security requirements of the information using the work of Dan Liu *et al.*[22] in the subsection 6.2.1. There, we will discuss how to apply their secure data analytics guided on edge computing to IoT for Blockchain. Later, we will use these security requirements to analyze the SoA and our work.

Then, to analyze the SoA, we divide it into three sections. Firstly, we explore the research related to the design of a Blockchain-based PKI (6.2.2). Secondly, we delve into studies that use IoT in Blockchain as data providers without security mechanisms (6.2.3). Lastly, we examine works that design a secure mechanism to upload general data to the Blockchain (6.2.4).

6.2.1. Data Security Requeriments

Liu *et al.* [22] compare the existing works in secure data analytics based on edge computing. For this comparison, they define a number of requirements that are divided into three categories: security, privacy, and performance. Despite being a work focused on edge computing, the security requirements are perfectly applicable to analyze IoT as Oracle in Blockchain. In total, five security requirements have been considered:

- Data Origin Authenticity (DOAu): It refers to the authenticity in an infrastructure of the device that generates particular data. It is called Authenticity (Au) in [22].
- Data Origin Traceability (DOTa): It refers to the capability of backward identification of a data generator from the data. It is called Traceability (Ta) in [22].
- Data Origin Integrity (DOI): It refers to the capability of proving that the data generated in a particular point, was not manipulated in the course to its final point. It is defined as Integrity (I) in [22].
- Data Origin Trustworthiness (DOTu): It is the capacity to prove that the entity that generated a particular data was not manipulated or attacked, i.e., it was in a trusted status when it generated the data. It is called Trustworthiness (Tu) in [22].
- Data Origin Freshness (DOF): It is the capacity to prove that the data was generated in an absolute timestamp. It is essential to avoid replay attacks and delay attacks (attacks in which a measurement is taken in a particular time, detained, and published later). It is an additional property that was not included in [22].

6.2.2. PKI in Blockchain.

In this section we will focus in those works that investigate solutions of PKI for IoT in existing Blockchain. There are many interesting works developing an IoT PKI creating their own Blockchain infrastructure or consensus protocol like [17, 23, 24, 25]. But those works cannot be applied to existing public Blockchain e.g. Ethereum or consortium Blockchain as Hyperledger Fabric. In our work, we want to apply secure sensors to existing Blockchains. Therefore, these solutions are out of our scope. We will focus on those that can be applied to the well-known Blockchains.

In IKP [26], Stephanos M. and Raphael M. R. create a system to contribute to the current Transport Layer Security (TLS) PKI. In their architecture, they provide incentives to Certification Authorities (CAs) with a correct behavior while punishing those with inappropriate practices automatically. While this solution offers a flexible and robust infrastructure for PKI, it lacks the specific focus on enabling smart contracts to actively engage in the public key infrastructure.

Ankush S. and Elisa B. [27] propose a system where the hash of the certificate is stored in a smart contract together with the ID of the device. In this solution, when an entity needs to check the veracity of a received certificate, she/he asks the smart contract if the hash of the certificate is

reliable or not. The advantage of this system lies in the dynamic revocation and addition of certificates.

Alexander Y. et al., [28] propose a system to implement the classic chain of trust in Solidity (programming language of Ethereum's Smart contracts). Each CA has its own smart contract where it uploads its certificate and stores the hashes of the certificates it issues. On the other hand, it requires modifying the X.509 standard with some minor additions.

All these systems require the use of X.509 certificates to identify the devices. These certificates are heavy (Google's certificate is 1.13 KB) and are complex to process in Solidity because absence core libraries for string manipulation [28]. The certificates would need to go along with each transaction to identify the device and be verified by the smart contract itself in each transaction which would increase the cost in each transaction.

To avoid this, we propose a system without certificates using what we define as Smart Certificate Authority (SCA). The SCA is deployed as a smart contract that checks if an entity meets certain requirements, and if so, instead of delivering a certificate, it simply stores the address of the entity along with its attributes. Due to the qualities of smart contracts, if an entity has been identified and authenticated by a smart contract, this process is trusted by the rest of the Blockchain and does not have to be repeated (verify once, authenticate any-when). As a result, any smart contract that wants to authenticate an entity simply has to query the SCA if the identity is stored, avoiding duplicate certificate verification in each communication.

6.2.3. IoT in Blockchain

In this section we discuss those papers that use IoT as a service for Blockchain and how they solve the identification problems posed by IoT.

Little information on this topic can be found in the SoA, as highlighted by the study conducted by Mohamed Laarabi in [29] in March 2022. In their study only two articles are detailed with scenarios where smart contracts receive data gathered from sensors, [30, 31].

In [30], the authors propose a system for managing the energy consumption of IoT actuators based on the measurements received by the IoT sensor. In this work they do not propose the identification method of the data in the smart contracts, but store the public credentials and signatures along with the data in Blockchain, and leave the actuators as responsible for somehow identifying the data. Thus, they use the Blockchain as a database. In our work, we have developed an infrastructure in which the smart contract itself identifies the senders.

The main of Carlos Molina-Jimenez *et al.* in [31] is highlighting that conventional business contracts can be automated using centralized applications, decentralized applications or combining both. Also, they focused

in the complexity of the last one. The work is presented using the example of selling to a customer Alice's personal information obtained through her IoT sensors. Data security, however, is not covered within the scope of this project.

Another recent work is the one proposed by Mohamed Ahmed *et al.* [13], which is focused on finding, defining, and proposing systems for measuring the quality dimensions relevant for IoT Data Qualification. This work presents the context of the medical equipment cold chain where IoT nodes provide the smart contracts with qualification data. It is the same use case where we will present our work, detailed in Section 6.4. In this use case, Mohamed Ahmed *et al.* define four main data quality dimensions: accuracy, completeness, consistency, and currentness. Also, they propose a method to calculate them. But as they recognize, the IoT data sources' security is a field that is not embarked upon in their work and yet must be addressed. This is where our research comes in, i.e., ensuring the non-manipulation of devices or their messages. However, it is outside of our scope to evaluate the quality dimensions of the messages. So we consider that this work has a great synergy with ours.

6.2.4. Oracles

In this subsection, we will examine the research conducted on uploading trustworthy information to the Blockchain. The purpose of this effort is to enable smart contracts to depend on this data, ensuring stakeholders can confidently execute high-impact tasks.

There are several Oracles designed to upload specific information that are excluded in the analysis since they cannot be directly applied to IoT like PriceGeth [32] used to publish price pairs or Augur [33] for market prediction.

Some of the works of the SoA found in that field propose servers or clients to feed smart contracts, never directly from IoT nodes, e.i. the smart contract never verifies the signature performed by the IoT device (edge-to-edge signature). Moreover, just one of them considers the integrity of IoT nodes.

In [34], they propose a system to feed smart contracts with information from reliable web pages using HyperText Transfer Protocol Secure (HTTPS), assuming that if these web pages are reliable for non-Blockchain applications with high impact, Blockchain applications can also use them. The system is called Town Crier. In this system, Intel SGX is used to guarantee the correct operation of the Oracle, which is not responsible for the reliability of the data, but the correct data source. The correctness of the data is guaranteed under the assumption of the validity of the data source, the reliable web pages. In their paper, DOAu, DOTa and DOI are guaranteed through the use of off-chain TLS certificates and Intel SGX-based remote attestation.

In the case of DOTu, it is achieved through the confidence that resides in the websites, and finally, DOF is provided using SGX clock and a public timestamp verification. This infrastructure is complete, but it can not be applied to IoT, since DOAu and DOTu are achieved by the general knowledge and trust in the data generators, the websites, which cannot be applied to IoT. The same problem is found in Chainlink [35].

DiOr-SGX [19] has similarities to Town Crier [34] because it uses Intel SGX to ensure the correct functioning of the Oracle, but differs by creating a decentralized system to ensure availability and adds a voting system and prestige rewards to choose the leader of the Oracles with the best response time. In this system, the smart contract generates an event to request data. Then, this event is read by the Oracle leader which requests the data from other Oracles (Oracle Nodes). These Oracles collect data from IoT nodes, and send it to the leader along with proof of their correct operation through Intel SGX. This system is focused on promoting the best self-organization to acquire the shortest response time. Also the leader performs a remote attestation process on the other Oracles to make sure that they did not manipulate the data. But nobody verifies the leader, and it does not provide any mechanism to ensure the veracity of the data i.e., it does not offer any mechanism for DOAu, DOTa, DOI, DOTu, or DOF because although there is a mechanism to check the correct status of the Oracles nodes, none identifies the origin of the data received by Oracle Nodes. On the other hand, there is no penalty mechanism for those who deliver data different from the average. Finally, this system, due to its decentralized nature of distributed data collection, where many nodes shall obtain the same data from different sources, can be applied in use cases such as the temperature of a city, but can hardly be applied to a cold chain where all the nodes that measure the temperature belong to the same entity.

Astraea [36] is a mechanism for contributing binary information (true or false) to the Blockchain. The information is provided through a system of voting and certification. All "players" have to contribute an amount of money to vote or certify and they lose money or are rewarded according to the data provided, which motivates them to behave honestly. The problem with this system is that it is only applicable to decidable and verifiable information that is accessible to a high number of players from different sources. However, this condition is not applicable in all scenarios. The same problem is found on other Oracles based in a reputation system [37].

In Jonathan Heiss' work [20], they propose two different systems to gather signed data from a sensor, through a gateway that processes and sends it to the Blockchain. The smart contract itself can check the correct processing of the incoming data using ZoKrates in the first solution and Intel SGX based remote attestation in the second solution. In both of them, the gateways process the IoT data, considering the verification of the IoT signature as part

of the data processing. Then, the smart contract attest the correctness of the data processing, and because the IoT signature verification is included in the data processing, the smart contract is indirectly verifying the IoT signature. This system provides DOAu, DOTa and DOI. However, both mechanics require a trusted and critical setup that is not explained in their proposal. They assume the existence of a trusted setup in every enrollment that can be verified by each stakeholder, making its real implementation very complex. Moreover, there is not process to probe the non-manipulation of the IoT node (DOTu). Finally, with not further details, the gateway accepts any signed data from the sensor, so old signed data would be accepted (DOF).

The work of Alia Al Sadawi *et al.* [38] is the only work in the SoA that claims being the first and only detailing a entire process of integrating IoT in Blockchain. They do it through the use of a hardware oracle with cryptographically attestable and anti-tampering properties. This secure IoT device measures CO2 levels and signs the outgoing data with a nonce. The information is sent to a fetching script that write it on the Blockchain through a transaction. At the end of the document, the authors perform a detailed security and vulnerability analyses to ensure the robustness of the smart contracts but not of the full system. Additionally, there are not details for a public attestation procedure of the hardware Oracle, so there is not a mechanism for proving to the infrastructure the trustworthiness of the attestable IoT device (DOTu). On the other hand, they do not provide details of any PKI, therefore there is not DOAu. Additionally, the measured data passes through a fetching script (e.g. a Python script) that sends it to the Blockchain, which the owner or an attacker could manipulate to send any arbitrary data losing DOI. Finally, even using a nonce to avoid digital signature repetition, the data could have been gathered and signed at any moment being vulnerable to delay attacks (DOF).

Even though hardware Oracle is a known category [9, 39], with their own standard and qualification analysis [14, 13], included in surveys as [11, 10], for the best of our knowledge, there is just one implementation in the SoA which lacks in some important details like the PKI and does not meet several security requirements. Our paper presents the only infrastructure capable of providing IoT-generated data directly to the smart contract with edge-to-edge signature, where the Blockchain can verify DOAu, DOTa, DOI, DOTu and DOF, with a dynamic enrollment process and applicable to Ethereum.

6.3. Background

6.3.1. Secure sensor

Dominic Pirker *et al.* [21] presented four novel solutions for achieving unquestionable trust in the measurements done by an IoT device. We will

consider their solution “Concept A” for our work, and in the following, it will be referred as Secure Sensor (SS). In the SS we have to differentiate three different elements:

- Controller: it is the core of IoT node itself that through a Turing-machine can perform any task.
- Sensor: it is the hardware extension connected physically to the controller that through SPI, I2C or buses receives command and sends the measured data.
- Hardware Security Module (HSM): it is a hardware module secured by design with the capacity to create key pairs, storing and using them.

Thus, SS is an IoT device with a controller, a sensor and a HSM. The distinctiveness of the SS from other IoT device architectures is the fact that the controller cannot communicate directly with the sensor but the communication is done through the HSM. As it is shown in Fig. 6.1, the controller, through a limited API can interact with the HSM, which is in charge of gathering the data from the sensor, signing using a nonce, and forwarding it to the controller together with the digital signature. The private key used for this digital signature is a sealed key, which means that it cannot be used for any other purpose. Because the element that generates the data is hardware-protected (shown green in Fig. 6.1) this device provides DOTu.

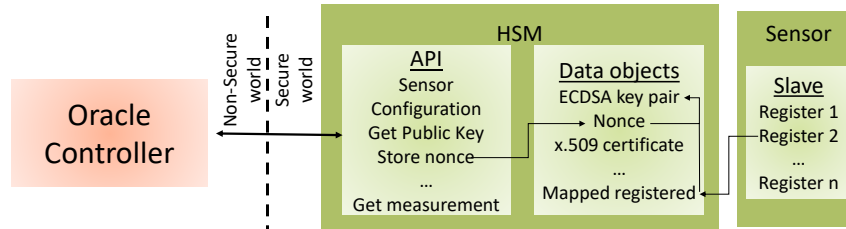


Figure 6.1: High level schematic of Secure Sensor.

The downside of this work is the complexity of distributing the public keys necessary for DOAu, and implementing a verifiable random nonce for introducing the DOF into the measurements.

6.3.2. Ethereum Addresses

Ethereum is the most popular Blockchain for IoT applications, and also for smart contracts in general [7]. Ethereum uses Elliptic Curve Digital Signature

Algorithm (ECDSA) with a 256-bit long private key and consequently a 512-bit public key. The Ethereum address associated with this private key is composed of the last 160 bits of the Keccak [40] hash of the public key. Therefore, in the following a key pair has the next elements: private key (Priv), public key (Pub) and Address. In this way, from a signature or a public key, the address can be easily derived, and we do not have to store 512-bit long public keys. Thus, in our approach developed over Ethereum platform, we use the Ethereum addresses as identifiers of the entities. Following Hilarie Orman's words [41]: "Who am I? you are your Blockchain address".

6.3.3. Blockchain and smart contracts

As it was commented in the Section I, Blockchain is a system developed by Shatosi Nakamoto in Bitcoin in 2008, as a distributed consensual peer-to-peer database [1], a perfect environment where including "smart contracts". It is a concept defined by Nick Szabo in 1997 [42] to formalize and secure relationships over computer networks. But it was not until 2014, where the smart contracts were implemented in Ethereum [43], allowing to execute scripts in a public Blockchain similar to Bitcoin that satisfies the definition of Nick Szabo, thus implementing real smart contracts.

Data in Ethereum are organized in blocks. The nodes add new blocks to update the database without deleting previous ones. Every block is known as the "father" of the next one. The time interval between blocks generation is called block-time. The consensus protocol defines the entity that adds new blocks to the chain. Bitcoin's consensus protocol is Proof of Work (PoW)[44], just like Ethereum's initially. In PoW, each new block proposes a mathematical problem that takes an average time equal to the block-time to be solved. The first node to solve the problem (the miner) publishes the new block where it includes the solution and the time at which the generation of the block started (the block timestamp). In PoW, the miner has some freedom in setting the block timestamp, which makes the block timestamp unreliable [45, 46].

In September 2022, Ethereum migrated to a different consensus protocol: Proof of Stake (PoS). This migration is known as The Merge [47]. This consensus protocol requires staking the crypto-coin of Ethereum, Ether. The entities staking Ether can propose and validate blocks are called validators. The validators are randomly selected using a random number to be a block proposer in every slot. This random number is generated by "The Beacon Chain"[48] through a decentralized system called Randao [49]. In this new consensus protocol, blocks can be added in 12-second slots, and the block timestamp is strictly defined by the slot in which it is published, which eliminates the validator's freedom to alter it. However, slots can be empty if the block proposer does not propose the block on time. Therefore, the block number does not determine the timestamp, but the slot number does.

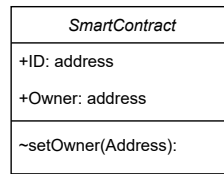


Figure 6.2: Abstract class "Smart contract" following the Unified Modeling Language.

Smart contracts executed on PoS provide the confidence of knowing that 2/3 of the network has validated its execution. However, the functionality of smart contracts remains the same to ensure backward compatibility. Smart contracts have addresses as identifiers, like the users. A widespread structure in smart contracts, which is followed in this work, is the definition of a user with special privileges that allows him/her to modify specific settings and data. This user is called the Owner and can, among other things, change the Owner of the contract. Fig. 6.2 shows the abstraction diagram of our smart contract following the Unified Modeling Language (UML) where the symbol \sim denotes that the method is only accessible to the Owner.

All requests to the smart contract are made through signed transactions [50]. They trigger operational logic on the smart contract with a computation cost. This computation cost is measured in *gas* [51]. Because they are executed in a huge number of nodes simultaneously, the gas is very expensive, so smart contracts should avoid high-cost computation tasks. Multiple transactions are grouped in a block that is then stored on the Blockchain. All transactions follow the same structure and the relevant fields for this work are:

- Raw transaction:
 - Sender: Address of the transactions signer.
 - Addressee: Address of the transaction recipient.
 - Data: Name of the calling functions and variables.

- Signature: Signature of the raw transaction.

6.3.4. Assumptions

In order for SS to work, several assumptions are made, which are detailed at the following:

- **Trusted manufacturer:** The manufacturer of the SS is well-known and public. In this way, it can certify the correct manufacturing of the device. It is a common assumption in HSMs e.g. the endorsement certificate in the Trusted Platform Module 2.0 standard for crypto-processors [52].
- **Trusted smart contract:** The smart contracts shall be free of bugs and verified by all the stakeholders before and after they are deployed in the Blockchain.
- **Not undetectable attacks to SS:** The HSM in the SS shall follow a security-by-design approach, avoiding any kind of software attack. On the other side, any physical assault would leave the device with visible damage, rendering it useless.

It is unnecessary to assume an invulnerable or reliable SS' microcontroller. Our solution will not be at risk even if an attacker can control it fully.

6.4. Use case: ensuring respect of the cold chain through smart contracts

As in [13], we present our proposal in the scenario of a cold chain, where the transported goods must maintain strict temperature conditions. In this scenario, the correct fulfillment of these conditions is an essential value of the product value. Furthermore, in the cited work, due to its proximity to the real business, the authors are provided with the actual strict temperature conditions of a medical product for blood testing. The non-accomplishment of this compliance requirement could lead to a breakage of the product. Therefore, in this scenario, not only the product distributor is responsible for the product's quality but also the complete supply chain.

There are at least four stakeholders in our scenario: the shipper (the originator of the transport request), the carrier, the receiver, and the IoT manufacturer (in charge of manufacturing the temperature sensors). The transported goods have temperature sensors with internet access (IoT nodes).

Then, the receiver requests a product with quality requirements. The shipper accepts the request by offering a product that meets the rates if it stays within the threshold temperature during transport. The carrier accepts the thresholds. Finally, the three agree on the penalties for infringement and choose a manufacturer for the IoT nodes.

However, if there is not enough confidence in the reliability of the system, there is no interest in the infrastructure. The following are the risks we identified in the use case:

- Sensors replacement with other IoT devices which could generate invalid data (DOAu).
- Origin of all valid data has to be identified to the unique data generator (DOTa).
- Manipulation of data collected by the sensors (DOI).
- Software manipulation of the Sensors (DOTu).
- Time modification on which IoT nodes collected the measurements (DOF).

In summary, such a use case essentially requires DOAu, DOTa, DOI, DOTu, and DOF of the IoT nodes so that the stakeholder can trust the system and they can set a smart contract for the enforceable agreement.

6.5. Design of the proposed system

This section firstly explains a high level view of our solution in the subsection 6.5.1. Secondly, the details of the PKI required for the system are explained in the subsection 6.5.2; next, the proposed solution to guarantee the freshness of the measurements is detailed in the subsection 6.5.3 and finally, the final process is fully presented in the subsection 6.5.5.

6.5.1. Proposed system

The process starts with a setup phase where the stakeholders agree on the cold-chain conditions by generating the qualification smart contract. In this smart contract, the stakeholders stipulate who will be the manufacturer of the sensors, the sensor model to be used and other legal information about the sensor (recalibration digital certificate or digital certificate by accreditation institution [14]). Then, they generate the SCA smart contract. The framework is used in the manner depicted in Fig. 6.3.

1. First, the sensor manufacturer generates, signs, and delivers a certificate to each device manufactured; this is the manufacturer certificate.
2. The shipper receives the sensor and prepares the package with the device, which will have to maintain a specific temperature throughout the entire cold chain. Then, the shipper pre-registers the package at the Blockchain with the package ID and the SS address.

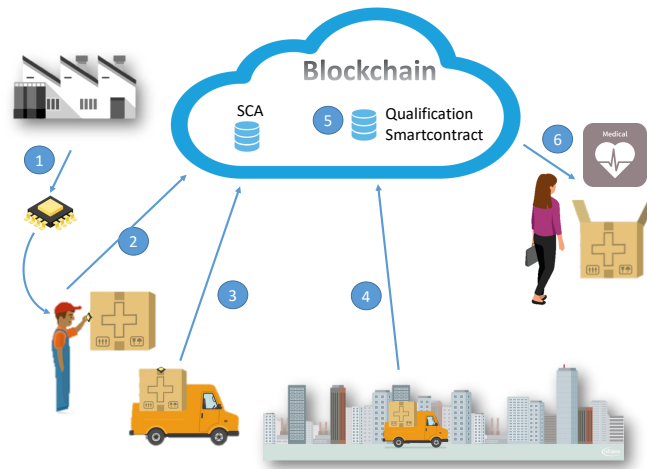


Figure 6.3: High-level scheme of a SS installed in a package of a supply chain sending data to the Blockchain.

3. The sensor then asks to be publicly identified in the SCA, creating a transaction that includes the manufacturer certificate. The SCA then initiates the verification process with a moderate gas cost. It will check that the identification request and the manufacturer certificate meets all the requirements, and if the request is valid, the SCA will store the SS address along with important information data about the sensor. Therefore, if an address is stored in the smart contract, means it passes successfully through the verification. This completes the registration in the PKI needed for DOAu.
4. When the SS uploads a data package to the Blockchain, it will first read a recently published nonce, explained with more detail in Section 6.5.3. Then the SS will sign the measured data together with the nonce. Data and signatures are added inside the transaction which is then signed again and sent to the Qualification Smart Contract (QSC).
5. Upon receipt of the transaction, the QSC will check that the KeyPair that signed the transaction has its address stored in the SCA (obtaining DOAu). If yes, it will verify other elements: the SS was used to sign the data (DOI and DOTu), and the nonce included in the data signed is fresh (DOF). There is no need to verify any certificate in this step.
6. Finally, the receiver, when receiving the package, reads the address of the sensor in the package and looks for the package's qualification data in the Blockchain.

6.5.2. PKI

The goal is to make smart contracts capable of recognizing the origin of the received data. In our scenario, these data come from SS, presented in the background.

As it was detailed in Section 6.3.1, the SS has a particular architecture that was shown in Fig 6.1, where three modules were distinguished: controller, HSM, and sensor. The device can create KeyPair with special features, the Secure-element KeyPair (SeKP). The Priv is always stored in the HSM and can only be used to sign data coming from the sensor and a nonce delivered from the microcontroller. With this signature verification, anyone can verify the DOTu and DOI of the signed data.

The device will interact with the Blockchain by sending and signing its transactions. The transaction structure is generated in the controller, which performs the hash and sends it to the HSM that signs it. However, due to the previously explained security limitations of SeKP, the SS cannot use it to sign Blockchain transactions which is essential to be able to interact with the Blockchain, so a second KeyPair is needed to sign them. This second KeyPair, without limitations, is called Owned KeyPair (OKP) and it is used exclusively to sign transactions on the Blockchain providing DOAu.

Ideally, only SeKP would be necessary, as it could also be used to provide DOAu, since verifying this signature allows to know who created the data. Nevertheless, its characteristics limit an actual implementation, as it cannot be used in the protocols of different scenarios e.g. SS could not have its own Blockchain address nor interact with TLS. In our case, the fact that SS has its own address in the Blockchain (which can only be achieved using OKP) facilitates the transparency and traceability of all transactions of the device in the Blockchain. Finally, these two KeyPairs are linked to each other and to the manufacturer through a proprietary certificate, called manufacturer certificate.

The manufacturer certificate contains the device model (ModelDevice), SeKP.Pub, OKP.Pub, and its signature. The manufacturer will keep the address of its signing key (ManKP.Address) updated in a smart contract, the Manufacturer smart contract (ManSC), shown in Fig. 6.4. The ManKP.Address can be updated only by the owner of ManSC.

The final part of our Blockchain-based PKI is the Smart Certificate Authority. It is a smart certificate that will transparently verify the Manufacturer's certificates and other information about the sensor when it is shipped in the package. It will check the requestor's information upon receiving a transaction. If it meets the requirements, the SCA automatically stores a copy of the validated addresses, OKP.address and the SeKP.address in the smart contract if and only if the requestor satisfies all the requirements. This mechanism has several advantages versus the classic certificate system:

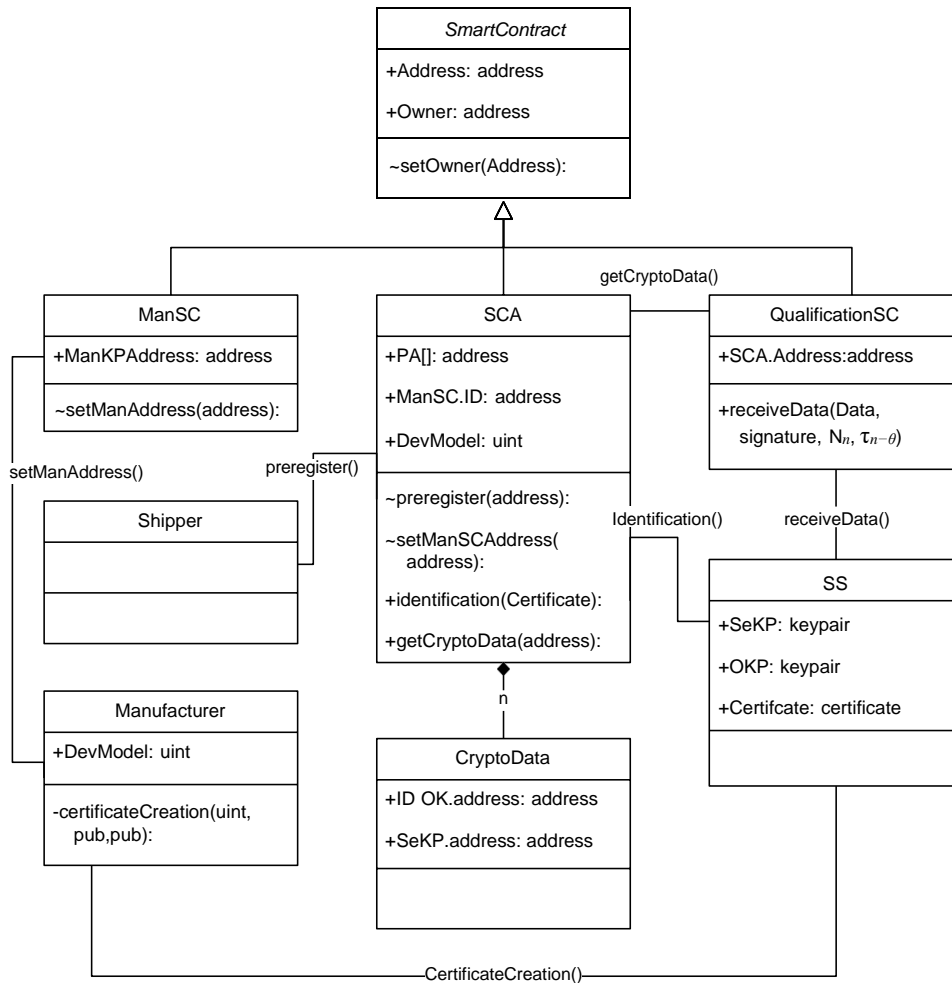


Figure 6.4: Class diagram of the complete infrastructure following the UML.

- Because of the smart contracts' features, they are as reliable as a certificate signed and validated by all the Blockchain infrastructure following the SCA's stipulated rules.
- Any entity with Blockchain access can verify an identity, including the smart contracts themselves.
- Because of Blockchain decentralization, this method has a very high availability.
- There is no need to keep an updated revocation list because the address stored in the smart contract can be dynamically removed.
- There is no need to verify a certificate because the response of SCA is always trusted. It reduces the computer processing consumption which

is essential in smart contracts.

The SCA receiving a certification request will check that:

1. The manufacturer certificate was signed by the manufacturer.
2. OKP.pub was preregistered by the shipper.
3. The model device (IDmodel) of the Secure Sensor is the one selected in the setup phase.

6.5.3. Freshness

This subsection will detail the method designed to guarantee the freshness of the actual data. As explained in subsection 6.3.1, SS includes a nonce in the signature when it gathers data. To guarantee the data freshness, the nonce must be unknown until it becomes publicly known at a time τ_l . When the actual data i is made public at a time τ_r , including in the signature the nonce, it is guaranteed that the data was generated in the uncertainty interval $\Delta\tau_i$:

$$\Delta\tau_i = \tau_r - \tau_l \quad (6.1)$$

In our infrastructure we use the blockhash as nonce. In the Blockchain Ethereum PoS, the blocks can be published in every slot. There is a slot every 12 seconds, which is called blocktime ($\Delta\tau_b$). The blockhash is made through the hash of the data in the block. One element that forms the block data is a random variable called RANDAO mix (Rm_n). Rm_n is included as part of the blockhash creation replacing the available variable in the blocks called *mixHash*, which will be deprecated after The Merge [53]. In PoS, the blockhash of the blocks cannot be considered random anymore because the proposer can create many blocks internally and publish the one that suits him/her. This means that smart contracts cannot use the blockhash for use cases such as lotteries and instead, they have to use directly Rm_n . However, the blockhash can still be used as an unknown number generator as Rm_n due to the properties of the hash operation (even if the proposer tries generating several blocks, all the possible blockhash resulting are unknown until the moment Rm_n is revealed). Therefore, in this section we will present an analysis of the variable Rm_n because its reveal time has the same uncertainty interval as the blockhash.

Considering Rm_n published at block number N_n at the slot n with a timestamp τ_n , it can be publicly calculated in the moment its parent block is published. Normally the parent block $N_n - 1$ is published in the previous slot, at slot $n - 1$ i.e. at τ_{n-1} , but as explained in Section 6.3.3 slots can be empty if the proposer does not propose the block on time. So, we define

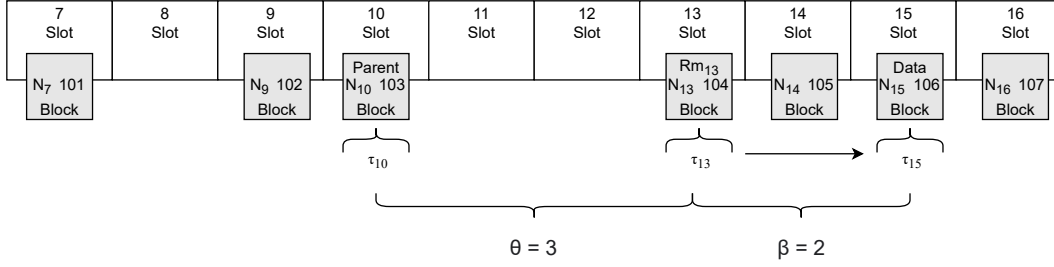


Figure 6.5: Graphic representation of the variables in (6.2).

θ as the difference of the slots between the slot containing the block N_n and the slot containing its parent block $N_n - 1$. That means that Rm_n is revealed at time $\tau_{n-\theta}$ (i.e. $R(Rm_n) = \tau_{n-\theta}$). Therefore, using the blockhash of block N_n at slot n , as nonce in SS when gathering data would mean that $\tau_l = \tau_{n-\theta}$. Inserting it in a block at slot $n + \beta$, where $\beta \in \mathbb{N} > 0$, would leave $\tau_r = \tau_{n+\beta}$, getting a uncertainty interval defined in (6.2). Fig. 6.5 shows a practice example of this equation where a SS uses the blockhash of block 104 as nonce to sign gathered data. Then, the data signed is inserted in block 106. The parent of block 104, block 103, is inserted in the slot 10.

$$\Delta\tau_i = \tau_{n+\beta} - R(Rm_n) = \tau_{n+\beta} - \tau_{n-\theta} = (\theta + \beta)\Delta\tau_b \quad (6.2)$$

However this uncertainty interval is insecure because Rm_n is known in advance by the block proposer of the slot n . Coordination between the carrier and the proposer can lead to a timestamp attack allowing the use of a measurement gathered a time $\Delta\tau_A$ before the reveal of Rm_n , what we call the PrevTime Attack (PTA). The proposer can be elected for several blocks in a row, increasing $\Delta\tau_A$. Also, there can be accidental empty slots which would help to predict Rm_n with a probability of ξ . The probability of being a proposer depends on the amount of money staked in the infrastructure. Being μ the probability of the attacker to be chosen as the proposer of the next block, the probability of knowing Rm_n with a time τ_A in advance is equal to:

$$Pr(\Delta\tau_A) = (\mu + \xi)^{\lceil \frac{\Delta\tau_A}{\Delta\tau_b} \rceil} \quad (6.3)$$

For an attacker investing six billion dollars, 16,3% of all the Ether staked and a 2,9% of the total ether supply at 22/07/2023,[54], $\mu = 0,163$ [55]. Also, between 22/07/2023 and 15/04/2023, 1.3% of the slots were empty slots, $\xi = 0,013$. With this values, $Pr(48) = 0,0009$. With sufficiently low probability, $\Delta\tau_A$ can be infinite. Assuming that the carrier always succeeds in performing PTA by obtaining an assumable time $\Delta\tau_{AA}$ such that $\exists \gamma \in \mathbb{N} : \Delta\tau_{AA} = \gamma \cdot \Delta\tau_b$. The new minimum uncertainty interval is:

$$\Delta\tau_i = \tau_{n+\beta} - (R(Rm_n) - \Delta\tau_{AA}) = (\theta + \beta + \gamma)\Delta\tau_b \quad (6.4)$$

Where $\tau_l = \tau_{n-1} - \Delta\tau_{AA}$ and $\tau_r = \tau_{n+\beta}$. Secondly, we assume that the carrier always tries to avoid sending a faulty measurement by using a measurement taken $\Delta\tau_{NA}$ time longer than $\Delta\tau_{AA}$, such that $\Delta\tau_{NA} \in \mathbb{R} > 0, \Delta\tau_A = \Delta\tau_{NA} + \Delta\tau_{AA}$.

$$Pr(\Delta\tau_{NA}) = (\mu + \xi)^{\lceil \frac{\Delta\tau_A}{\Delta\tau_b} \rceil} = (\mu + \xi)^{\lceil \frac{\Delta\tau_{NA}}{\Delta\tau_b} \rceil + \gamma} \quad (6.5)$$

Thus, with $\mu = 0,163$, $\xi = 0,013$ and $\gamma = 4$ (48 seconds of $\Delta\tau_{AA}$), in order to perform the simpler 12-second attack of $\Delta\tau_{NA}$, $Pr(12) = 0,0002$. Although low, this probability is still too high to ignore, but easily indemnizable. To compensate for the probability of 0.02% of performing a successful PTA, each time the carrier inserts an incorrect measurement in the smartcontract, it is considered to have attempted an unsuccessful PTA. Then, it shall pay an additional penalty for those times it was successful, equivalent to the 0.02% of the package price.

Finally, with this mechanism, QualificationSC can estimate a highly reliable uncertainty interval of the timestamp of the measurements. Each time a SS sends measurements using a blockhash as nonce, the smartcontract will get the timestamp of the father's block used as blockhash ($\tau_{n-\theta}$), and subtract 48 seconds to it and estimate that the measurement was generated in some moment between the calculated time and the current time with a probability of 99.98%. Still, this solution has a drawback, smart contracts in Ethereum, and thus in the majority of Blockchains, cannot access to the timestamp of previous blocks, and so, it cannot access $\tau_{n-\theta}$. To solve this problem and avoid using Oracles to provide this data, we developed a novel optimistic approach that is explained in the next section.

6.5.4. Inserting reliable information of previous blocks to a smartcontract

During the execution, a smart contract can access to the current time, which in PoS is accurate. Additionally, a smart contract can access the blockhash of the last 256 blocks but, it cannot collect any additional data about these blocks like the timestamp. The timestamp of previous blocks cannot be derived using the block numbers because even if the blocks are published in 12 second slots, some slots can be empty without a proposed block, therefore, between consecutive blocks, the time gap can be higher than 12 seconds. An attacker investing six billion dollars could easily exploit it to sign at measure at block N_n and send it at block $N_n + k$ with a real-time gap major than $k * \Delta\tau_b$.

Nevertheless, a smart contract can recreate the blockhash of the block N_n on the execution if all the needed data is provided. All the variables that

make up a blockhash are: ParentHash, UncleHash, Coinbase, Root, TxHash, ReceiptHash, Bloom, Difficulty, Number, GasLimit, GasUsed, Time, Extra, MixDigest and Nonce. Using all of these variables and comparing the resulting hash with the blockhash N_n collected inside the execution, a smart contract can rely on the provided data, that means, in the timestamp. The problem with this method is that the verification process is highly gas-consuming (221570 gas). To reduce the gas taxes, we go through an optimistic approach similar to the one used in the optimistic rollups [56]. In this approach, all the functions necessary to verify the results of a call are integrated in the smart contract, but this verification is not executed as a general rule to reduce costs. When a function of a smart contract is called externally, the caller directly provides the result and it is considered valid without going through further on-chain verification. Then, a time is given for anyone to verify the result off-chain and to denounce the invalidity of the provided value. If this occurs, the smart contract itself verifies the result, reverses the transaction if necessary and performs the stipulated penalties.

By implementing this approach in our smart contract QualificationSC, SS itself can provide the timestamp of the parent of block N_n , which blockhash was used as nonce in the signing process, where n is the slot from where the blockhash was gathered. Its parent block was published at slot $n - \theta$. The smart contract relies at first in this value using it to calculate $\Delta\tau_i$. Then, a time of 3 minutes (15 slots) is provided for any claimer to claim the invalidity of the timestamp provided by SS and propose a new one. If someone does, the person in charge of the sensor (SensorResponsible) can accept the new timestamp without the necessity of reconstructing the blockhash, getting a very reduced penalty. If the SensorResponsible refuses the new timestamp, the claimer can process a "judgment" providing all the needed information to the smart contract, so it can recreate the blockhash of $N_{n-\theta}$. If the smart contract can successfully recreate the blockhash, meaning that the new timestamp proposed by the claimer was correct, the SensorResponsible has to pay all the expenses transactions and a small penalty. The judgment is a very unlikely call, because the SensorResponsible will accept the new timestamp proposed by the claimer if it is correct without the need to go through the judgment process. The judgment costs 240510 gas, equivalent to less than \$10 at 24/07/2023.

With this solution, the SS itself can send the timestamp of the parent's block, which blockhash was used as a nonce to QualificationSC. Then, the smart contract can trust it without increasing the on-chain costs.

6.5.5. Detailed process

In this subsection, we will include a detailed explanation of the PKI in the cold chain scenario.

In the setup phase, the stakeholders must detail the characteristics of

Algorithm 1 Certificate Creation

Internal Inputs MaK_priv
External Inputs $OKP.pub, SeKP.pub, DevModel$
 $Cert_{raw} \leftarrow (DevModel || OKP.pub || SeKP.pub)$
 $Signat \leftarrow sign(MaK_priv, Cert_{raw})$
 $Cert \leftarrow (Cert_{raw} || Signat)$
return $Cert$

the cold chain. They stipulate the manufacturer, the SS model ($DevModel$), sensor certificates [14], the assumable time $\Delta\tau_{AA}$ and the qualification requirements. With these data, they can deploy the smart contracts SCA and QualificationSC. Next, the manufacturer deploys its own smart contract, the manufacturer smart contract (ManSC), where it dynamically updates its key used (ManKP) to sign the manufacturer certificates (Cert). Any entity (including smart contracts) can consult the address of the manufacturer in ManSC.

After the setup phase, the process sequence starts. The class diagram and the sequence diagram of the infrastructure can be found in Fig. 6.4 and Fig. 6.6, respectively.

6.5.5.1. CertificateCreation(Manufacturer \rightarrow SS)

The first phase of the sequence is Certification Creation in which, once SS is manufactured, the manufacturer reads its public keys $SeKP.pub$ and $OKP.pub$ and creates a certificate including $DevModel$. Next, the raw certificate is signed using the manufacturer's private key $MK.priv$. Finally, the signature is attached to the raw certificate, creating the certificate. It is the only phase that must be performed in a controlled environment and is detailed in Algorithm 1.

6.5.5.2. Preregistration(Shipper \rightarrow SCA)

The SS device is sent to the shipper, who installs it in the package to be shipped and prepares it to start the cold chain. She/he preregisters $OKP.address$ in SCA, adding it to a sheet of Preregistered Addresses (PA).

6.5.5.3. Identification(SS \rightarrow SCA)

Then, SS requests are identified in SCA using its certificate. SS uses its $OKP.priv$ to sign the Blockchain transaction and when SCA receives it, the smart contract first checks that the transaction sender (Tx.sender) is in the PA. Secondly, it validates that the certificate was signed with the address indicated in ManSC. Next, it asserts that the model defined in the

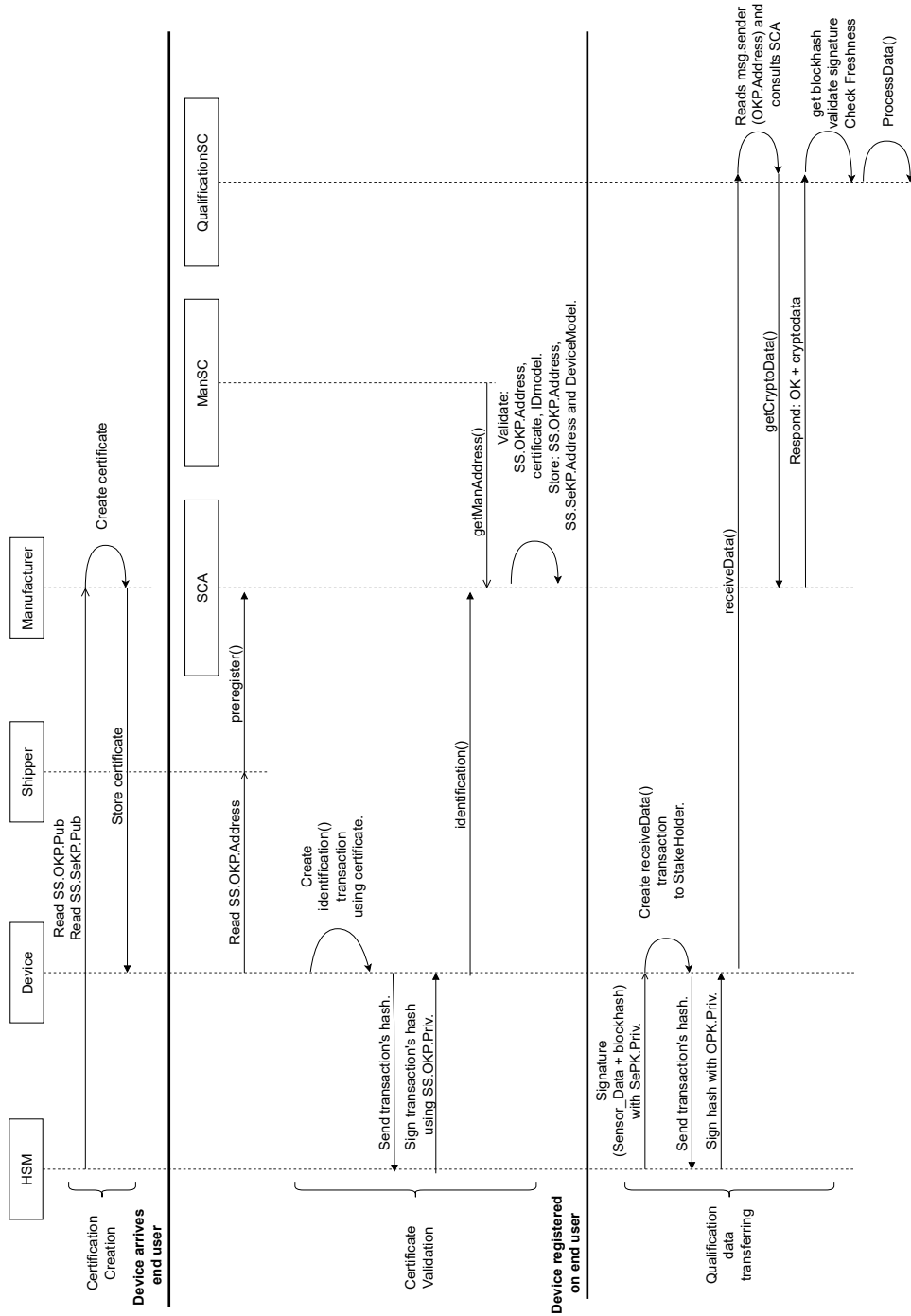


Figure 6.6: Sequence diagram of the complete infrastructure following the UML.

Algorithm 2 SCA identification

```

Internal Inputs  $DevModel$ 
External Inputs  $Tx.sender, Cert$ 
  if  $!(Tx.sender \subset PA)$ 
    return No valid
   $Cert_{raw}, Signat \leftarrow Cert$ 
   $Address_{signer} \leftarrow recoverAddress(Cert_{raw}, Signat)$ 
   $ManKP.Address \leftarrow getManKP.Address()$ 
  if  $(Address_{signer} \neq ManKP.Address)$ 
    return No valid
  if  $(Cert.DevModel \neq DevModel)$ 
    return No valid
   $OKP.Address \leftarrow getAddress(Cert.OKP.pub)$ 
  if  $(Tx.sender \neq OKP.Address)$ 
    return No valid
   $SeKP.Address \leftarrow getAddress(Cert.SeKP.pub)$ 
   $addCryptodata(OKP.Address, SeKP.Address)$ 
return

```

certificate is the same model defined in SCA. Finally, the smart contract verifies that the TX.sender is the owner of the certificate and, if everything was correct, SCA will store OKP.Address linked to the SeK.address defined in the certificate. It is not needed to show ownership SeK.priv because the manufacturer certificate is proof enough that the owner of OK.priv is the only owner of SeK.priv. The pseudocode of the function *identification()* is found in Algorithm 2.

6.5.5.4. ReceiveData(SS \rightarrow QualificationSC)

When the SS'certificate is validated, it can start transferring qualification data. Firstly, it will read the last blockhash at slot n and will provide it to the Hardware Security Module (HSM) as nonce. The HSM gathers real data from the sensor and signs it using the SeKP.priv together with the nonce. Then, the HSM sends the result to the Oracle Controller. The last one generates the transaction *receiveData()* in Algorithm 3 with the inputs: real data, the signature, the number N_n and, the timestamp of the parent block of N_n , $\tau_{n-\theta}$. The SS signs the transaction with OKP.priv and sends it to QualificationSC.

QualificationSC receives the transaction, asserts its authenticity checking the sender address (OKP.address) in the SCA, and obtaining SeKP.address from it. From N_n , the smart contract gets the blockhash used as nonce and verifies the SeKP signature. Finally using the time $\tau_{n-\theta}$, the time of the current block $tau_{n+\beta}$ and $\Delta\tau_{AA}$, defined in the setup phase, the

Algorithm 3 QualificationSC receiveData

```

Internal Inputs  $T_{AA}$ 
External Inputs  $data, Signat, N_n, \tau_{n-\theta}$ 
   $SeKP.address \leftarrow getCryptoData(Tx.sender)$ 
  if  $!(SeKP.address)$ 
    return No valid
   $nonce \leftarrow blockhash(N_n)$ 
   $Content \leftarrow (data||nonce)$ 
   $Address_{signer} \leftarrow recoverAddress(Content, Signat)$ 
  if  $(Address_{signer} \neq SeKP.address)$ 
    return No valid
   $\tau_{n+\beta} \leftarrow currenttime$ 
   $\Delta\tau_i = \tau_{n+\beta} - (\tau_{n-\theta} - \Delta\tau_{AA})$ 
   $processData(Data, \Delta\tau_i)$ 
return

```

QualificationSC can calculate uncertainty interval $\Delta\tau_i$ (6.4), estimate when the measurement was gathered with high reliability.

If the result is successful, the data have proved to have DoA, DOI, DOTu and DOF, and QualificationSC can process the data with all the guarantees. Finally, the package receiver can read the qualification data and track it back to the data generator, obtaining DOTa.

6.6. System Implementation

In this section, we implement the infrastructure in a real use case. A packet in a cold chain must maintain a temperature between the T_u and T_l within a safety margin, ω . Additionally, we identify τ_u and τ_d as the times required to climb from the "Secure Zone" to the "Dangerous Zone" and vice versa, respectively, as seen in Fig. 6.7. To ensure that the packet never enters the "Dangerous Zone" we must take samples with a period less than τ_p .

$$\tau_p = \tau_u + \tau_d \quad (6.6)$$

Table 6.1: Measurements of non-optimized SS

Operation	Time
Get RANDAO mix	140 ms
Data generation	303 ms
TX generation	638 ms

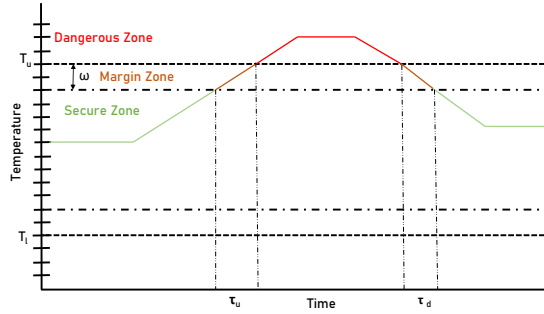


Figure 6.7: Graphical representation of the "Dangerous Zone", "Margin Zone" and "Secure Zone" of the temperature of a product in a cold chain.

From Mohamed Ahmed in [13] we take $T_l = +2^\circ C$ and $T_h = +8^\circ C$, and we consider a safety margin, $\omega = 1^\circ C$. From [57, 58] we set a continuous temperature change velocity (V_T) of a non refrigerated package of $0,1^\circ C/min$.

With this data, we calculate $\tau_p = 20min = 1200s$, therefore we set 1200 seconds as the measurement period and $3^\circ C$ and $7^\circ C$ as limits temperature in the qualification data. We consider 48 seconds as our assumable time $\Delta\tau_{AA}$ (i.e. $\gamma = 4$). Also, we set 120 seconds as the maximum time for our transaction to be accepted ($\beta = 10$) [59] and θ equal to 1 because skipped slots are very unlikely [54]. From (6.4), we get a minimum $\Delta\tau_i$ of 72 seconds with $\beta = 1$ and a maximum of 180 seconds with $\beta = 10$ which is much lower than the measurement period, 1200 seconds.

In the implementation we used a real HSM, the same that was used by Dominic *et al.* in [21], the *Blockchain Security 2Go starter kit R2* [60] connected to a low-price System on Chip (SoC), Raspberry Pi 4B: Broadcom BCM2711, Quad core Cortex-A72 (ARM v8) 64-bit SoC @ 1.5 GHz 8 GB LPDDR4-3200 SDRAM. Using a not optimized code we got the results shown in Table 6.1 where: *Get blockhash* is the time needed to ask for the blockhash of the latest block, *Data generation* represents the hash of the data and the signing operation by the SS, and the *Tx generation* is the time taken to build the raw transaction and signing it for second time. Finally, the verification of the incoming data in QualificationSC has a cost of 21830 gas, which at 22/7/2023 is equivalent to \$0.86.

6.7. Security Analysis

In this section, we provide an analysis following the indications of Security Requirements presented by Dan Liu *et al.* in [22] and explained in Section 6.2.1.

- DOAu: Every IoT node has a unique, irreplaceable and irreplicable private key which provides the IoT node with a unique address. Before accepting any data, the smart contract confirms that the sender address belongs to an accepted IoT node with a valid HSM (Owner, manufacturer, type).
- DOTa: Blockchain stores all the transaction history with their sender address. Any entity with access to the Blockchain can track the data back to the origin.
- DOI: Thanks to the use of a IoT device with hardware based security, the SS, the measurements gathered from the environment are signed in the HSM even before they can be accessed by the controller of the IoT device. Then, this data and its signature is verified by a smart contract thanks to our Blockchain-based PKI. In this way, we get end-to-end integrity protection of the data.
- DOTu: Once the identity is confirmed, QualificationCA receives the validated SeKP from SCA, and the smart contract verifies that the signature on the data was generated by SeKP before accepting the data, ensuring that the generator was a HSM in a SS. As indicated in the section 6.3.1, knowing that the HSM in a SS generated the data guarantees the trustworthiness.
- DOF: Guarantying the data freshness is essential to avoid delay attacks and replay attacks. In delay attacks, the attacker generates a bunch of correct measurements at time τ and uses them as measurements of other posterior times. Through the use of the blockhashes as nonces in the signatures, we can estimate a time slot when the data was generated of 72 seconds minimum and 180 seconds maximum with a probability of 99.98 % in Ethereum (considering an attacker investing six billion dollars). The error margin can be divided by six by extending 12 seconds the assumable time τ_{AA} .

In Table 6.2 we compare our work with those Oracles that claim can be used to send IoT information to the Blockchain. As can be observed, our solution is the only one achieving this level of information security.

Table 6.2: Comparison of Oracle protocols

Oracle Protocols	Hardware Oracle requirements					
	DOAu	DOTa	DOI	DOTu	DOF	Scalable
DiOr-SGX [19]	✗	✗	✗	✗	✗	✓
Jonathan [20]	✓	✓	✓	✗	✗	✗
Alia [38]	✗	✓	✓	✗	✗	✓
Our solution	✓	✓	✓	✓	✓	✓

6.8. Conclusion

This paper presents a set of Ethereum smart contracts that performs the authentication and attestation of IoT devices and recognizes the timestamps of data collection. Usually, any IoT device's owner controls the data collected. However, there are several use cases where Blockchain depends on sensor measurements, meaning the sensor owner could mislead the involved smart contracts. In our solution, the IoT device owner does not have any control over the IoT data. To do so, we developed an infrastructure where smart contracts receiving a measurement authenticate the sender, attest the hardware-based secure sensor and calculate the data freshness before accepting it for a low gas cost. In order to accomplish this, we measured the temperature using a hardware-protected IoT device, and designed a novel PKI to quickly authenticate IoT devices and their hardware-protected data on public Blockchains without certificates. Moreover, we developed and analyzed the tools to demonstrate the freshness of the IoT data. In this research, we proved that it is possible to send non-manipulable data from IoT devices to smart contracts, non-manipulable even by controlling the IoT device. Thus, it paves the way for the creation of several new apps based on smart contracts and allows the use of Ethereum in a variety of new scenarios involving IoT. Furthermore, its applicability to other Blockchains, like Hyperledger Fabric or Arbitrum can be studied to get a more accurate timestamp thanks to their better response time. Additionally, connecting remote attestation tools with IoT devices may make it possible to preprocess the data securely.

References

- [1] Satoshi Nakamoto. Bitcoin : A Peer-to-Peer Electronic Cash System. , pages 1–9, 2008. Available online: <https://bitcoin.org/bitcoin.pdf>, Last accessed on 2023-10-21.
- [2] Ethereum Foundation. Ethereum white paper. <https://ethereum.org/en/whitepaper/>, Last accessed on 2022-08-18.

-
- [3] Rajesh Gupta, Sudeep Tanwar, Neeraj Kumar, and Sudhanshu Tyagi. Blockchain-based security attack resilience schemes for autonomous vehicles in industry 4.0: A systematic review. *Computers & Electrical Engineering*, 86:106717, 2020.
 - [4] Mahdi H Miraz. Blockchain of things (bcot): the fusion of blockchain and iot technologies. In *Advanced applications of blockchain technology*, pages 141–159. Springer, 2020.
 - [5] Florian Wessling, Christopher Ehmke, Marc Hesenius, and Volker Gruhn. How much blockchain do you need? towards a concept for building hybrid dapp architectures. In *2018 IEEE/ACM 1st International Workshop on Emerging Trends in Software Engineering for Blockchain (WETSEB)*, pages 44–47, 2018.
 - [6] Hamda Al-Breiki, Muhammad Habib Ur Rehman, Khaled Salah, and Davor Svetinovic. Trustworthy blockchain oracles: Review, comparison, and open research challenges. *IEEE Access*, 8:85675–85685, 2020.
 - [7] Ana Reyna, Cristian Martín, Jaime Chen, Enrique Soler, and Manuel Díaz. On blockchain and its integration with iot. challenges and opportunities. *Future generation computer systems*, 88:173–190, 2018.
 - [8] Giulio Caldarelli. Understanding the blockchain oracle problem: A call for action. *Information*, 11(11):509, 2020.
 - [9] Shuai Wang, Hao Lu, Xingkai Sun, Yong Yuan, and Fei-Yue Wang. A novel blockchain oracle implementation scheme based on application specific knowledge engines. In *2019 IEEE International Conference on Service Operations and Logistics, and Informatics (SOLI)*, pages 258–262, 2019.
 - [10] Hamda Al-Breiki, Muhammad Habib Ur Rehman, Khaled Salah, and Davor Svetinovic. Trustworthy blockchain oracles: Review, comparison, and open research challenges. *IEEE Access*, 8:85675–85685, 2020.
 - [11] Abdeljalil Beniiche. A study of blockchain oracles. *arXiv preprint arXiv:2004.07140*, 2020.
 - [12] Olivér Hornyák and George Farid Alkhoury. Smart contracts in the automotive industry. In Károly Jármai and Katalin Voith, editors, *Vehicle and Automotive Engineering 3*, pages 148–157, Singapore, 2021. Springer Singapore.
 - [13] Mohamed Ahmed, Chantal Taconet, Mohamed Ould, Sophie Chabridon, and Amel Bouzeghoub. Iot data qualification for a logistic chain traceability smart contract. *Sensors*, 21(6):2239, 2021.

- [14] Kruno Miličević, Luka Omrčen, Mirko Kohler, and Ivica Lukić. Trust model concept for iot blockchain applications as part of the digital transformation of metrology. *Sensors*, 22(13):4708, 2022.
- [15] Konstantinos Christidis and Michael Devetsikiotis. Blockchains and smart contracts for the internet of things. *Ieee Access*, 4:2292–2303, 2016.
- [16] Sin Kuang Lo, Yue Liu, Su Yen Chia, Xiwei Xu, Qinghua Lu, Liming Zhu, and Huansheng Ning. Analysis of blockchain solutions for iot: A systematic literature review. *IEEE Access*, 7:58822–58835, 2019.
- [17] Peichang Shi, Huaimin Wang, Shangzhi Yang, Chang Chen, and Wentao Yang. Blockchain-based trusted data sharing among trusted stakeholders in iot. *Software: practice and experience*, 51(10):2051–2064, 2021.
- [18] Mahmoud Ammar, Bruno Crispo, and Gene Tsudik. Simple: A remote attestation approach for resource-constrained iot devices. In *2020 ACM/IEEE 11th International Conference on Cyber-Physical Systems (ICCPS)*, pages 247–258. IEEE, 2020.
- [19] Sangyeon Woo, Jeho Song, and Sungyong Park. A distributed oracle using intel sgx for blockchain-based iot applications. *Sensors*, 20(9):2725, 2020.
- [20] Jonathan Heiss, Anselm Busse, and Stefan Tai. Trustworthy pre-processing of sensor data in data on-chaining workflows for blockchain-based iot applications. In *International Conference on Service-Oriented Computing*, pages 133–149. Springer, 2021.
- [21] Dominic Pirker, Thomas Fischer, Harald Witschnig, Rainer Matichek, and Christian Steger. Trustful remote-sensing architectures based on hardware-security. In *2021 IEEE 12th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON)*, pages 0256–0263. IEEE, 2021.
- [22] Dan Liu, Zheng Yan, Wenxiu Ding, and Mohammed Atiquzzaman. A survey on secure data analytics in edge computing. *IEEE Internet of Things Journal*, 6(3):4946–4967, 2019.
- [23] Jiafu Wan, Jiapeng Li, Muhammad Imran, Di Li, et al. A blockchain-based solution for enhancing security and privacy in smart factory. *IEEE Transactions on Industrial Informatics*, 15(6):3652–3660, 2019.
- [24] Zackary Hess, Yanislav Malahov, and Jack Pettersson. *Æternity blockchain*. , 2017. <https://whitepaper.io/document/14/aeternity-whitepaper/>, Last accessed on 2022-09-21.

-
- [25] Deepak Puthal and Saraju P. Mohanty. Proof of authentication: Iot-friendly blockchains. *IEEE Potentials*, 38(1):26–29, 2019.
- [26] Stephanos Matsumoto and Raphael M Reischuk. Ikp: turning a pki around with decentralized automated incentives. In *2017 IEEE Symposium on Security and Privacy (SP)*, pages 410–426. IEEE, 2017.
- [27] 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.
- [28] Alexander Yakubov, Wazen Shbair, Anders Wallbom, David Sanda, et al. A blockchain-based pki management framework. In *The First IEEE/IFIP International Workshop on Managing and Managed by Blockchain (Man2Block) colocated with IEEE/IFIP NOMS 2018, Taipei, Taiwan 23-27 April 2018*, 2018.
- [29] Mohamed Laarabi, Badreeddine Chegri, Abdelilah Maach Mohammadia, and Khaoula Lafrioui. Smart contracts applications in real estate: A systematic mapping study. In *2022 2nd International Conference on Innovative Research in Applied Science, Engineering and Technology (IRASET)*, pages 1–8. IEEE, 2022.
- [30] Seyoung Huh, Sangrae Cho, and Soohyung Kim. Managing iot devices using blockchain platform. In *2017 19th international conference on advanced communication technology (ICACT)*, pages 464–467. IEEE, 2017.
- [31] Carlos Molina-Jimenez, Ellis Solaiman, Ioannis Sfyarakis, Irene Ng, and Jon Crowcroft. On and off-blockchain enforcement of smart contracts. In *European Conference on Parallel Processing*, pages 342–354. Springer, 2018.
- [32] Shayan Eskandari, Jeremy Clark, Vignesh Sundaresan, and Moe Adham. On the feasibility of decentralized derivatives markets. In *International Conference on Financial Cryptography and Data Security*, pages 553–567. Springer, 2017.
- [33] Jack Peterson, Joseph Krug, Micah Zoltu, Austin K Williams, and Stephanie Alexander. Augur: a decentralized oracle and prediction market platform. *arXiv preprint arXiv:1501.01042*, 2015.
- [34] Fan Zhang, Ethan Cecchetti, Kyle Croman, Ari Juels, and Elaine Shi. Town crier: An authenticated data feed for smart contracts. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, pages 270–282, 2016.

- [35] Steve Ellis, Ari Juels, and Sergey Nazarov. Chainlink. , 2017. <https://whitepaper.io/document/14/aeternity-whitepaper/>, Last accessed on 2022-09-21.
- [36] John Adler, Ryan Berryhill, Andreas Veneris, Zissis Poulos, Neil Veira, and Anastasia Kastania. Astraea: A decentralized blockchain oracle. In *2018 IEEE international conference on internet of things (IThings) and IEEE green computing and communications (GreenCom) and IEEE cyber, physical and social computing (CPSCom) and IEEE smart data (SmartData)*, pages 1145–1152. IEEE, 2018.
- [37] Adán Sánchez de Pedro Crespo, Daniele Levi, and Luis Iván Cuen-de García. Witnet: A decentralized oracle network protocol. *CoRR*, abs/1711.09756, 2017.
- [38] Alia Al Sadawi, Mohamed S. Hassan, and Malick Ndiaye. On the integration of blockchain with iot and the role of oracle in the combined system: The full picture. *IEEE Access*, 10:92532–92558, 2022.
- [39] Marco Autili, Francesco Gallo, Paola Inverardi, Claudio Pompilio, and Massimo Tivoli. Introducing trust in service-oriented distributed systems through blockchain. In *2019 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW)*, pages 149–154, 2019.
- [40] Guido Bertoni, Joan Daemen, Michaël Peeters, and Gilles Van Assche. Keccak. In Thomas Johansson and Phong Q. Nguyen, editors, *Advances in Cryptology – EUROCRYPT 2013*, pages 313–314, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.
- [41] Hilarie Orman. Blockchain: The emperors new pki? *IEEE Internet Computing*, 22(2):23–28, 2018.
- [42] Nick Szabo. Formalizing and securing relationships on public networks. *First Monday*, 2(9), Sep. 1997.
- [43] Vitalik Buterin et al. A next-generation smart contract and decentralized application platform. *white paper*, 3(37), 2014.
- [44] Markus Jakobsson and Ari Juels. Proofs of work and bread pudding protocols. In *Secure information networks*, pages 258–272. Springer, 1999.
- [45] Gabriel Estevam, Lucas M Palma, Luan R Silva, Jean E Martina, and Martin Vigil. Accurate and decentralized timestamping using smart contracts on the ethereum blockchain. *Information Processing & Management*, 58(3):102471, 2021.

-
- [46] Yuan Zhang, Chunxiang Xu, Nan Cheng, Hongwei Li, Haomiao Yang, and Xuemin Shen. Chronos +: An accurate blockchain-based time-stamping scheme for cloud storage. *IEEE Transactions on Services Computing*, 13(2):216–229, 2019.
- [47] Ethereum Foundation. The merge. <https://ethereum.org/en/upgrades/merge/>, Last accessed on 2022-08-18.
- [48] Ethereum Foundation. The beacon. <https://ethereum.org/en/upgrades/beacon-chain/>, Last accessed on 2022-08-18.
- [49] Ethereum Foundation. The randao. <https://github.com/ethereum/consensus-specs/blob/dev/specs/phase0/beacon-chain.md#randao>, Last accessed on 2022-08-18.
- [50] Ethereum Foundation. Transactions. <https://ethereum.org/en/developers/docs/transactions/>, Last accessed on 2022-08-18.
- [51] 24 gwei | ethereum gas tracker | etherscan.
- [52] Trusted Computing Group. Tpm 2.0 library, Mar 2021.
- [53] Danny Ryan Mikhail Kalinin. Eip-4399. <https://github.com/ethereum/EIPs/blob/master/EIPS/eip-4399.md>, Last accessed on 2022-08-18.
- [54] Etherscan Team. Beaconsan. <https://beaconsan.com/statistics>, Last accessed on 2022-10-20.
- [55] Ethereum Foundation. Proof-of-stake (pos). <https://ethereum.org/en/developers/docs/consensus-mechanisms/pos/>, Last accessed on 2022-08-18.
- [56] Corwin Smith, Awosika Emmanuel, Joshua, Shelley Olivia, and Sam Richards. Optimistic rollups. <https://ethereum.org/en/developers/docs/scaling/optimistic-rollups/>, Last accessed on 2022-09-05.
- [57] Jian Sun, Mingkan Zhang, Anthony Gehl, Brian Fricke, Kashif Nawaz, Kyle Gluesenkamp, Bo Shen, Jeff Munk, Joe Hagerman, Melissa Lapsa, et al. Dataset of ultralow temperature refrigeration for covid 19 vaccine distribution solution. *Scientific Data*, 9(1):1–8, 2022.
- [58] Joseph Habiyaremye. Fridge data for 18 days. <https://www.kaggle.com/datasets/josephsoso/fridge-data-for-18-days>, Last accessed on 2022-08-22.
- [59] Lin Zhang, Brian Lee, Yuhang Ye, and Yuansong Qiao. Evaluation of ethereum end-to-end transaction latency. In *2021 11th IFIP International Conference on New Technologies, Mobility and Security (NTMS)*, pages 1–5. IEEE, 2021.

- [60] Markus Moesenbacher. Silicon trust webinar may 2021 secora™ blockchain. <https://silicontrust.org/wp-content/uploads/2021/05/SECORA-Blockchain-.pdf>, Last accessed on 2022-08-23.

Chapter 7

RESEKRA: Remote Enrollment Using SEaled Keys for Remote Attestation

Ernesto Gómez-Marín ^{1,2}, Luis Parrilla ², Gianfranco Mauro ^{1,2}, Antonio Escobar-Molero ¹, Diego P. Morales, ² and Encarnación Castillo ²

1. Infineon Technologies AG, Am Campeon 1-15, 85579 Neubiberg, Germany

2. Departamento de Electrónica y Tecnología de Computadores, Universidad de Granada, 18071 Granada, Spain

MDPI Sensors

- Received: 31 May 2022, Revised: 24 June 2022, Accepted: 2 July 2022, Published: 5 July 2022
- 10.3390/s22135060
- Impact factor: 3.847
- JCR Rank: 19/64 in category Instruments and instrumentation (Q2)

Abstract: This paper presents and implements a novel remote attestation method to ensure the integrity of a device applicable to decentralized infrastructures, such as those found in common edge computing scenarios. Edge computing can be considered as a framework where multiple unsupervised devices communicate with each other with lack of hierarchy, requesting and offering services without a central server to orchestrate them. Because of these characteristics, there are many security threats, and detecting attacks is essential. Many remote attestation systems have been developed to alleviate this problem, but none of them can satisfy the requirements of edge computing: accepting dynamic enrollment and removal of devices to the system, respecting the interrupted activity of devices, and last but not least, providing a decentralized architecture for not trusting in just one Verifier. This security flaw has a negative impact on the development and implementation of edge computing-based technologies because of the impossibility of secure implementation. In this work, we propose a remote attestation system that, through using a Trusted Platform Module (TPM), enables the dynamic enrollment and an efficient and decentralized attestation. We demonstrate and evaluate our work in two use cases, attaining acceptance of intermittent activity by IoT devices, deletion of the dependency of centralized verifiers, and the probation of continuous integrity between unknown devices just by one signature verification. Keywords: Remote attestation; Edge Computing; Internet of Things; embedded systems; Trusted Platform Module

7.1. Introduction

The number of things connected to the internet (Internet of Things, IoT) is growing exponentially, from 6.1 billion in 2018 to 14.7 billion in 2023 [1]. This growth is continuous, and there is no evidence that it will stop.

The current IoT structure relies on cloud computing. An IoT device collects information from the environment (sensor), sends it to a distant centralized server, which processes it along with currently stored information and information from other sensors, and generates, whether required, an action response. The response is then sent to a new IoT device, namely an actuator, which performs the appropriate action often in the nearby of the sensor. This is a simple and secure system from the service provider's point of view, but due to the remoteness of the server from the nodes, it generates a high volume of data traffic and a high latency not suitable for real-time IoT applications. Moreover, it is not scalable enough with the current growth of IoT devices because of the large network use and the burden on cloud servers [2, 3].

In this context, it is convenient for data to be processed at the edge for shorter response times, more efficient processing, and less pressure on the network. The technologies that enable computation at the edge of the

network, on the data flow between the cloud services and the IoT services are known as edge computing [2]. In this scenario, the devices at the edge of the network that provide services (processing and/or data storage) are called edge nodes, while user devices are those that require real-time interaction or high storage capabilities. In this paradigm, user devices will act as edge nodes whenever possible, offloading their computing task to adjacent nodes if too burdensome at a given time [3].

On the other hand, the inherent distributed structure of edge computation implies additional security challenges over cloud computing. Edge computing has to protect different layer of technologies (from cloud to IoT devices) as in cloud computing, but it also needs to provide a distributed global connectivity of heterogeneous devices between the different layers. Rodrigo Roman et al. [4] define it as a combination of “the worst-of-all-worlds”. The scenario becomes more complicated when we consider that user devices continuously enter and leave the local network, that means, close mobile users (mobile subscribers), e.g., mobile phones or cars, which is called Mobile Edge Computing (MEC) [5].

In addition, it should be considered that the possible impact of an attack can be very high. Given the wide variety of situations in which edge computing can be used, the consequences can range from affecting our private information, the daily lives of users, and, more indirectly, the industrial ecosystem or critical infrastructure. It is therefore easy to override the possible benefits of edge computing by the losses that can result from relying too much on such technology [4]. Most edge devices do not have a user interface, which causes attacks to go unnoticed by most users [6]. Moreover, one of the main currently open problems in MEC is the security, in particular, the robustness of MEC servers [7]. Hence, verifying the status of IoT devices, edge nodes or not, becomes a complex and critical task in edge computing. This leads to the need of implementing a remote attestation system for edge devices and users [8].

In remote attestation, a Verifier checks the correct status of a device (Attestor) remotely. This solution has been widely investigated in the literature, but when applied to IoT systems, the current RA protocols are hard to scale. To overcome this challenge, several works recently proposed Collective RA (CRA) protocols. However, these solutions brought up new open issues [8]: the need of scalables and decentralized key management allowing mobility, the acceptance of intermittent activity of IoT nodes, which is essential for edge computing, and the resistance against the attack Time Of Check To Time Of Use (TOCTTOU) [9], which is barely covered in the State of the Art (SoA). In this paper, we present a solution for these problems through our remote attestation method, RESEKRA.

In our proposal, we attest remotely the state of an untrusted device through the support of a standard Root of Trust (RoT) consisting of a

Trusted Platform Module [10]. This is a work already achieved and applied to edge computing in [11, 12, 13], but we go further.

Additionally to the results obtained in [11, 12, 13], we provide the following features:

- Proving authenticity in any communication also proves its correctness.
- No pre-shared secret is needed between the Verifier and the Attestor, which makes the system able to easily include new devices which is essential for systems with a variable number of nodes such as edge computing.
- Allowing secure software updates on the Attestor.
- Enabling offline remote attestation. The Attestor can be verified by the end user themselves.

For achieving these features, we assume that:

- The Attestor shall include the needed hardware root of trust (RoT).
- A trusted Attestor manufacturer is available.
- The attacker cannot modify the RoT.

Our system, called RESEKRA, has been implemented on a Raspberry Pi by connecting a TPM and a pressure sensor to achieve a secure sensor acting as an Attestor that connects remotely to an external computer acting as a Verifier. The system attains an online and dynamic enrollment phase that works even on untrusted state devices, which allows it to be used in decentralized key management, where multiples entities are responsible for the key generation, distribution and regeneration with the advantage of fault tolerance, scalability [14] and flexibility. This makes RESEKRA the first remote attestation system for large networks of IoT devices with decentralized key management [8]. Moreover, Attestor can prove correct software status directly to the user devices, which make it perfect for users that just entered in the edge network, that means, mobile subscribers in MEC. Additionally, our scheme allows asynchronous booting, where the programs of the IoT devices are initialized in a random order, which is required for Linux-based systems, and finally, RESEKRA does not interfere with updates; therefore, a trusted online software updated system could be easily included in our system. To the best of our knowledge, there is no other system able to provide these services.

The rest of the manuscript is structured as follows: Section 7.2 is devoted to the related work, while Section 7.3 presents the technologies needed to understand the system. Section 7.4 details the RESEKRA design, whose

implementation is presented in Section 7.5 as a solution for real scenarios. In Section 7.6, we will analyze the effects of the common attacks to the remote attestation systems in RESEKRA, and in Section 7.7, the conclusions of the work are carried out.

7.2. Related Work

Remote attestation is a process that has been studied extensively for many years and has recently focused on IoT nodes, creating their own field, Collaborative Remote Attestation (CRA) [8]. Due to the heterogeneity of IoT devices, there are several research studies of software-based solutions to increase scalability such as [15, 16, 17], but all require setting up one-hop networks between the Verifier and Attestor. This limitations make their implementation challenging. In addition, software-based solutions make strong assumptions about the limits of attacker capabilities, thus reducing their reliability [8]. To solve the latter problems, the use of a Root of Trust (RoT) residing in the device has been proposed in the literature [18]. This RoT is usually a mix of hardware and software. In [8], the authors divide RoT used in CRA into two main categories: those using hardware with the minimal security capabilities and those using a TPM.

In the branch of RoT based on hardware with the minimal security capabilities—also known as hybrid attestation techniques [19, 20]—the authors specify their non-standard security hardware designs. SMART [21], ERASMUS [22], TrustLite [23] and SEED [20] provide high-level details of their hardware solutions. SEED and ERASMUS are the only solutions also requiring a Real-Time Clock, which is used to make the attestation to start from the Attestor itself. LISA [24] requires the same hardware architecture as SMART, but additional hardware is added to authenticate the Verifier and avoid DoS attacks. All the solutions mentioned above require the hardware design and manufacturing of novel security hardware, making them hard to implement. Another solution of the SoA is Hatt [19] where very limited additional hardware is required: “Physical Unclonable Functions” (PUF) and a ROM for the attention code. Still, they do not explain how to protect the PUF from being accessed by a malicious code. Finally, the research SARA [25] defines the requirements of their root of trust without providing any particular details. From all the presented works, only ERASMUS considers TOCTTOU attacks in their respective adversarial model.

On the other hand, other solutions choose security hardware with the highest security standards such as Trusted Platform Module (TPM). In “Remote Enrollment using SEaled Keys for Remote Attestation” (RESEKRA), we opted for the use of TPM. As Tan et al. argue in “TPM-enabled Remote Attestation Protocol” (TRAP) [13], the cost of current TPMs is relatively low compared to the price of sensors, both in price and area. And as final

product that can be found in the market, they can be easily implemented in real applications. As in the work proposed by Miguel Calvo and Marta Beltran [11] and MTRA (TRAP-based system) [12], we propose the use of a TPM where the Platform Configuration Registers (PCR) values, further discussed in 7.3.1.8, are used as proof of the current state of the IoT device. MTRA also considers TOCTTOU in their adversary analysis.

RESEKRA differs from all of the above works, whether based on hardware with the minimal security capabilities or TPM-based, in that none of them have any of the following features: enabling dynamic and online enrollment, which is essential for edge computing and even more for Mobile Edge Computing; decentralized key management and decentralized Verifier structure, removing single points of failure and trusted third parties; and correct analysis of devices with intermittent activity, which is critical in IoT computing because IoT devices are not always online. Additionally, we also consider TOCTTOU attacks, which are only covered in Erasmus and MTRA. All of them are open issues not addressed in the current Collaborative Remote Attestation solutions of the SoA [8].

7.3. Background

This section will outline the tools and technologies necessary for understanding RESEKRA: TPM2.0, Integrity Measurement Architecture (IMA) [26] and Core Root of Trust of Measurement (CRTM) [27].

These three elements are part of the Attestor. The CRTM ensures the correctness of firmware configuration and IMA. The IMA guarantees the correctness of the software configuration at runtime. Finally, the TPM2.0 ensures the reliability of all the information when shared externally (results generated by IMA and CRTM, among others). In Nomenclature part, we will summarize all the notations used in the document.

7.3.1. TPM 2.0

The Trust Platform Module 2.0 [10] provides standardized specifications for security coprocessors. Hereafter, security coprocessors that follow these specifications will be called TPM, and the device that has the TPM will be called the Host-TPM. These specifications have many details and functionalities, and only the features needed for our system will be described below.

7.3.1.1. Virtual Memory

The TPM has a limited protected “real” memory, but credentials and sensitive data can be stored in the non-protected memory of the Host-TPM (computer that is using the TPM). This material is protected through

signatures and encryption to ensure integrity and confidentiality. With this system, the TPM can use the Host-TPM device's memory as a virtual protected memory of the TPM.

7.3.1.2. Key Attribute: “Restricted”

The keys generated by the TPM have attributes that cannot be modified and which limit the use of these keys. Many of them must be verified to guarantee the security of the system. One of these essential attributes is Restricted. Keys with this attribute sign only TPM-generated digests in the signing process and will never sign a document starting with the value `"0xf544347"`, which is also called `TPM_GENERATED`.

7.3.1.3. TPM_GENERATED

Data that begin with the code `TPM_GENERATED` can be signed by a Restricted key if and only if it was generated by the TPM. If the authenticity of the Restricted key is guaranteed, so is the veracity of the information related to the TPM. When used correctly, this allows much information to be remotely and reliably derived from the TPM in addition to the PCR values, such as the characteristics of the private keys stored in the TPM. This is a functionality that is rarely used in the state of the art and is the first reason why we stand out. With it, we can remotely complete the entire enrollment process of the IoT device to be attested.

7.3.1.4. Policies

TPM2.0 offers a large set of policies that are not commonly used in typical applications. To the best of our knowledge, only [13] uses these available policies when sealing secret keys with PCR values. The use of these policies is one of the contributions carried out in RESEKRA. Only if the policies under which the TPM objects were created are satisfied, the object can be unsealed and then used. The three policies used in this work are the following:

- `tpm2_policypcr`: Seals the object to the value of one or more PCRs of the TPM.
- `tpm2_policycountertimer`: Seals the object to the number of times the TPM has been restarted.
- `tpm2_policyauthorize`: Seals the object to the policies signed by a Trusted Third Party (TTP). This policy allows changing the policies of a remotely signed object. With this policy, in RESEKRA, the Verifier is able to remotely update the other two policies, thus enabling flexible updating of the correct PCR value, and consequently, asynchronous booting and remote updates performing.

The name of an object (*Object_name*) is computed from the public information of the object (*Object_Pubdata*) as the public key (*Object_PuB*), the attributes, and policies. As a consequence, the *Object_name* can be used to assert the integrity of the object’s policies.

7.3.1.5. Attestation Key

Attestation key (AK) is used to verify the internal information of the TPM. As mentioned above, one of its most important attributes is “Restricted”. The key name (*AK_name*) is computed from the public information of the attestation key (*AK_Pubdata*), as the public key (*AK_PuB*), attributes, and policies.

7.3.1.6. Endorsement Key

The Endorsement Key (EK) is an assymmetric key stored in the real memory of the TPM, which comes with a certificate signed from the TPM Manufacturer. The EK together with the Endorsement Key certificate (EK certificate) are used for two purposes: to verify the existence of the TPM and its manufacturer and to verify that an object is stored in the TPM. If an object, such as a private key, is shown to be stored in the TPM, its characteristics and attributes can be trusted as long as the TPM is trusted, such as the “Restricted” attribute.

It is a very restricted key and can only be used to decrypt data with a specific structure. Therefore, the TPM can prove ownership of the EK just through decryption operations [28]. For this task, we use the operations “makecredential/activatecredential”.

7.3.1.7. Makecredential/Activatecredential

An external entity performs the *makecredential* operation using as inputs the Public Endorsement Key (*EK_PuB*), the name of a TPM object, usually the name of an attestation key (*AK_name*), and a secret value:

$$Credential = makecredential(EK_PuB, AK_name, secret) \quad (7.1)$$

This command creates the so-called *Credential*. The TPM receives it and computes *activecredential* and, if and only if the *AK_name* object belongs to the TPM, it will decrypt the *Credential* by retrieving the secret. By showing knowledge of the secret, the Host-TPM proves that it has a TPM with EK and that the object *AK_name* is real and accurate.

7.3.1.8. Platform Configuration Registers

The Platform Configuration Registers (*PCR*) are the basis for TPM-based remote attestation, which are also called Trusted Attestation Protocol (TAP) [29]. These registers can be updated through the “Extension” operation only [30]:

$$PCR\ new\ value = hash(PCR\ old\ value || data\ to\ extend) \quad (7.2)$$

“Because of the one-way nature of a secure digest, there is no way to undo a measurement” [30]. Therefore, once a measurement has been stored in the PCR, no one, even with the highest privileges can override the effect of this measurement in the PCR, making PCRs perfect for verifying the integrity of the measurement list generated by the Integrity Measurement Architecture (IMA).

7.3.2. Integrity Measurement Architecture (IMA)

The Integrity Measurement Architecture (IMA) is a tool available in Linux responsible for measuring the files before they are accessed, storing the measurements in a list and extending them to the PCR. The IMA keeps a runtime measurement list; therefore, if any new or edited file is accessed, the path, name and content are measured, as shown in Equations (7.3) and (7.4), and they are included in the measurement list with the corresponding extension to the PCR.

$$filedata\ hash = hash(filedata) \quad (7.3)$$

$$measurement = hash(filedata\ hash, file\ path|name) \quad (7.4)$$

Any program or file that would modify the system architecture such as the IMA itself would be measured before being executed, and it would leave an indelible mark on the PCR. Providing the measurement list together with the PCR will ensure the integrity of the measurement list. Therefore, for trusting in this attestation, it is needed to trust in the first program that started the measuring process and thus was never measured: the Core Root of Trust of Measurement.

7.3.3. Core Root of Trust of Measurement

As pointed out in [27], the Core Root of Trust of Measurement (CRTM) is the “first piece of BIOS code that executes on the main processor during the boot process. On a system with a Trusted Platform Module the CRTM (Core Root of Trust of Measurement) is implicitly trusted to bootstrap the

process of building a measurement chain for subsequent attestation of other firmware and software that is executed on the computer system". The CRTM is essential to being able to trust in IMA. The only way to trust in a CTRM is by trusting in the device's manufacturer.

7.4. RESEKRA Description

As will be detailed in this section, the main contributions of RESEKRA can be summarized as follows:

- The use of Sealed Keys that can be used by the proprietary (the Attestor) temporarily and only when the current device status is approved by the Verifier, thus proving correct Attestor status only by proving ownership of the Sealed Key.
- The same Sealed Key can be used with different device status (always approved before-hand by the Verifier), allowing software updates without continuous revocations and avoiding a complex PKI.
- The creation of the Sealed Key can be realized completely remotely in a untrusted Attestor, allowing great flexibility and a plug-and-play business model perfect for Edge computing.

Figure 7.1 shows the general scheme of the remote enrollment process in RESEKRA. In the next subsections, the roles and operations performed in each of the steps will be detailed.

7.4.1. Roles

The main elements included in RESEKRA are the following:

- Attestor: edge devices with the hardware RoT, i.e., TPM and CRTM.
- Hardware Provider: the entity in charge of manufacturing the Attestor and providing the firmware and RoT. It is considered a trusted entity.
- Software Provider: the entity in charge of designing the Attestor software and creating the RML. It is a trusted entity.
- Programmer: the entity in charge of programming the Attestor. It is a not trusted entity.
- Deployer: the entity responsible for physically deploying the device and providing an authentication method. This entity is semitrusted, because it does not need a high level of knowledge to securely deploy the device, since there is no offline enrollment process and it has no

interest in hacking the system because it is one of the stakeholders interested in providing the service. On the other hand, it would require a high level of expertise to manipulate physically the root of trust and go unnoticed by the rest of stakeholders.

- Verifier: the entity in charge of verifying the root of trusts of the Attestor, creating the SeK, verifying the Attestor status and authorizing the use of the SeK in the TPM.
- TPM: Hardware security module following the TPM 2.0 specifications.
- Edge computing user: the entity that requires communication with the Attestor to receive data (sensor) or to analyze/store data (Edge node).

In this paper, we do not focus on authentication. We consider that the Deployer is responsible for providing an authentication method for the Attestor, edge computing user and Verifier.

7.4.2. Certification of Manufacturer

In this first phase, the Attestor manufacturer and programmer shall create the needed certificates.

7.4.2.1. Hardware Provider

When the Attestor is fabricated, the Device Manufacturer stores the necessary certificates (Manufacturer Certificates) in the device to identify the existence of the necessary root of trust (TPM and CRTM) on the Attestor. This certificate is signed by the Hardware Provider and linked to the EK. The Verifier can trust several Device Manufacturers.

7.4.2.2. Software Provider

The software provider installs all the needed software in the Attestor and generates a list of measurements that will be used as reference, the Reference Measurements List (RML). This list is then signed by the software provider. The RML has to be provided to the Verifier and updated in case of software update. We will provide two possible solutions in Section 7.5. The software provider can be any stakeholder, e.g., Hardware provider, deployer or end user.

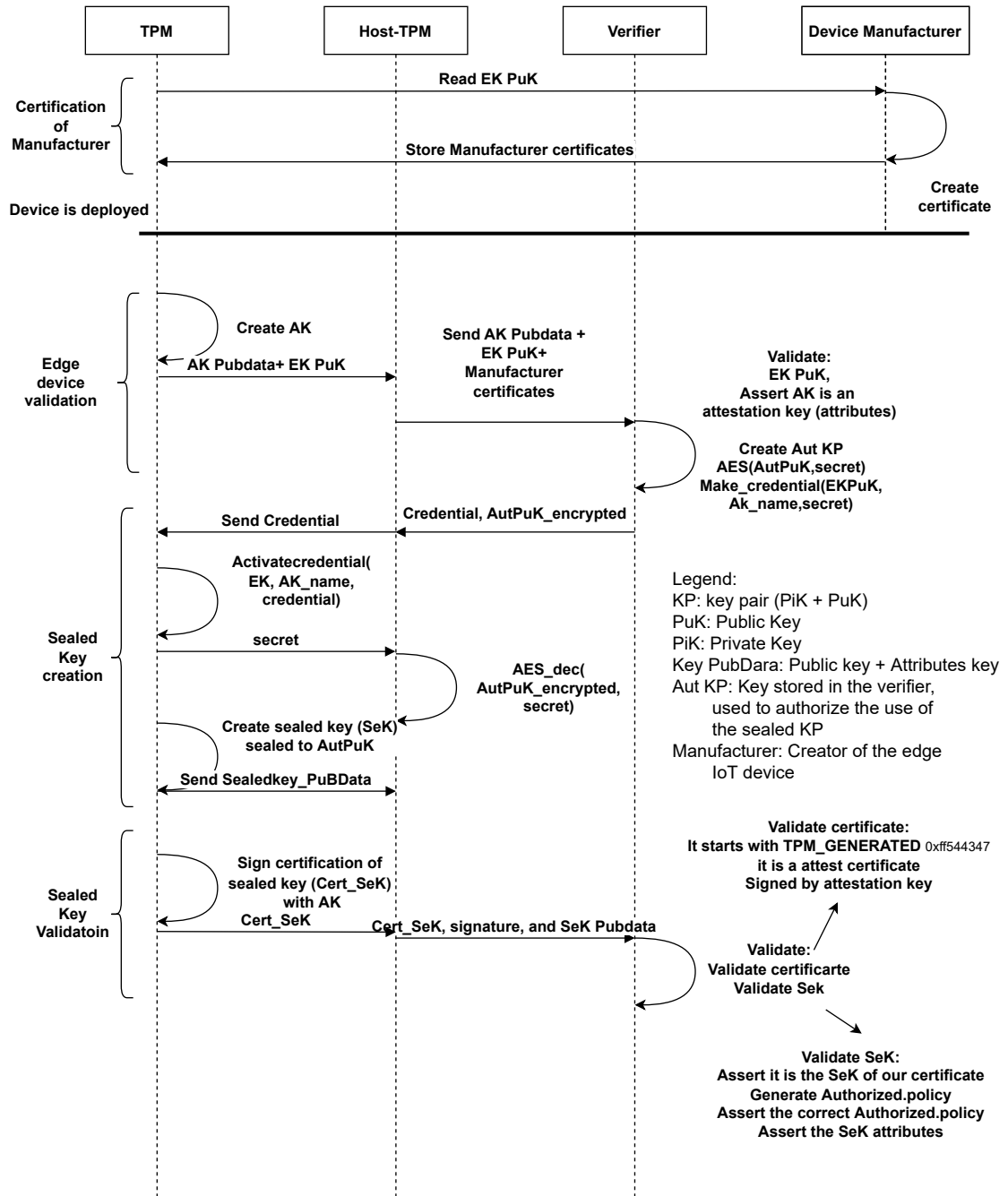


Figure 7.1: Remote enrollment process. The scheme represents all the enrollment process from the device manufacturer to the supervised creation of the sealed key.

7.4.3. Remote Enrollment

At this moment in the process, the device can be deployed by the owner in any considered location. Because the rest of RESEKRA will work remotely, the deployment can be realized by inexperienced staff. Once the device is deployed, the Verifier has to assert the device's hardware Root of Trust and to manage the creation and the validation of some special keys in the Attestor. It is a completely remote process and can be realized even in a untrusted Attestor. Therefore, the Verifier's role can be dynamically changed, and even multiple Verifiers can work at the same time, sharing trust and responsibility.

7.4.3.1. Edge Device Validation

Once the IoT device has been deployed, the Host_TPM requests the TPM for the public Endorsement Key (EK_PuK) and orders the creation of the Attestation Key (AK). It sends the public information of Attestation Key (AK_Pubdata) and the public Endorsement Key (EK_PuK) to the Verifier. The Verifier checks that the Endorsement Key belongs to a genuine TPM from a trusted manufactured edge device. The check is done through the Attestor's Manufacturer Certificates. Then, it verifies the attributes of AK (the Restricted attribute, among others).

7.4.3.2. Sealed Key Creation

The Verifier knows that the EK_PuK belongs to a trusted manufactured edge device but has not yet asserted that the Attestor is the owner of this EK. Then, the Verifier creates the asymmetric key pair, Authorizer, and encrypts the public key (Aut_PuK) with symmetric encryption using "Secret" as the symmetric key.

Next, it encrypts "Secret" using Makecredential and using EK_PuK and AK_Pubdata as input to create Credential. If the Host-TPM shows knowledge of Aut_PuK, it proves to own the TPM, EK, and AK with the asserted attributes. Finally, the Verifier sends Credential and Aut_PuK encrypted to Host-TPM. If Host-TPM shows knowledge of Aut_PuK, it proves the ownership of EK.

When Credential reaches Host-TPM, it decrypts the Credential with Activatecredential obtaining "Secret" and uses it to decrypt Aut_PuK. Now, because it has Aut_PuK, it can instruct TPM to create a sealed key (SeK) with the "tpm2_policyauthorize" using Aut_PuK as the authorizing key. The output of running the sealed key creation script is shown in Figure 7.2.

```

pi@raspberrypi: ~$ sudo sh 4_KCV.sh
Credential received

AuthPuK encrypted:
PKm6Zhk5SXUqWcrDkaAQZOMSKfsN60VX2j+jMxoz0bEnT9oR+cGriFmv18dpuc6Vl065U+GjkGXTAALAVi5FQFgKCBdG1kM0bbaI+wortUiA14vSvFmYmVDyZqUqRiS2gwjP3m
jHtYUig/axonYcMa9oUqRUOSUNkz06Qdo81qzpyDlPyZsCPS+15WQuONSyyAqderzC+YT9sCUZusqHhP/Uhk1E17itZtszF61F+Wk4465U+60EwhhKq/BYYG9T5wnBm95187Nn
+ntP/I6LePCwFplwaoDyRwhIT236PTxwshy8dLsYr7w7YAAqL167KIFr2bIk4zgLIR/YGV0sMMAPJmJd3AddExzt/gG9TCMcFgnwxyRJRzhdAcMkop5tohzFvkz7Wsz50aR1rN
w8iVzbih7GnbHMKiF/fNqKfxHrb2AFOX1qkOgZ1SCAR7SeEbrNOgha9jEtabzBo7Nvz1/2F9zRH5rq1FjQ2bZwUL0Jr7pJmSH0MjFDRXr7Fae5CR6nQrv6pDF2PSER2ZKarCh
2ny8GP+H1h0SeuFo/8jHBDabrT1xu0XqLKMwX3G4Wj4uh2d5r1mk5WYYM1Q==
Decrypting AESkey

837197674484b3f81a90cc8d46a5d724fd52d76e06520b64f2a1da1b331469aa
certinfo: bac659de12b8b2869f9b86c7e3301e6892955286cf90ae79dfc79abe9542a928

-----

Decrypting Auth PuK (AES decryption)
Key size: 32 Bytes
ciphertext in base64 size: 601 Bytes
bac659de12b8b2869f9b86c7e3301e6892955286cf90ae79dfc79abe9542a928
Decrypted text is:
----BEGIN PUBLIC KEY-----
MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAmZ6BrR3jeccdLx6NnuUr44Xv3U9mSeFwCRiQEJ5C2mg1r5BYtg18sy+wu1VDWqdInYDk1WZa5/6jnaG0A9JEAiPRbkw
KgC3aDD3g6e088b8kp+HzupRPKbgS6xP1OVjmwz2VCmv99C25HdG/wHjgK100LBDHDQaspdS1q5oABv80J5+2PTa9qI7giM8vnn7u94XaFxfiCuDNoa33n2TSU+R4Pqj+Qp9Ign
7DeZxJ+z2PpSX1Cx0Sxx790ymRyavfN/PeM4XEVHEExWgl6Vs8ITV5pbcfJWiSUH3bTVcKtWUm/62frmJyYAYHme9gMfCVmcsSjcfLBHQ550FTwDejmQIDAQA8
-----END PUBLIC KEY-----

written to ./AESkey.credential

-----

Loading Auth public key into the TPM
name: 000b9146cfb4a9c60d8eacde73ecf4b729aa17aaa3d58611a5db88903daa75704d44

Creating authorization policy with the public key
19c004bbff8d386b26e1ade115fa1acdaf8ca2ee9e39067662a24978ba1f43c2

Generating primary key in owner hierarchy
Creating and loading ECC-256 key pair under the authorization policy (Sealed Key)
attributes:
  value: fixedtpm|fixedparent|sensitive|dataorigin|decrypt|sign
authorization policy: 19c004bbff8d386b26e1ade115fa1acdaf8ca2ee9e39067662a24978ba1f43c2

```

Figure 7.2: Output of the script for sealed key creation.

7.4.3.3. Sealed Key Validation

The TPM generates an attestation certificate for SeK (Cert_SeK). This certificate is asserting that the object with the name SeK_name is stored in the TPM; hence, the TPM is bearing it. Then, the TPM signs it with the AK and sends it to the Verifier together with SeK_Pubdata.

The Verifier has to validate this information once received. First, the Verifier checks that the certificate starts with TPM_GENERATED and is signed with the suspected AK. This means that the TPM has generated the certificate if AK is a real AK, which will be verified in the next step. Finally, the Verifier needs to assert that this is the correct type of certificate.

- It starts with TPM_GENERATED;
- It is signed by AK;
- It is an attest certificate.

This validation means that Cert_SeK is genuine. Now, the Verifier will recompute the SeK_name locally from SeK_Pubdata. It shall match with the name backed by the Cert_SeKcheck, and finally, the Verifier will verify all the features of SeK:

- The sealed key is the same being attested by the certificate;
- The policy used to create the sealed key is correct (tpm2_policyauthorize using Aut_Pub). This point is essential, because it proves knowledge of the Aut_Pub;
- Verifying the attributes of the SeK.

If all verifications are successful, the Verifier has been able to create completely remotely, without the need for pre-shared secret, a sealed key in the Host-TPM's TPM. The Verifier would have complete control over this sealed key. The Attestor will be able to use this key only when satisfying the signed requirements imposed by the Verifier.

7.4.4. Attestation

At this point, the CRTM measures the BIOS and the IMA measures the integrity of all files executed. This is a runtime measurement, so when any new or modified file is executed, it is measured, and the result is stored in the measurement list and extended to the PCR.

The Attestor initiates the remote attestation by sending a request to the Verifier, and the last one replies by sending back a random nonce to the Attestor. The TPM generates a Quote signed by the AK, which includes the nonce to avoid replay attacks. The Quote is a certificate including several important parameters; those most relevant for RESEKRA are:

- TPM_GENERATED, to confirm the veracity when signed by the AK;
- The random nonce, to avoid replay attacks;
- Reset value of TPM. Value that changes when Host-TPM is rebooted;
- Value of PCR.

The Attestor sends the Quote together with the measurement list to the Verifier. It asserts the veracity of the certificate (TPM_GENERATED, signature, and random nonce) and uses the measurement list to rebuild the PCR value found in Quote. If the reconstruction is correct, it means that the measurement list was not adulterated.

Finally, the Verifier compares the measurement list with the Reference Measurement List (RML). The Verifier may have obtained this RML through the Host-TPM signed by the trusted Device Manufacturer or from the cloud. Now, the Verifier can check program by program if there is a difference and where.

When the list is approved by the Verifier, the Verifier can create an authorization to use the SeK for the particular values of RESET and PCR

```
TPM_policies.PolicyPCR_creation(Hex.decode(computedPcrSha256));  
TPM_policies.PolicyReset_creation(resetCount);  
String authorization_signature = RSAk.sign_byte(TPM_policies.Last_policy);
```

Figure 7.3: Section of the Verifier code where it computes the policies for using the SeK and signs it if and only if the Attestor passed through the attestation process.

specified in Quote, and it sends the authorization signed by AuTK to the Host-TPM. Figure 7.3 shows the particular code to generate and sign the combined policy. The complete codes are accessible at: [31].

Once the authorization is received, the Host-TPM sends it to the TPM to unlock the SeK. The TPM verifies the signature and checks that it meets the requirements of the authorization (RESET and PCR values), being able to use the SeK until these requirements change, i.e., until it reboots or until a new or modified program is used. If the PCR value changes, the Attestor will request a new authorization from the Verifier; however, if a program that was not supposed to be executed is run, the Attestor will not obtain a new authorization until it is restarted and ask for a new one. A device will not be able to reuse an old authorization because those are sealed to the RESET value of the TPM, which prevents the use of authorizations for old decommissioned software.

7.4.5. Daily Life Functionality

Whenever the edge device proves ownership of the SeK, the TPM will first verify the signature of the policy provided by the server, secondly, it will assert the signed policies (values of PCR and Reset), and finally, it will grant access to sign with the SeK, proving the correctness of the software, and therefore, no additional attestation is needed. Figure 7.4 shows the script using the SeK to sign the measure of a pressure sensor satisfying the policy that was set when the SeK was generated in Section 7.4.3.2 (authorization policy in Figure 7.2).

This key can also be used to prove authentication. In protocols such as the well-known cryptography protocol Transport Layer Security (TLS) [32], or signing transactions for blockchain, asymmetric keys are used for the authentication, thus; the Sealed key pair can be used to prove correctness at the same time that authentication. The end user can trust in the SeK because it is coming with a certificate from a trusted third party, which is commonly the Verifier itself.


```
pi@raspberrypi:~$ sudo sh test_SeK.sh
name: 000b9146cfb4a9c60d8eacde73ecf4b729aa17aaa3d58611a5db88903daa75704d44
c83a71dba04be92b2bb3a0b47c437ef3540ffd193d7236ad30a9168146ded143
5291ea4bb3ae260837aa280477b84ecd9410e8bb520195f306a87c790f3ecba7
verifying signature
Satisfying policies
c83a71dba04be92b2bb3a0b47c437ef3540ffd193d7236ad30a9168146ded143
5291ea4bb3ae260837aa280477b84ecd9410e8bb520195f306a87c790f3ecba7
policies satisfied
19c004bbff8d386b26e1ade115fa1acdaf8ca2ee9e39067662a24978ba1f43c2
```

Figure 7.4: Output of the script using the sealed key to sign a measured value after satisfying the policies.

7.4.6. Implementation

For implementing our system, we employ a low-price System on Chip (SoC), Raspberry Pi 4B: Broadcom BCM2711, Quad core Cortex-A72 (ARM v8) 64-bit SoC @ 1.5 GHz 8 GB LPDDR4-3200 SDRAM and a pressure sensor DPS310 Pressure Shield2Go [33]. Additionally, we have used a TPM IRIDIUM9670 TPM2.0 LINUX [34], a hardware security module by Infineon Technologies GMBH specifically designed for Raspberry Pi SoCs. In Figure 7.5 is shown the setup. Our implementation lacks CTRM, but it is a feasible solution for the real-world market [35].

The software has been developed over WenXin’s code, which is available in a public repository [36]. This software presents a classic remote attestation with Attestor and Verifier. We have used this repository as a baseline for the implementation of our presented novelties, and the result can be found in [37] for the Attestor and [31] for the Verifier.

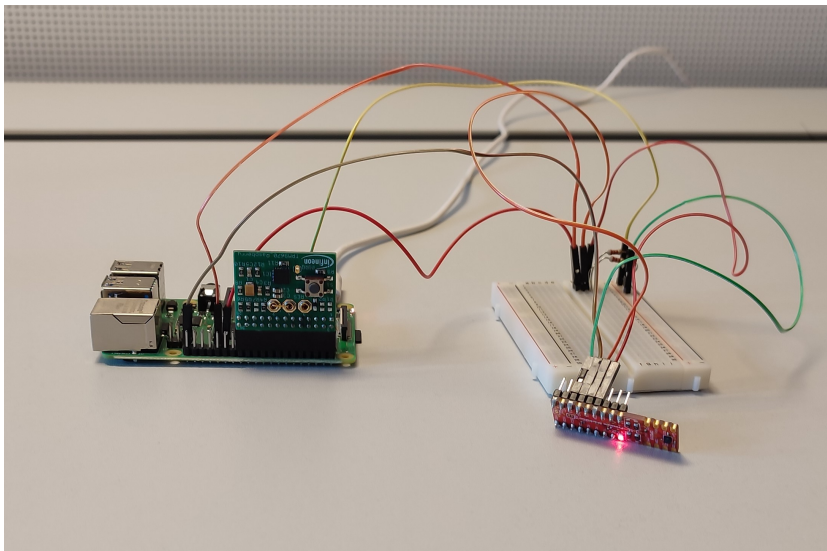


Figure 7.5: Experimental setup. Raspberry Pi 4 with TPM (green hardware) and pressure sensor (red hardware).

7.5. RESEKRA Use Cases

There are several ways to use RESEKRA, depending on whether the edge device programmer is the same as the manufacturer, regarding how the Verifier obtains the RML, how it obtains the trust on the sealed key, or who takes the role of the Verifier. In the process, we consider that there is a trusted system to provide the identity of the ED devices to the end user. Two use cases were considered for RESEKRA: edge computing as a trusted service and edge computing as a trusted service—No TTP online.

In the case of edge computing as a trusted service, we have the case of a company that offers edge computing for a fee. This company is responsible for the deployment and maintenance of edge nodes. Additionally, in this use case, we can find the manufacturers of the edge devices, which are trusted by all. The Verifiers, in this case, are a cloud service with high latency and some downtime. Finally, we find the end users who rely on the Verifiers to use the reliable edge device services requiring very low latency.

The second use-case, edge computing as a trusted service—No TTP online, is similar but with a fundamental difference: there are no cloud services offering the Verifier service.

There are other possible use cases where the Reference Measurement List is trusted through a voting method or is provided along with the edge device software from a public repository such as [38]. Edge devices that have already been validated can also be used as a Verifier.

7.5.1. Edge Computing as Trusted Service

In edge computing as a trusted service, we have (1) the trusted entity in charge of designing, programming, and manufacturing the edge devices for edge computing (Trusted Designers); (2) several semi-trusted investors in charge of deploying and maintaining the edge devices (Semitrusted Maintainers); (3) Trusted Clouds and (4) mobile end users, e.g., a car.

7.5.1.1. Roles

The Trusted Designers are responsible for designing an edge device with CRTM and TPM. They are also responsible for designing the secure software. Finally, they manufacture the product and store all necessary certificates on the edge device.

Semitrusted Maintainers are entities with little technical knowledge. They are semi-trusted because they have no interest in making attacks into the system, but they have no knowledge of how to prevent them. They have the capacity and interest to deploy and maintain an edge device in a specific area, such as a non-profit interested organization, e.g., a government, or an interested profit organization with a business model based on subscriptions

or blockchain using utility tokens, e.g., Helium [39].

The Semitrusted Cloud (SC) is responsible for verifying and signing the authorizations for the SeKs of edge devices. Every end user has a set of trusted SC. The SC is semitrusted because the end user only trusts some of them.

The last role is the end user, who needs to trust edge devices to use edge computing services.

7.5.1.2. Process

First, Semitrusted Maintainers purchase an Edge-Device N (ED_N) $\forall N \in \mathbb{N}$ from Trusted Designers and deploy it wherever they see a fit. Thanks to the remote and versatile key creation system in RESEKRA, there can be several independent SCs verifying the integrity and authenticity of the received documents. These SCs can belong to third parties. The ED_N communicates with a set O_N of SCs, sends them the reference measurement list and the manufacturing certificates signed by the Trusted Designers. A subset P_N of O_N approves the Section 7.4.3 Remote Enrollment and issues a certificate with the SeK generated and ED_N 's identifier. The identifier has to be an universally unique identifier [40] to avoid Wormhole attacks [41], which is explained in more detail in Section 7.6.3. In our scenario, we propose using the last 128 bits of the hash of the EK_PuB as an identifier.

Because the process Section 7.4.3.1 Edge device validation has to be realized just once per Verifier, and the processes Section 7.4.3.2 Sealed Key creation and Section 7.4.3.3 Sealed Key validation are performed just in unusual moments, normally after each reset, the interactions between ED_N and SCs are infrequent. Therefore, the number of Verifiers in the subset P_N will not affect the service quality. Moreover, the SeKs are stored in the virtual memory of the TPM.

The edge device subsequently requests a Section 7.4.4 Attestation to every SC in P_N . A subset Q_N of P_N approves the attestation and grants the authorization to use the SeKs, as long as the device is not rebooted or its software is not modified. Additionally, a time limit can be added to this authorization.

At this moment, the ED_N can act as a Verifier for the nearest IoT devices by using a software-based remote attestation such as Software-based ATTestation (SWATT) [15], where the time response is essential, and therefore, a one-hop network is a fundamental requirement. We do not provide further details here, since it is beyond the scope of this paper.

Finally, end users, by requesting the services of ED_N , have to provide a priority organized set G of SCs and a Random Nonce (RN). If any of the elements of G belong to Q_N , the ED_N will sign the RN using the SeK approved by the most priority SC belonging to G and Q_N , and it will furnish

the corresponding certificate. Since the identity in the certificate is based on the EK_PuK , the identity provider just has to provide one identifier for ED_N , and it is valid for all the ED_N 's SeKs. Otherwise, the ED_N is considered untrusted by the end user.

When ED_N is trusted, the end user requests the edge computing services (gathering data or processing data) and measures the required time of the service. At the end of the service, the ED_N signs the hash of the RN with the results of the service using the SeK. If the final signature never arrives or takes longer than expected, the end user does not trust the results and reports the irregularity.

However, it needs to check the correctness of the edge devices before using their services. The car only has the name of the edge devices and a set of cloud services that are trusted for this car but are not accessible in real time.

Figure 7.6 shows an scenario where a car enters in a location with an edge computing network, and it needs edge services from 3 EDs such as gathering data from sensors or using computer processing. However, it needs to check the correctness of the edge devices before using their services. The car only has the identifier of the edge devices and a set of SCs G that are not accessible in real time.

Firstly, the car requests a signed RN from the three EDs, each one with three different subsets Q , but only ED_1 and ED_3 have passed through a remote attestation of at least one of the SCs belonging to G . Then, by simply signing the RN with their SeKs, ED_1 and ED_3 prove their correctness, and the car puts the trust in them and starts using their services.

7.5.2. Edge Computing as Service—No TTP Online

In edge computing as service—No TTP online, we consider the need of avoiding the use of a Semitrusted Cloud. Perhaps, because the internet connection is not available, there are not SCs of G belonging to Q or simply to avoid the use of a TTP.

In this field, we have the same roles as in the previous case, except for the absence of SC.

Processes

When the end user wants to use the services of the edge device, and there is not any trusted third party available to act as a Verifier, the end user himself can act as a Verifier.

The end user himself realizes the Section 7.4.3 Remote Enrollment. Then, the ED provides the RML signed by the ED's programmer. If ED passes the Attestation Section 7.4.4, the end user will authorize the use of the SeK and requests the Edge computing services (gathering data or processing

data). Upon completion of the service, the ED will use the SeK to sign the service result.

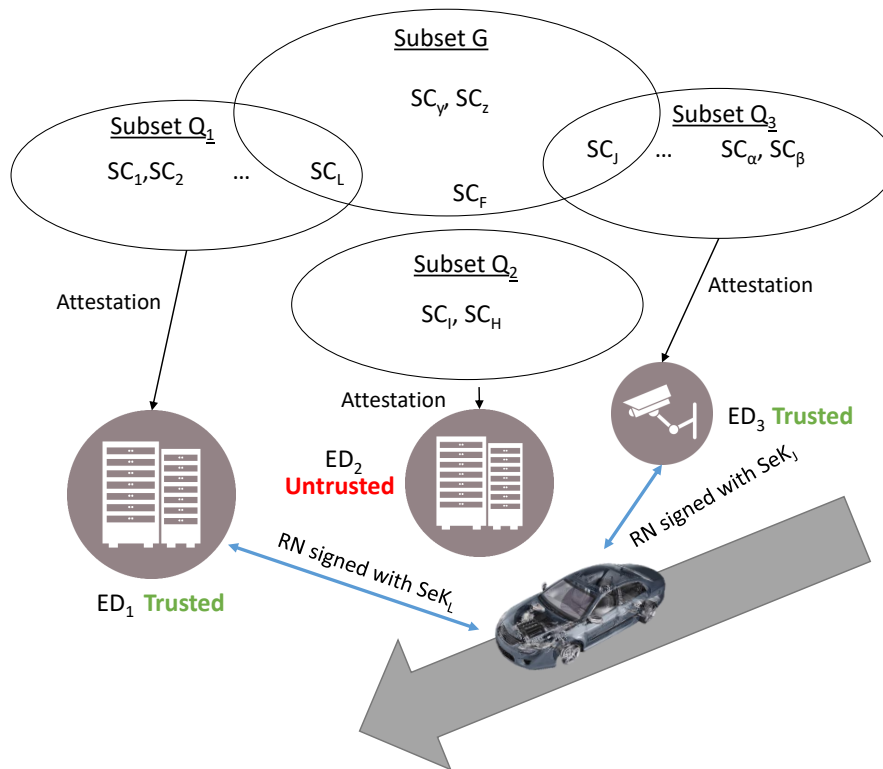


Figure 7.6: Visualization of edge computing as trusted service. A car reaching a location needs the local services of several edge and IoT devices. All these devices have many SeKs available validated by an SC each. The car just trusts in some of the SCs, those which belong to subset G.

Throughout the communication process, a constant status of the software is ensured by continuously proving the SeK ownership. The process of document verification and SeK creation should be created beforehand to avoid delaying the start of the edge computing service. Figure 7.7 represents the differences between the two use-cases presented in this section. On the right of the image, all the phases (in purple) are processed between car and edge device, and there is no need of third party acting as a Verifier.

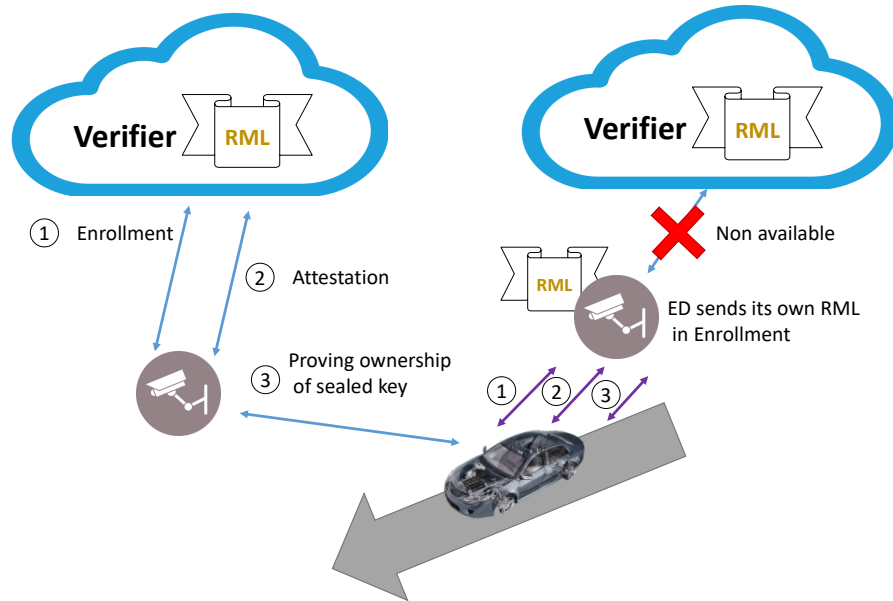


Figure 7.7: Visualization of the two use cases, edge computing as service (in blue) and edge computing as service—No TTP online (in purple).

7.6. Security Analysis

In this section, we provide an analysis of the most common attacks on remote attestation present in the state of the art. We do not consider Denial of Service in all the analysis.

7.6.1. Man-In-the-Middle Attack (MIM attack)

The entire RESEKRA scheme was developed with MIM attacks in mind. The most sensitive section of RESEKRA is Section 7.4.3 Remote Enrollment, since we consider the Host-TPM untrusted for the remote communication.

All communications between the ED and Verifier are protected with TLS, but we take into account that even in this communication, an MIM can obtain complete control of Host-TPM. All communication between the Verifier and TPM goes through the Host-TPM. The attacker would be able to read all certificates and information transferred, but it would not be able to modify it, because it is signed by the TPM. Moreover, even with the complete control of the Host-TPM, the attacker cannot force the TPM to sign fake certificates or quotes because all the TPM generated data are signed by a restricted key (AK).

The AK is proved to be a restricted key by using the Endorsement Key. The attacker could obtain knowledge of Auth_PuK and generate a

fake SeK out of the TPM and store it in the TPM. Following this process, the attacker could generate a Cert_SeK signed by the AK, but it would fail while asserting the SeK attributes Section 7.4.3.3. Therefore, all Section 7.4.3 Remote Enrollment is protected from MIM attack.

7.6.2. Impersonation and Replay Attack

An attacker could obtain the certificates stored in Host-TPM, but it would not be able to prove EK ownership and thus would fail to perform the Section 7.4.3 Remote Enrollment.

Validation of AK and SeK is done through Auth_PuK, which is randomly generated. Therefore, we avoid replay attacks in Section 7.4.3 Remote Enrollment.

Note that Section 7.4.4 Attestation is realized with an RN and signed by AK; therefore, replay attacks and impersonation are avoided.

Finally, to show the correct status of the software to end users, the SeK is used to sign RNs, avoiding replay attacks.

7.6.3. Wormhole Attack

This attack was introduced by Yih-Chun Hu [41]. In this attack, the attacker uses legitimate data or information from an edge computing network at a given location and “tunnels” it in some way to another edge computing network without adding appreciable delay. The attack will then be explained, which is followed by a possible defense strategy:

1. The attacker compromises one edge device close to the end user (ED_C).
2. The attacker tunnels the network of ED_C with another far network with an ED from the same deployer (ED_f).
3. The end user starts the authentication method with ED_C and passes through because it is talking with the expected node.
4. When the attacker receives a random nonce to be signed with SeK, it tunnels this random nonce to ED_f , obtains the signature and certificates ED_f 's SeK, and sends them back to the end user.
5. Finally, the end user trusts ED_C without access to ED_C 's SeK.

Since RESEKRA relies on asymmetric cryptography (EK, AK, and SeK) that is randomly generated and is therefore unique, it is easy to avoid this kind of attack.

To avoid this attack, the certificate issued from SCs shall include the universally unique identifier [40] (ID) of ED_C , and this ID should be provided to

the end user as part of the ED_C 's identification information. In Section 7.5.1, we use the last 128 bits of the hash of the EK_PuB as the ID.

When verifying the SeK's certificate, the end user will find out that the SeK of ED_f does not belong to ED_C .

7.6.4. Interference Attacks

In the interference attack, an MiM provides a false attestation response (Quote) to Verifier. It will not pass the process, and ED will be considered untrustworthy, even though the Verifier was unable to verify the actual Quote.

The Quote is coming with a signature and RN. When a false Quote is provided, the Verifier will recognize it is not coming from the real ED and will refuse it without affecting the attestation process.

7.6.5. Time-Of-Check-To-Time-Of-Use Attack

Time-Of-Check-To-Time-Of-Use (TOCTTOU) is a present vulnerability in Remote Attestation scenarios [9]. In TOCTTOU, the ED provides evidence of having the appropriate software at the time of the Attestation, but it uses different software when the ED is about to provide services.

At a general level, this attack should be avoided by proving access to the SeK at the moment ED provides the services.

One possible attack would be to have the correct software when proving ownership of the SeK at the beginning of the service and then calling a manipulated program when providing the real services. This attack should not be possible in an IMA that is bugs-free because any program able to call a manipulated program would not pass the attestation.

However, if in an unlikely situation were to happen, the ED will not be able to provide the second signature of the result, and thus, the end user will notice this problem when receiving the result. If so, the ED owner can initiate a fast debugging process.

7.6.6. Verifier-Based DoS Attack

In Verifier-Based DoS (VBDoS) attack, the attacker acts as a Verifier and requests many attestations to the ED, which becomes overloaded and cannot provide its normal service.

The remote enrollment and attestation from ED to SC is started by the ED, and they are not routine procedures.

The attack could come from the end user when it is in charge of attesting the ED through the full RESEKRA process or just validating the use of the SeK. The solution for this attack should be provided by the ED owner as part of the normal DoS attacks to their EDs. It is the responsibility of the ED owner to develop a scheme for providing ED services to the end user,

allocate resources, and control and avoid misuses or malicious uses of the ED resources by the end user.

7.6.7. Non-Verification DoS Attack

It is a new kind of attack developed by us. In this attack, the attacker is able to interrupt the Verifier service on an edge computing network. It could happen through attacking one of the hops in the connection between the Verifier and Attestor or with a DoS attack to the Verifier.

In a common remote attestation scheme, throughout the interruption, the end user cannot trust the EDs, and even when the EDs are in a correct state, if trust is essential, the attacker produces a DoS to the local edge computing infrastructure.

Due to the decentralized Verifier infrastructure in RESEKRA, a DoS to multiple Verifiers is very unlikely. If the connection between EDs and Verifier is broken, the ED still has access to the SeK; therefore, it can still prove its correct status. If by an unlikely circumstance, the ED loses access to the SeK, several local Verifiers can be run-time assigned, or the end user itself can perform the remote attestation. To the best of our knowledge, our system is the first remote attestation protocol able to resist this attack.

7.7. Conclusions

In this paper, a new process for remote attestation focused on edge computing has been presented. The process, called RESEKRA, has been designed and implemented in a real sensor, introducing new features with regard to the SoA. The first characteristic is the remote enrollment, when an IoT device can connect with a new Verifier without having to know a shared secret, allowing Attestors and Verifiers to dynamically join and leave the structure. The second special feature of RESEKRA is the use of asymmetric keys sealed to the correct state of the software, which allows the correct state of the device to be proven directly to the end user while reducing the workload on the Verifier and allowing the IoT sensor to operate with intermittent activity by requesting a remote attestation when it deems it necessary.

These qualities enable the sought-after decentralized verification system in the SoA, integrate remote attestation into Mobile Edge Computing, and finally, allow the end user to trust the edge devices without any trusted authority or entity within the edge. It allows for further research into edge computing by taking for granted the trust of the devices in the environment. Furthermore, RESEKRA can also have a significant academic impact on other lines of research in IoT computation where trust in IoT devices is crucial, e.g., supply chain, IoT in blockchain, or Industry 4.0.

On the other side, RESEKRA—as with all Remote Attestation systems based on RoT—requires a reliable manufacturer to produce the trusted device, but then, RESEKRA can be used in many other scenarios inside and outside Mobile Edge Computing. It can be used from applications as simple as smart homes—where a simple set up is essential—attesting the end users’s devices using a mobile phone. Other applications as complex as Industry 4.0 use several Verifiers from the same owner to avoid the highly feared DoS attacks in manufacturing affairs. It can also be used for lower consumption applications, where sensors operate with intermittent activity to save battery life. In general, RESEKRA and therefore its results can be applied in any scenario beyond the use cases presented where the assumptions detailed in the introduction are met.

In other respects and future work, we envision integrating the Direct Anonymous Attestation Scheme. It would allow proving ownership of an SeK without giving any additional device identity information. This would preserve the privacy of the Attestor and enable the attestation of privacy-sensitive devices such as cars or mobile phones. On the other side, we are currently working to delete the use of private keys in the Verifier and move it to a smart contract in the blockchain, making the blockchain itself being able to attest the IoT devices before accepting a transaction.

Nomenclature

The following abbreviations are used in this manuscript:

IMA	Integrity Measurement Architecture
CTRM	Core Root of Trust of Measurement
AK	Attestation Key
EK	Endorsement Key
SeK	Sealed Key
KP	Key Pair
PuB	Public Key
Pubdata	Public information of a key (public key, policies and attributes)
TPM	Hardware security module following TPM standard
Host-TPM	Device owner of TPM
TPM_GENERATED	4 bytes-code “0xff544347”. Files starting with this code and signed by a restricted key were generated by a TPM
PCR	Platform Configuration Registers
ML	Measurement List
RML	Reference Measurement List
Cert_SeK	Attestation certificate of Sealed Key
ED_N	Edge device number N

SC	Semitrusted Cloud
O_N	Set of SCs ED_N communicate with
P_N	Set of SCs that accept the Remote enrollment of ED_N
Q_N	Set of SCs that accept the Attestation of ED_N
G	Set of priority organized SCs trusted by the end user

References

- [1] Cisco, U. *Cisco Annual Internet Report (2018–2023) White Paper*; Cisco: San Jose, CA, USA, 2020.
- [2] Shi, W.; Cao, J.; Zhang, Q.; Li, Y.; Xu, L. Edge Computing: Vision and Challenges. *IEEE Internet Things J.* **2016**, *3*, 637–646. <https://doi.org/10.1109/JIOT.2016.2579198>.
- [3] Liu, D.; Yan, Z.; Ding, W.; Atiquzzaman, M. A survey on secure data analytics in edge computing. *IEEE Internet Things J.* **2019**, *6*, 4946–4967.
- [4] Roman, R.; Lopez, J.; Mambo, M. Mobile edge computing, Fog et al.: A survey and analysis of security threats and challenges. *Future Gener. Comput. Syst.* **2018**, *78*, 680–698. <https://doi.org/10.1016/j.future.2016.11.009>.
- [5] (ISG), M.E.C.M.E.I.S.G. Mobile Edge Computing; Market Acceleration; MEC Metrics Best Practice and Guidelines. Available online: https://www.etsi.org/deliver/etsi_gs/MEC-IEG/001_099/006/01.01.01_60/gs_MEC-IEG006v010101p.pdf (accessed on 23 May 2022).
- [6] Xiao, Y.; Jia, Y.; Liu, C.; Cheng, X.; Yu, J.; Lv, W. Edge Computing Security: State of the Art and Challenges. *Proc. IEEE* **2019**, *107*, 1608–1631. <https://doi.org/10.1109/JPROC.2019.2918437>.
- [7] Abbas, N.; Zhang, Y.; Taherkordi, A.; Skeie, T. Mobile Edge Computing: A Survey. *IEEE Internet Things J.* **2018**, *5*, 450–465. <https://doi.org/10.1109/JIOT.2017.2750180>.
- [8] Ambrosin, M.; Conti, M.; Lazzeretti, R.; Rabbani, M.M.; Ranise, S. Collective Remote Attestation at the Internet of Things Scale: State-of-the-art and Future Challenges. *IEEE Commun. Surv. Tutorials* **2020**, *22*, 2447–2461.
- [9] Raducu, R.; Rodríguez, R.J.; Álvarez, P. Defense and Attack Techniques Against File-Based TOCTOU Vulnerabilities:

A Systematic Review. *IEEE Access* **2022**, *10*, 21742–21758. <https://doi.org/10.1109/ACCESS.2022.3153064>.

- [10] Trusted Computing Group *TPM 2.0 Library*; 2021. Available online: https://www.etsi.org/deliver/etsi_gs/MEC-IEG/001_099/006/01.01.01_60/gs_MEC-IEG006v010101p.pdf (accessed on 23 May 2022).
- [11] Calvo, M.; Beltrán, M. Remote Attestation as a Service for Edge-Enabled IoT. In Proceedings of the 2021 IEEE International Conference on Services Computing (SCC), Chicago, IL, USA, 5–10 September 2021; pp. 329–339. <https://doi.org/10.1109/SCC53864.2021.00046>.
- [12] Tan, H.; Tsudik, G.; Jha, S. MTRA: Multi-Tier randomized remote attestation in IoT networks. *Comput. Secur.* **2019**, *81*, 78–93. <https://doi.org/10.1016/j.cose.2018.10.008>.
- [13] Tan, H.; Hu, W.; Jha, S. A TPM-enabled remote attestation protocol (TRAP) in wireless sensor networks. In Proceedings of the 6th ACM Workshop on Performance Monitoring and Measurement of Heterogeneous Wireless and Wired Networks, Miami, FL, USA, 31 October 2011; pp. 9–16.
- [14] Wang, Y.; Attebury, G.; Ramamurthy, B. A survey of security issues in wireless sensor networks. *IEEE Communications Surveys & Tutorials* **2006**, *8*, 2–23. doi:10.1109/COMST.2006.315852.
- [15] Seshadri, A.; Perrig, A.; Van Doorn, L.; Khosla, P. SWATT: Software-based attestation for embedded devices. In Proceedings of the IEEE Symposium on Security and Privacy, Berkeley, CA, USA, 12 May 2004; pp. 272–282.
- [16] Spinellis, D. Reflection as a mechanism for software integrity verification. *ACM Trans. Inf. Syst. Secur. (TISSEC)* **2000**, *3*, 51–62.
- [17] Seshadri, A.; Luk, M.; Perrig, A.; Van Doorn, L.; Khosla, P. SCUBA: Secure code update by attestation in sensor networks. In Proceedings of the 5th ACM workshop on Wireless Security, Los Angeles, CA, USA, 29 September 2006; pp. 85–94.
- [18] Francillon, A.; Nguyen, Q.; Rasmussen, K.B.; Tsudik, G. Systematic Treatment of Remote Attestation. Cryptology ePrint Archive, Report 2012/713. 2012. Available online: <https://ia.cr/2012/713> (accessed on 4 July 2022).

- [19] Aman, M.N.; Basheer, M.H.; Dash, S.; Wong, J.W.; Xu, J.; Lim, H.W.; Sikdar, B. HAtt: Hybrid remote attestation for the Internet of Things with high availability. *IEEE Internet Things J.* **2020**, *7*, 7220–7233.
- [20] Ibrahim, A.; Sadeghi, A.R.; Zeitouni, S. SeED: Secure non-interactive attestation for embedded devices. In Proceedings of the 10th ACM Conference on Security and Privacy in Wireless and Mobile Networks, Boston, MA, USA, 18–20 July 2017; pp. 64–74.
- [21] Eldefrawy, K.; Tsudik, G.; Francillon, A.; Perito, D. Smart: Secure and minimal architecture for (establishing dynamic) root of trust. In Proceedings of the Ndss, San Diego, CA, USA, 5–8 February 2012; Volume 12, pp. 1–15.
- [22] Carpent, X.; Tsudik, G.; Rattanaivanon, N. ERASMUS: Efficient remote attestation via self-measurement for unattended settings. In Proceedings of the 2018 Design, Automation & Test in Europe Conference & Exhibition (DATE), Dresden, Germany, 19–23 March 2018; pp. 1191–1194.
- [23] Koeberl, P.; Schulz, S.; Sadeghi, A.R.; Varadharajan, V. TrustLite: A security architecture for tiny embedded devices. In Proceedings of the Ninth European Conference on Computer Systems, Amsterdam, The Netherlands, 14–16 April 2014; pp. 1–14.
- [24] Carpent, X.; ElDefrawy, K.; Rattanaivanon, N.; Tsudik, G. Light-weight swarm attestation: A tale of two lisa-s. In Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security, Abu Dhabi, United Arab Emirates, 2–6 April 2017; pp. 86–100.
- [25] Dushku, E.; Rabbani, M.M.; Conti, M.; Mancini, L.V.; Ranise, S. SARA: Secure asynchronous remote attestation for IoT systems. *IEEE Trans. Inf. Forensics Secur.* **2020**, *15*, 3123–3136.
- [26] Safford, D.; Kasatkin, D.; mzohar. Integrity measurement architecture (IMA), 2013. Available online: <https://sourceforge.net/projects/linux-ima/> (accessed on 4 July 2022).
- [27] Cooper, D.; Polk, W.; Regenscheid, A.; Souppaya, M. BIOS protection guidelines—NIST, 2011. National Institute of Standards and Technology. <https://nvlpubs.nist.gov/nistpubs/legacy/sp/nistspecialpublication800-147.pdf> (accessed on 7 July 2022)
- [28] Trusted Computing Group[™]. Endorsement Key (EK) and Platform Certificate Enrollment Specification. 2013. Available online: <https://trustedcomputinggroup.org/wp-content/uploads/IWG-EK-CMC-enrollment-for-TPM-v1-2-FAQ-rev-April-3-2013.pdf> (accessed on 20 May 2022).

- [29] Trusted Computing Group. TCG Trusted Attestation Protocol (TAP) Information Model for TPM Families 1.2 and 2.0 and DICE Family 1.0 Revision 0.36. https://trustedcomputinggroup.org/wp-content/uploads/TNC_TAP_Information_Model_v1.00_r0.36-FINAL.pdf (accessed on 4 July 2022)
- [30] Arthur, W.; Challener, D.; Goldman, K. *A Practical Guide to TPM 2.0: Using the New Trusted Platform Module in the New Age of Security*; Springer Nature: London, UK, 2015.
- [31] Ernesto. RESEKRA: Verifier. Available online: https://github.com/ernesto418/Remote_attestation_server/ (accessed on 21 June 2022).
- [32] Dierks, T.; Rescorla, E. The Transport Layer Security (TLS) Protocol Version 1.2. RFC5246. 2008. Available online: <https://www.rfc-editor.org/info/rfc5246> (accessed on 20 May 2022).
- [33] Technologies, I. DPS310 Pressure Shield2Go. Available online: <https://www.infineon.com/cms/en/product/evaluation-boards/s2go-pressure-dps310/> (accessed on 21 June 2022).
- [34] Technologies, I. IRIDIUM9670 TPM2.0 LINUX. Available online: <https://www.infineon.com/cms/en/product/evaluation-boards/iridium9670-tpm2.0-linux/> (accessed on 23 May 2022).
- [35] Mode, M. Secured Boot for ARM Processor Platforms. Available online: https://www.infineon.com/dgdl/Infineon-ISP-Use-Case-Secured-boot-for-ARM-processor-plaforms-ABR-v01_00-EN.pdf?fileId=5546d4625bd71aa0015c0b1fcee0851 (accessed on 20 May 2022).
- [36] Leong, W.; Yushev, A. OPTIGA™ TPM Application Note Remote Attestation. Available online: <https://github.com/Infineon/remote-attestation-optiga-tpm> (accessed on 23 May 2022).
- [37] Ernesto. RESEKRA: Attestor. Available online: https://github.com/ernesto418/Remote_attestation_TPM/ (accessed on 21 June 2022).
- [38] Linux Kernel. Available online: <https://github.com/torvalds/linux> (accessed on 20 May 2022).
- [39] Helium ©, People-Powered Network, 2021 Helium Systems Inc. Available online: <https://www.helium.com/> (accessed on 28 January 2022).
- [40] Leach, P.; Mealling, M.; Salz, R. A universally unique identifier (uuid) urn namespace. *RFC 4122* **2005** <https://www.rfc-editor.org/rfc/rfc4122> (accessed on 4 July 2022).

-
- [41] Hu, Y.C.; Perrig, A.; Johnson, D.B. Wormhole attacks in wireless networks. *IEEE J. Sel. Areas Commun.* **2006**, *24*, 370–380.

Part III

Conclusions

Chapter 8

Conclusions

The science of today is the technology of tomorrow.

Edward Teller

This thesis offers a solution to the current blockage that IoT technology encounters in its development and implementation due to the lack of cybersecurity. In order to achieve the required trust in IoT devices and their data, the thesis identifies and describes the security requirements necessary to obtain information security in IoT: confidentiality, fine-grained authorization, integrity, authenticity, trustworthiness, traceability and freshness. All these security requirements are achieved in this PhD through the use of intensive use of cryptography, HSM integrated into IoT devices and using of blockchain. Also, the manuscript has more than one solution to some security requirements to increase versatility. The following explains how the achievements — explained in Chapter 3 — solve each of the security requirements:

- Integrity: to reach a secure data transfer in IoT or any other telecommunication ecosystem, data integrity is essential and basic. With the help of cryptography, hashes and digital signatures, the integrity is accessible. This thesis attains these features in multiple ways protecting the data hash with the algorithms ECDA or uploading the hash to blockchain. These methods are as reliable as the authentication of the identities in the system.
- Authentication: this feature typically involves human intervention in usual ecosystems, like the one providing digital certificates to humans or websites. However, this procedure is not suitable in IoT due to the challenge of identifying devices by a human and the number of machines to identify. This problem is solved in [1] through the use of a novel concept called Smart Certificate Authorities. With this proposal, a smart contract verifies the devices's identities, owners, and features.

- **Traceability:** through the use of blockchain and immutable transactions, the data signed by the IoT device is easily tracked back to the device that generated it. Additionally, keeping an auditable and updated list of the owners of the devices, as explained in [2], makes it possible to even know who was responsible for the IoT device at the moment the data was generated.
- **Confidentiality:** it is one of the drawbacks of using blockchain. In [2], we developed and implemented a solution based on HL. Firstly, the solution takes advantage of the consortium blockchain to keep the confidentiality within the stakeholders. Secondly, the use of the Private Collections to keep confidentiality between the organizations to which the stakeholders belong.
- **Fine-grained authorization:** while confidentiality is currently a challenge in blockchain technologies, fine-grained authorization is a challenge in all huge data databases with multiple readers. One of the branches of the SoA seeks to solve it through the use of cryptography ABE. However, it has several drawbacks. In our work [3], we analyzed them and proposed a solution for each of them by mixing other cryptography protocols tools and creating our own revocation mechanism, and thus achieving a secure fine-grained authorization.
- **Trustworthiness:** the thesis proposes two different solutions to ensure the trustworthiness of the device that generated the data. The first one is using a SS as explained in [1]. The SS and the infrastructure build around it can protect the IoT device and then, its data. The second solution is remote attestation. With the use of a more complex IoT device, the work [4] found a method to apply remote attestation to IoT devices and in this way to verify its software status.
- **Freshness:** this is one of the most complex challenges in the SoA, verifying when a measurement was gathered without having contact with the IoT device. The work [1] accomplishes this by statistically analyzing an attacker's capacity to predict the blockhashes of Ethereum. Then, it uses this number in the SS as a nonce to ensure the freshness.

This doctoral thesis introduces numerous innovative solutions to address the problem of the insecurity of the devices. It marks a step forward the accomplishment of a secure IoT environment, which, in turn, leads to the increase of automation and quality of human products and activities.

8.1. Future trends

Finally, as a result of the experiences gained in this investigation, we venture to predict the main trends in IoT security:

- The logistics of products and consumer goods is a complex field with great potential. Both the complexity and the potential will increase due to the growth of available data due to the continuing deployment of IoT devices in the industry and the increased data-sharing between the factories. This will make big data technologies more relevant to the industry on its way to Industry 4.0.
- Edge computing, although an old branch, is not a mature enough technology to be implemented in a secure way. Many researchers are finding new use cases for it, and at the same time, there is also much research focusing on enhancing the security of this complex field. In the future, the security gaps will be so low that the today's timid experimentation will bear great fruit and set examples to follow. Thus, there will be a disruptive point where edge computing will be introduced quickly in the real world. Then, smart things will be commonplace in our vicinity, traffic lights, street lights, computers, police drones, etc.
- Blockchain is a highly controversial technology due to the scams and speculation that have been associated with it. However, at the same time, blockchain is a very promising technology. Over time, more secure environments are being created in conjunction with the development of decentralized applications, which will facilitate the creation and implementation of future applications. Eventually, blockchain will be known more for the vents it brings to society than as a form of speculation.

References

- [1] Ernesto Gómez-Marín, Luis Parrilla, José L. Tejero López, Diego P. Morales, and Encarnación. Castillo. Towards sensor measurement reliability in blockchain. *UNDER REVIEW by Sensors*, 2023.
- [2] Ernesto Gómez-Marín, Valerio Senni, Luis Parrilla, Jose L. Tejero López, Encarnación Castillo, and Davide Martintoni. An innovative strategy based on secure element for cyber-physical authentication in safety-critical manufacturing supply chain. *Applied Sciences*, 13(18), 2023.
- [3] Ernesto Gómez-Marín, Davide Martintoni, Valerio Senni, Encarnación Castillo, and Luis Parrilla. Fine-grained access control with user revocation in smart manufacturing. *Electronics*, 12(13), 2023.
- [4] Ernesto Gómez-Marín, Luis Parrilla, Gianfranco Mauro, Antonio Escobar-Molero, Diego P. Morales, and Encarnación Castillo. Resekra: Remote enrollment using sealed keys for remote attestation. *Sensors*, 22(13), 2022.

*Now this is not the end.
It is not even the beginning of the end.
But it is, perhaps, the end of the beginning.*

Winston Churchill

