



**Catarina Alexandra
Correia Da Silva**

**Permissão Para Partilha Seletiva em Ambientes IoT
Selective Sharing Permissioning in IoT Environments**



Catarina Alexandra
Correia Da Silva

Permissão Para Partilha Seletiva em Ambientes IoT
Selective Sharing Permissioning in IoT Environments

“Historically, privacy was almost implicit, because it was hard to find and gather information. But in the digital world, whether it’s digital cameras or satellites or just what you click on, we need to have more explicit rules - not just for governments but for private companies.”

— Bill Gates



Universidade de Aveiro
2021

**Catarina Alexandra
Correia Da Silva**

Permissão Para Partilha Seletiva em Ambientes IoT
Selective Sharing Permissioning in IoT Environments

Dissertação apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Engenharia de Computadores e Telemática, realizada sob a orientação científica do Doutor João Paulo Silva Barraca, Professor auxiliar do Departamento de Eletrónica, Telecomunicações e Informática da Universidade de Aveiro, e do Doutor Vítor Jesus Silva, Professor auxiliar na School of Business da Aston University.

This work is partially funded by
NGI Trust, with number 3.85, Pro-
ject CASSIOPEIA.

o júri / the jury

presidente / president

Professor Doutor Arnaldo Silva Rodrigues de Oliveira

Professor Auxiliar do Departamento de Eletrónica, Telecomunicações e Informática, Universidade de Aveiro

vogais / examiners committee

Professor Doutor Miguel Filipe Leitão Pardal

Professor Auxiliar no Instituto Superior Técnico, Universidade de Lisboa

Professor Doutor João Paulo Silva Barraca

Professor Auxiliar do Departamento de Eletrónica, Telecomunicações e Informática, Universidade de Aveiro (Orientador)

agradecimentos / acknowledgements

Em primeiro lugar, agradeço aos meus orientadores, Prof. Doutor João Paulo Barraca e ao Prof. Doutor Vítor Jesus pela oportunidade de desenvolver este trabalho e por todo o apoio dado ao longo destes meses. Agradeço também ao Doutor Gilad Rosner por toda a ajuda e por todo o tempo que disponibilizou para a realização deste trabalho. Aproveito também para agradecer ao IT por fornecer os recursos e condições para a realização desta Dissertação.

Agradeço ao meu noivo, Mário Antunes, a quem com muito amor dedico este trabalho, por toda ajuda e compreensão. Agradeço também à minha família, em especial à minha mãe, Maria João Correia, por me ter dado toda a ajuda para percorrer este caminho.

Por fim, agradeço às minha amigas por me ouvirem sempre, aos colegas de laboratório e aos colegas que colaboraram no projeto, Muhammad Waqas e Antonio Nehme.

Palavras Chave

Privacidade, Dispositivos inteligentes, Internet das coisas, Sensores, Transparência, Controlo

Resumo

O uso crescente de dispositivos inteligentes para monitorização de espaços tem provocado um aumento das preocupações sobre a privacidade dos utilizadores destes espaços. Face a este problema, a legislação sobre o direito à privacidade tem sido trabalhada de forma a garantir que as leis existentes sobre este tema são suficientemente abrangentes para preservar a privacidade dos utilizadores. Desta forma, a investigação neste tópico evolui no sentido de criar sistemas que garantam o cumprimento destas leis, ou seja aumentam a transparência no tratamentos dos dados dos utilizadores. No contexto desta dissertação, é apresentada uma estratégia baseado num demonstrador para fornecer um controlo ao utilizador sobre os seus dados armazenados durante a utilização temporária de um ambiente inteligente. Para além disso, esta estratégia inclui garantias de transparência, evidencia o direito ao esquecimento, fornece a capacidade de consentimento e prova desse consentimento. É também mencionada neste documento uma estratégia para um controlo de privacidade neste tipo de ambientes. Esta dissertação foi desenvolvida no âmbito do projeto CASSIOPEIA onde o caso de estudo se foca no *SmartBnB problem* onde um utilizador arrenda uma casa inteligente durante um tempo limitado. Este documento apresenta o sistema desenvolvido que garante a privacidade e controlo do utilizador sobre os seus próprios dados.

Keywords

Privacy, smart devices, Internet of Things, Sensors, Transparency, Control

Abstract

The increasing use of smart devices for monitoring spaces has caused an increase in concerns about the privacy of users of these spaces. Given this problem, the legislation on the right to privacy has been worked to ensure that the existing laws on this subject are sufficiently comprehensive to preserve the privacy of users. In this way, research on this topic evolves in the sense of creating systems that ensure compliance with these laws, that is, increase transparency in the treatment of user data. In the context of this dissertation, a demonstrator-based strategy is presented to provide users control over their stored data during the temporary use of an intelligent environment. In addition, this strategy includes transparency guarantees, highlights the right to forgetting, provides the ability to consent and proof of that consent. A strategy for privacy control in such environments is also mentioned in this paper. This dissertation was developed within the CASSIOPEIA project where the case study focuses on the SmartBnB problem where a user rents a smart home for a limited time. This paper presents the developed system that ensures the user's privacy and control over their data.

Contents

Contents	i
List of Figures	v
List of Tables	ix
Glossário	xi
1 Introduction	1
1.1 Motivation	2
1.2 Hypothesis and Objectives	3
1.3 Contributions	5
1.4 Document Structure	5
2 Background	7
2.1 Introduction	8
2.2 Current Privacy Landscape	8
2.2.1 Privacy by default and Privacy by design	9
2.2.2 Privacy and Data Protection	10
2.2.3 Privacy and Security	11
2.2.4 Privacy boundaries	12
2.3 User-centric Design and Solutions	14
2.3.1 Consent	15
2.3.2 Access Control	17
2.3.3 Delegation Problem	18
2.4 Internet of Things Landscape	19
2.4.1 The impact of IoT in privacy	21
2.4.2 Smart Homes	23
2.4.3 Privacy on a smart home	24
2.4.4 Solutions for smart homes	26

2.5	Related technologies	28
2.6	Conclusion	30
3	The CASSIOPEIA project	31
3.1	Introduction	32
3.2	Description of CASSIOPEIA project	32
3.3	Problem Scope	34
3.4	Proposed Solution	35
3.5	Conclusion	36
4	Architecture and Specifications	37
4.1	Introduction	38
4.2	Use Cases and Scenarios	38
4.3	Requirements	43
4.4	System Architecture	47
4.4.1	Data Manager	49
4.4.2	Receipt Manager	50
4.4.3	Privacy Manager	51
4.4.4	Signature Application	52
4.5	Demonstrator workflow	54
4.6	Persistence	59
4.7	Permissions	59
4.8	Conclusion	60
5	Implementation	61
5.1	Introduction	62
5.2	Architecture	62
5.3	Implementation details	67
5.3.1	Privacy Manager	68
5.3.2	Data Manager	68
5.3.3	Receipt Manager	69
5.3.4	Signature Application	70
5.4	Home Assistant integration	71
5.5	Sequence Diagrams	74
5.6	Conclusion	79
6	Results and Evaluation	81
6.1	Introduction	82
6.2	Privacy principles response	82

6.3	Evaluation of the cryptographic overhead on wireless sensor networks	83
6.3.1	Delay analysis	84
6.3.2	Size analysis	86
6.3.3	Power consumption analysis	87
6.4	Home Assistant integration results	87
6.5	Prototype results	89
6.6	Conclusion	100
7	Conclusion	101
7.1	Introduction	102
7.2	Conclusions	102
7.3	Future work	103
7.3.1	Proposed solution to delegation problem	104
A	CASSIOPEIA project	107
A.1	CASSIOPEIA use cases and scenarios	108
A.2	CASSIOPEIA architecture	111
B	Evaluation of the cryptographic overhead on wireless sensor networks	117
B.0.1	Hardware	119
B.0.2	HTTP	119
B.0.3	MQTT	120
C	Experiment Results - Evaluation of the cryptographic overhead on wireless sensor networks	123
	References	127

List of Figures

2.1	Relation between Data Protection, Privacy and Security	11
2.2	Solove's Privacy Taxonomy	13
2.3	Data lifecycle	15
2.4	Consent lifecycle	17
2.5	Internet of Things (IoT): Generic idea	19
2.6	Example of a consent request	22
2.7	Example of a periodic notification	22
2.8	Generic smart Home working	24
3.1	Interaction between different users	34
4.1	Feature requirements process	44
4.2	General architecture, the databases are placeholder for their implementation.	47
4.3	Interactions of the renter before or during the stay	49
4.4	Signature process	52
4.5	Initial phase in the demonstrator usage	56
4.6	Receipt generation, signature and storage.	57
4.7	<i>Data Manager Service</i> interactions.	57
4.8	Data deletion request at the end of the stay.	58
5.1	Architecture of the system	62
5.2	Relation database model of <i>Data Manager Service</i>	63
5.3	Relation database model of <i>Privacy Manager Service</i>	63
5.4	Distributed Hash Table (DHT) work	65
5.5	DHT Overlay Network	65
5.6	DHT model	65
5.7	Access through Token Authentication	68
5.8	Delays measured on the experiments.	73
5.9	Personal data anonymization	76
5.10	Remove personal data from the system	76

5.11	List of the accepted devices	77
5.12	Get all information about all stays	77
5.13	Register a new renter/stay in <i>Privacy Manager Service</i> and <i>Data Manager Service</i>	77
5.14	Export data to a Comma-Separated Value (CSV) file	78
5.15	Information about a specific stay	78
5.16	Manager consent provided and the entities involved in this consent	78
5.17	Get the entities involved in the personal data process	79
6.1	Principles relating to processing of personal data	82
6.2	Bases for processing personal data	83
6.3	Elliptic-curve DiffieHellman (ECDH) delay analysis	85
6.4	Cipher delay analysis	85
6.5	Central Process Unit (CPU) processing time for a single message.	87
6.6	Home Assistant Home View using the 2D and 3D floorplans	88
6.7	Owner view - Cannot see any device	89
6.8	Home Assistant views for the Renter (Restricted) and for the Owner (Home)	89
6.9	Add a new policy	91
6.10	Add a new device	92
6.11	Add a new entity	92
6.12	Register new user	92
6.13	List the available entities	93
6.14	List the users in the system	93
6.15	List the availble policies	93
6.16	List the available devices	93
6.17	Create new stay	94
6.18	Present new receipt to sign	94
6.19	Receipt signature process	95
6.20	Validation process of the receipt signature	95
6.21	Check the stay in the <i>Data Manager Service</i>	95
6.22	Wireshark capture about new register	96
6.23	Data Explorer of the InfluxDB for the temperature sensor	97
6.24	Data Manager access using the receipt	97
6.25	Data state updated to “Requested”	98
6.26	Notification about data deletion request	98
6.27	Data state updated to “Removed”	98
6.28	Wireshark capture about data deletion	99
A.1	Overview of the components in the Demonstrator	111

A.2	Interactions of the <i>Privacy Manager Service</i> and the remaining services	112
B.1	Delays measured on the experiments.	118
B.2	WSN architecture for HTTP.	120
B.3	HTTP messages exchange	120
B.4	WSN architecture for HTTP.	121
B.5	MQTT messages exchange	121

List of Tables

2.1	Digital assistants alternatives	27
4.1	Use Case 1 - Create new stay	39
4.2	Use Case 2 - Request the removal of renter's personal data	39
4.3	Use Case 3 - Notification about data removed	40
4.4	Use Case 4 - Data anonymization	41
4.5	Use Case 5 - Renter verify devices	41
4.6	Use Case 6 - List all information regarded by all the stays	42
4.7	Use Case 7 - Export personal data	42
4.8	Use Case 8 - Access to unified control system to manage personal data	42
4.9	Use Case 9 - Privacy Policies of different entities that process data	43
4.10	Use Case 10 - Control who can see/use renter's personal data	43
4.11	Functional Requirements	45
4.12	Non-functional Requirements	46
4.13	Receipt structure and possible option to fill the fields	51
5.1	Devices used in the project	74
6.1	Ratios for the experiment	87
A.1	Use Case 1 - Prior to arrival	108
A.2	Use Case 2 - Arrival and Setup	108
A.3	Use Case 3 - Lock, Doorbell, Lights	109
A.4	Use Case 4 - Cameras and Security	109
A.5	Use Case 5 - Departure and Post-rental	110
A.6	Receipt structure and possible option to fill the fields	115
B.1	Devices and characteristics	119
C.1	Results from the experiment HTTP.	124
C.2	Results from the experiment MQTT.	125

Glossário

ABE	Attribute Based Encryption	MNO	Mobile Network Operator
ACC	Access Control Contracts	MQTT	Queuing Telemetry Transport
ACID	Atomicity, Consistency, Isolation, Durability	NFC	Near Field Communication
AES	Advanced Encryption Standard	NGI	Next Generation Internet
API	Application Programming Interface	NoSQL	Not Only Structured Query Language
CASSIOPEIA	Contextually-Appropriate Selective Sharing IoT Open-standard PErmissioning Architectures	OTA	Over-the-Air
CC	Citizen Card	P2P	Peer-to-Peer
CPU	Central Process Unit	PII	Personally Identifiable Information
CSRF	Cross Site Request Forgery	PKI	Public Key Infrastructure
CSV	Comma-Separated Value	RDMS	Relational Database Management System
DHT	Distributed Hash Table	REST	Representational state transfer
ECDH	Elliptic-curve DiffieHellman	REST	Representational State Transfer
eSIM	embedded Subscriber Identification Module	RFID	Radio frequency Identification
GDPR	General Data Protection Regulation	RPi	Raspberry Pi
HTTP	Hypertext Transfer Protocol	SHIB	Smart Home based the IoT-Blockchain
IERC	European Research Cluster on the Internet of Things	SIM	Subscriber Identification Module
IoT	Internet of Things	UMA	User-Managed Access
JSON	JavaScript Object Notation	URL	Uniform Resource Locator
M2M	Machine-to-Machine Communication	US	United States
		UUID	Universally Unique Identifier
		V2V	Vehicle-to-Vehicle Communication
		WSN	Wireless Sensor Network

CHAPTER 1

Introduction

1.1 MOTIVATION

The smart environment's goal is to improve the quality of human life in terms of comfort and efficiency. The IoT paradigm is one of the enabler technologies that allow us to create and develop smart environments. Security and privacy are considered key issues in any real-world smart environment based on the IoT paradigm [1].

The IoT is considered the next revolutionizing wave of our society. Smart homes, factories, and cities are being equipped with a huge number of IoT sensing devices [2]. It is an expanded network based on the Internet, and its goal is to achieve real-time interaction among things, machines, and humans [3]. There are a huge number of IoT applications [4]. For example, the authors of [5] describe a complete smart system to control air pollution, creating a smart environment that uses comprehensive intelligent services aimed at monitoring and managing air pollution. Air pollution levels can be measured using remote sensors. Additionally, IoT technology can be integrated to remotely detect pollution without any human interaction.

Nowadays, there are several smart environments where we interact. This leads to privacy concerns and the exposure of our data in online services. The main problem is the data collected without the user's consent and how it can affect his privacy and his interests. The increasing usage of smart devices in everyday environments, without proper regulation for them, is a potential hotspot for privacy breaching and data theft. One such example is surveillance cameras. Cameras, ideally, should not capture images from other people without their consent because it can be a privacy invasion [6].

According to the legislation, the right to privacy is considered a fundamental human right as explained in chapter 2. In a smart environment, the user should be informed about the environmental conditions (which kind of devices are being used), and especially the user should give consent for the data collection. Moreover, the user should be informed about what happens with his data and in what way. Furthermore, the user should be informed and consent to all data flow from the beginning when the consent is given until the end of the data life. When this data is not needed, then it should be removed. The data should be collected with a specific purpose, and that purpose should be explained when the user concedes his consent. When this purpose is fulfilled, the data should be removed. In addition, the user should have the power over his data, this means that the user can request for his data to be deleted.

Assuming that smart devices and surveillance cameras exist in our lives, how can we provide privacy in these environments? The data subject should be informed about his data and he should provide consent for data collection. Knowledge about data flow and the entities involved in the processing of personal data should be transparent, understandable, and provide easy access. It is important to guarantee that all the process is transparent, trusted, and cryptographic secure. In addition, it is important to consider that the data subject uses smart devices controlled by another entity (owner of the smart environment). In this way, the temporary usage of a smart environment increases privacy problems.

A common smart environment is the smart home. Typically, smart homes have a large

number of smart devices collecting data. These smart devices could be sensors, actuators or surveillance cameras. Most of the times, these devices are connected to an unified control platform to control all devices in a central point of control. Currently, most homes are not considered smart since the owner needs to use several applications to control the devices in the space. Considering the unified control platform as the platform to control all devices in the smart environment, it is required to have a way to control this platform to ensure the user's privacy.

SmartBnb Problem occurs when a host rents an apartment with IoT devices to a guest. This operation involves the delegation of device functions to the guest and the management of the guest's collected personal data. The host should not have access to the guest's collected personal data by the devices. Furthermore, other users on the house should not have access to the personal data from other users. Smart homes include a range of devices for sensing and automation, such as thermostats, video cameras, door locks, smart TVs, among others. All these devices can collect personal data of different degrees of sensitivity and some allow control of its features.

The dissertation was conducted under the scope of the Contextually-Appropriate Selective Sharing IoT Open-standard PErmissioning Architectures (CASSIOPEIA) project, which addresses the **SmartBnb Problem**, where the **Selective Sharing** is a critical dimension of privacy. CASSIOPEIA was funded by Next Generation Internet (NGI) Trust, coordinated by Instituto de Telecomunicações (Aveiro, Portugal) and with Birmingham City University (United Kingdom) and Doctor Gilad Rosner, as partners. CASSIOPEIA project focuses on the SmartBnb problem and it focuses on the user's privacy management, controlling the unified control platform to provide privacy to the data subject. The main motivation for the CASSIOPEIA project was the missing of a comprehensive solution for the SmartBnb problem. The project takes a user-centric approach, which means that the user has the right to control their own data. The user-centric view of IoT and smart homes means strong privacy, not just confidentiality but also flexible sharing arrangements that align with social norms for information sharing and user-centric design principles.

Simply, the main motivation for this dissertation is to provide more control and transparency for the data subject in a temporary smart environment usage. The idea is to create a demonstrator that enables the user to have control over his data. As a demonstrator, the study case is about the SmartBnb problem but the concept should apply to any smart environment. Thus, the same data subject can use different smart environments with different specifications and requirements.

1.2 HYPOTHESIS AND OBJECTIVES

The idea of the system is to provide a prototype that simulates a system to provide control and transparency for the data subject. The data subject should be informed about his data flow and should be able to control his data. To control the permissions of smart home devices, it is possible to use two different approaches:

- Each device has its screen, app, or a set of controls. Each manufacturer creates its own set of interactions, its conception of identity relationships (owner, user, admin, guest), and its privacy characteristics for sharing data or device functions;
- Devices are controlled by an internet giant such as Amazon or Google. These companies will dictate the shape of privacy and sharing arrangements since they control the platform for in-home device services

Currently, smart home device privacy and sharing arrangements are inconsistent, piecemeal, inconvenient, and/or controlled by large internet companies who have their own visions of how people should and should not be able to share their data.

IoT devices collect data without displaying how data is treated in the background, neither if this data will be deleted. The relationship between owner and guest is absent from today's smart home architecture.

A system that manages user's data should have some important properties:

- The first consideration is that the system should be hard to deceive and should try to ensure that people are informed correctly about their data. In terms of usability, it is required that the user accesses the correct data. In terms of privacy, it is mandatory that the user only accesses his data. The security of the system is not the focus of this dissertation but the minimal requirements of security should be ensured.
- Another consideration is that the system should be prepared to handle a big number of users and be able to scale fast. It is expected that the same user has more than one smart environment and the number of users increases over time. The system should be prepared to scale adding more capacity to the system. The system should be prepared to handle changes on the requirements and be easy to add new functionalities or new components to increase the user's privacy. It is possible to have more requirements on the system usage. Thus, the system should be prepared to add more features or components.
- Moreover, the system should be designed to be easy and safe to use and the integrity of the proof of the consent should be always ensured. The consent should be provided at the beginning of the usage of the system.
- However, during the smart environment usage more components can be added. If these new components process the user's data, then the user should be informed transparently and the consent should be requested again. The objective is to have an informed user and provide him with the power to decide which entities are allowed to process his data.

This dissertation includes a research work about how open-standard or open-source technologies can be used to create usable and transparent architectures enabling device owners to selectively collect, share, and retain data from users. The idea is to apply existing technologies to enable a broad set of interactions and an expansive conception of privacy. In addition, the dissertation includes the development of new components to increase the user's privacy and data control.

1.3 CONTRIBUTIONS

During this work, it was developed a testbed with the following services: Privacy Management that manages the stays within a specific smart home; Data Management that allows the user to control his personal data; and Receipt Management that generates and securely stores the receipts associated with the user's data. Part of this work was integrated within the CASSIOPEIA project. CASSIOPEIA was an international project developed between the University of Aveiro and Birmingham City University.

Alongside this work, three publications [7]–[9] were accepted and presented at peer reviewed international conferences. The Permission and Privacy Challenges in Alternate-Tenant Smart Spaces [7] is a joint publication that was published during the CASSIOPEIA project and describes the implemented demonstrator of how smart environments can operate in a privacy-respecting manner. It was presented at the Open Identity Summit 2021 ¹ conference. In the paper named *esim* suitability for 5G and B5G enabled IoT verticals [8], we explore the usability of *esim* on vertical using 5G that can benefit from adopting *esim*. The paper presents an overview of *esim*, discuss its main features, compare it to the physical Subscriber Identification Module (SIM) card, and specify the main characteristics of each vertical market, as the use of hardware tokens is relevant as secure authentication sources. In the paper named Hands-on evaluation of the cryptographic overhead on wireless sensor networks [9], we tested different security configurations using the two most used transport protocols (Hypertext Transfer Protocol (HTTP) and Queuing Telemetry Transport (MQTT)). We measured their effects on five commonly used embedded devices in IoT Wireless Sensor Network (WSN)s: ESP8622, ESP32, RPi1, RPi2, and RPi3. We considered three different metrics for evaluating each configuration: power consumption, message delay, and additional message length (bytes). Both were presented in the FiCloud 2021 ² conference, and I was the first author.

1.4 DOCUMENT STRUCTURE

The document is organized as follows: chapter 2 presents the most relevant background; chapter 3 describes the most important details of the CASSIOPEIA project that was the basis for this work project, the problem scope and the proposed solution; chapter 4 describes the architecture and the specifications including the requirements and the use cases; the details about the implementation are described in chapter 5 and the results are presented in chapter 6; and finally, chapter 7 presents the main conclusions and the future work.

Appendix A presents the details concerning the CASSIOPEIA project; Appendix B presents the details concerning an evaluation of the cryptographic overhead on WSN; and Appendix C presents the experimental results of this evaluation.

¹<https://oid2021.compute.dtu.dk/>

²<http://www.ficloud.org/2021/>

CHAPTER 2

Background

2.1 INTRODUCTION

The work developed and presented in this dissertation is based on the search for a strategy to enable selective sharing permission in IoT environments. Thus, the first step was to survey the state of the art on privacy, IoT, and how IoT environments affect or may affect the privacy of users of these environments.

The first requirement was to understand the concept of privacy and the main concepts associated with it. In addition, it is important to understand how users' privacy should be guaranteed through current legislation. We did not focus on the General Data Protection Regulation (GDPR) although many definitions established in this regulation were used. However, the focus was on establishing a set of general parameters that must be guaranteed and respected when a smart environment is used by an ordinary user.

There is a large number of privacy concerns in different scenarios. In this document, we focus on the smart environments that contain smart devices collecting data permanently. Thus, we present a definition of IoT and the most relevant technologies related to it. The IoT environments include a large number of different scenarios. Additionally, our focus is on smart homes and how the temporary usage of this smart environment can affect the user's privacy.

This chapter presents the background mentioned above and it is organized as follows: section 2.2 defines the current privacy landscape. The users interests and protection are described in section 2.3. Finally, section 2.4 presents the IoT definitions and the most relevant technologies.

2.2 CURRENT PRIVACY LANDSCAPE

According to the Oxford Dictionary of English, one definition of privacy is “the state of being alone and not watched or interrupted by other people” and “the state of being free from the attention of the public”. This is difficult to execute when considering that a purpose of new technologies is to connect everything to the Internet such as people, devices and vehicles. Users of the Internet can share private information through computer, smartphones or other devices giving the consent for that or involuntarily. However, this can happen offline. When the user is exposed to smart devices, collecting permanently data the user is not actively connected to the Internet, but his privacy is at risk. Consequently, we need to consider the online and offline privacy problems in light of the large volume of devices.

The right to privacy is considered a fundamental human right in various jurisdictions. The authors of [10] found value in a classic definition of privacy by Alan Westin: privacy is “the claim of individuals, groups, or institutions to determine for themselves when, how, and to what extent information about them is communicated to others”. Also, they considered useful Westin's view that privacy protects four states: solitude, intimacy, anonymity and reserve. The increasingly omnipresent sensors and the huge number of IoT devices in our lives decrease privacy.

In recent years, there has been a huge growth in the complexity and volume of global data flows and data processing and, consequently, there are huge amounts of information available

to a wide range of parties. This creates a lack of transparency, since the user does not know how his personal data is processed, nor does he have control over it. The main consequence is the lack of privacy [11]. Privacy is more about than the ability to store, process and analyse information. It matters to know what kind of information is collected and in what manner, who needs to have access to personal information, and for what purpose. The durability of these data needs to be available, its format and degrees of accuracy and precision are also important topics to take into account when we want to guarantee privacy [12].

Privacy is defined by social psychologist Irving Altman as “an interpersonal boundary control process” and “selective control of access to the self or ones group”. Moreover, Altman considers that privacy mechanisms serve to define the limits and boundaries of the self; people sometimes make themselves accessible to others and sometimes close themselves off from others. Each user should be able to decide and be informed about what happens to his personal data. Furthermore, the user should be able to selectively share and control his data. This intrinsic boundary is essential to give the user the control and guarantee his privacy because he can share his data but as an option and not involuntarily.

2.2.1 Privacy by default and Privacy by design

Privacy by design should be a critical requirement for services and products provided to third parties and individual customers such as social networks and search engines. Many users have limited knowledge about the technical and informatic specifications and hence are not in a position to take relevant security measures to protect their data. Privacy by design is a proactive approach rather than a reactive measure. It prevents privacy-invasive events before they happen [13]. This principle should be applied to the entire lifecycle of data.

Basic protection is always necessary and typically it is recognized as privacy by default. Moreover, providers have to enable users to better protect their data by providing appropriate privacy tools such as access control, encryption, and anonymization. Privacy by design includes the idea that systems should be designed in such a way to avoid or minimize the amount of personal data processed [14].

Privacy by default principle includes the following statements [13]:

- **Purpose Specification:** the purposes for which personal information is collected, used, retained, and disclosed shall be communicated to the individual (data subject) at or before the time the information is collected. Specified purposes should be clear, limited, and relevant to the circumstances.
- **Collection Limitation:** the collection of personal information must be fair, lawful, and limited to what is necessary for the specified purposes.
- **Data Minimization:** the collection of personally identifiable information should be kept to a strict minimum. The design of programs, information and communication technologies, and systems should begin with non-identifiable interactions and transactions, as a default. Wherever possible, identifiability, observability, and linkability of personal information should be minimized.

- **Use, Retention, and Disclosure Limitation:** the use, retention, and disclosure of personal information shall be limited to the relevant purposes identified to the individual, for which he or she has consented, except where otherwise required by law. Personal information shall be retained only as long as necessary to fulfill the stated purposes, and then securely destroyed.
- Default settings should be the most privacy-protective.

The general requirements to guarantee privacy are based on some aspects. There should be limits to the collection of personal data and any data should be obtained lawfully and appropriately, with knowledge or consent of the data subject. Personal data should be relevant for the purpose and this purpose should be specified, limited, and respected. Personal data should not be disclosed, made available, or used for other purposes. Personal data should be protected against risks such as loss or unauthorized access, destruction, use, modification, or disclosure of data. There should be a general policy of openness about developments, practices and policies concerning personal data.

2.2.2 Privacy and Data Protection

A possible definition to personal data is “any information relating to an identified or identifiable natural person (data subject); an identifiable natural person can be identified, directly or indirectly, in particular by reference to an identifier such as a name, an identification number, location data, an online identifier or to one or more factors specific to the physical, physiological, genetic, mental, economic, cultural or social identity of that natural person” and it is present on GDPR. This regulation was adopted by the EU as a successor of the privacy Directive 95/46/EC [15]. Laws and regulations define a set of practices that are acceptable and unacceptable.

It is necessary to protect the data resulting from the huge amount of data collected. All the data collected is defined as personal or non personal regardless of context and regulation is applied [16]. The GDPR requires that personal data shall be processed lawfully, fairly, and transparently, and it is collected for specific, explicit, and legitimate purposes. The data shall be adequate, relevant, and limited to what is necessary. The data shall be kept in a form that permits identification of data subjects for no longer than necessary for the purpose. And, the process of personal data shall be done in a manner that ensures appropriate security of the personal data (protection against unauthorized or unlawful processing and loss, destruction, or damage) using appropriate measures (integrity and confidentiality).

The authors of [17] believe that “privacy and data protection are products of distinct practices and regimes of enunciation, such as politics, law, ethics, economy, religion and so on, and that the challenge is not so much to find the foundational unity “behind” these, than it is to understand how, each being singular, they interact and articulate.” Additionally, the authors explain the right to data protection as a “set of fair information practices or as the regulation and organization of the conditions under which personal data can be lawfully processed”

Privacy and personal data protection are two interrelated terms that are often used interchangeably, but they constitute two discrete and different notions [18]. These concepts are closely related, but they are not synonymous.

In practice, we can define data protection as a set of policies to protect the data, and permanently we need to protect this data. We need to ensure the validity of the data. When this data is considered sensitive or personal it is mandatory to guarantee its privacy. Privacy is about the laws to apply to guarantee the preservation of personal privacy and data protection.

The Figure 2.1 depicts the relation between data protection, privacy, and security.

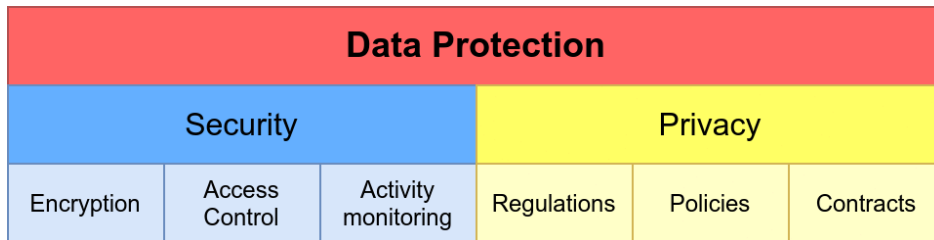


Figure 2.1: Relation between Data Protection, Privacy and Security

We can assume that data protection includes the privacy legislations and norms to ensure correct usage of the system and security is about how we can apply this in terms of implementation. subsection 2.2.3 describes more about the connection between privacy and security.

2.2.3 Privacy and Security

Protecting personal privacy in information systems is becoming increasingly critical with the extensive use of networked systems and the Internet. These technologies provide opportunities to collect large amounts of personal information about online users, potentially violating those users' privacy [19].

One of the first stages of a project is the specification of the requirements including the privacy policies. The mechanisms to implement these policies are the security techniques such as encryption and access control. Encryption includes choosing the more adequate algorithms and parameters such as key size. The privacy policies include what data is necessary to protect and depending on the context it can mean what data is necessary to cipher or apply access control.

Privacy and security are not the same thing and implementing security mechanisms is not enough to guarantee privacy. Security considers the confidentiality, integrity, and availability of information. It is often necessary to have privacy facilitating the control of information flows and helps to ensure the correctness of data. It is possible to have high levels of security but no privacy or some sort of privacy without security. Ensuring the confidentiality and availability of information does not represent anything about how and when this data will be used or processed. Offering strong security does not solve privacy problems. Security fails to address questions such as scope, purpose and use, adequacy, lifetime, or access [12].

Illegitimate use of data is unauthorized such as when it is stolen, altered, or viewed by the wrong party. This is the security domain that protects data from being inappropriately accessed, modified, or shared. Legitimate uses of data are those that have been authorized. However, there are a lot of legitimate data uses that may be problematic or harmful. Just because something is legal does not mean it is positive. While illegitimate uses of data must be avoided with security. Legitimate but harmful uses of data must be interrogated through the lens of privacy protection. For example, in countries where companies can collect individual data with only minimal notification and personal data can be used in ways that people did not expect or did not consent to [20].

2.2.4 Privacy boundaries

Advances in technology require that we rethink privacy due to the increasing abilities to control, detect, record, find and manipulate several devices, and the way how these devices interact in our lives.

Privacy legislation has differences between the United States (US) and Europe. One of the major differences is the power and breadth of the right to privacy. Europe's legislation is more careful than the US that is more about legal intervention before and after a privacy violation occurs.

The people's expectation about their information being collected or used is another essential dimension to consider. In the US the reasonable expectation of privacy is used when privacy is violated but it has a problem as the expectation is continuously changing. In Europe, the Charter of Fundamental Rights of the EU has the following position: "Everyone has the right to respect for his or her private and family life, home and communications."

Typically, our home is the place where we feel more comfortable and where more private things are. With the introduction of virtual assistants, networked and interactive devices, increasing risks of privacy arise [21]. Ways to control privacy lack in such a system and this will increase with future ubiquitous computing systems [22].

We can establish boundaries of data collected according to the context of the information. These boundaries must be physical, such as walls, or data-type boundaries such as personally identifiable versus non-identifiable ones, or regulatory boundaries such as telecommunications. Sharing information outside these boundaries may be experienced as a privacy violation.

Gary T. Marx has done extensive research in the areas of privacy differentiating between four borders that are perceived as a privacy violation. *Natural borders* are considered the physical limitations of observations. *Social borders* are the expectations about confidentiality for members of certain social roles such as family members. This also includes expectations that these members will not access information addressed to the user. *Spatial and temporal borders* are the usual expectations of people that parts of their life, both in time and social space, can remain separated from each other. Finally, we have *borders due to ephemeral or transitory effects*. These describe a fleeting moment, an unreflected action that one hopes to be forgotten soon [12].

Solove [12] created a privacy taxonomy which means an overview of the activities that

lead to privacy problems. He groups such activities into four sets as we can see in Figure 2.2. Several entities can collect data and most of the time this occurs voluntarily by the data subject, but hidden or forced collections can happen and they violate privacy. Data holders process the data and it can affect the data subject’s privacy. Aggregation, identification, secondary use activities, and exclusion are possible ways to reduce the data subject privacy. Information dissemination activities then propagate the information outward from the data holder.

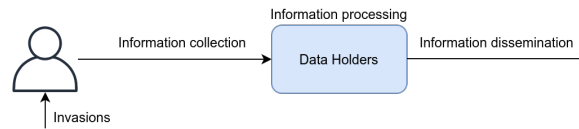


Figure 2.2: Solove’s Privacy Taxonomy

Whenever the user’s personal information crosses these boundaries, without his knowledge, privacy is affected. Privacy violations can be seen as involuntary border crossings.

In IoT environment, privacy violations can occur when data leaks from one context to another, such as from home to the workplace. Related to this we have Altmans theories of boundary management, which claim that privacy is part of peoples ability and need to negotiate the boundaries between themselves and others, and with society at large. The presence of IoT can change our mental boundaries in our homes and private lives. We can have boundaries limited by the walls of our house but the IoT devices and consequently access to the Internet traverses these boundaries.

Territorial privacy [22] is a concept that moves away from the information-centric view in traditional systems to a context-centric approach. In the information-centered approach, privacy is controlled by protecting particular information. With territorial privacy, the privacy decisions of users are often based on their physical or spatial context. This technique offers a more usable and intuitive view of controlling individual privacy. In a shared personal environment, such as home, comprehensive privacy systems can be realized to adapt the behavior of system components to the users privacy preferences.

The boundaries present on territorial privacy must be important to separate the physical and the virtual boundary on a smart environment, such as a smart home [23]. Invisibly embedded sensors, actuators, and in particular wireless communications could widen the boundaries of a private territory far beyond its physical boundaries. As a consequence, the ability to perceive and control who is observing or disturbing users in their private territory will decrease or even cease to exist. Thus, perceiving and controlling those new virtual territorial boundaries with the ease of traditional privacy controls outline the main goals of the territorial privacy concept.

According to the literature [22], [23] there are future challenges to territorial privacy. Interesting challenges are related to dynamic adaptation to changing privacy requirements and taking multiple individuals. In shared and public environments an emphasis needs to be placed on informing the user about potential privacy implications, while services under the

users full control should still respect privacy preferences. Intrusions into private territories and intrusion chains need to be studied and modeled. The demarcation of private territories requires context information and mechanisms to determine observers. Expressive policies are required to specify user's preferences that can also be dynamically adapted to changing situations and requirements.

2.3 USER-CENTRIC DESIGN AND SOLUTIONS

The user's interests and protection is one of the priorities when a system is developed. This action includes strong privacy defaults, appropriate communication, and user-friendly options which mean user-centric approaches.

GDPR introduces obligations on service providers to grant users with rights to data erasure, to object to processing, to the portability of data on request, and to protect profiling. Additionally, service providers need to ask for explicit consent to collect data for specific purposes. These protection measures enhanced the control of end-users over their data [24].

Online identities are associated with individuals and improper handling of these identities may therefore affect them. Placing individuals at the center of identity management and empowering them with tools to actively manage their identity may help limit the privacy risks provoked by the information society [25].

As said above, the huge amount of personal data is increasing with the growth of the usage of smart devices. The identity of personal data owners can be a problem due to the relation between the data and a specific person. Personally Identifiable Information (PII) is a vital role in providing user-centric services. There are vast companies that collect, store and process the PII of their customers using different applications [26]. The GDPR stipulates that websites must obtain consent from the users to collect personal information such as IP addresses and cookies. Collecting anonymous data and deleting identifiers from the database limit the ability to derive value to the user such as personalized content.

There are several proposals to address this problem focusing on anonymizing data before sending them to a third party such as a server. Data anonymization is the process of protecting private or sensitive information by erasing or encrypting identifiers that connect an individual to stored data. However, attackers can use deanonymization methods to retrieve the data anonymization process. The authors of [27] consider that this approach gives the user no control over the information shared. Thus, they propose an approach based on anonymizing data in the user's mobile devices, before they are sent to a third party.

The authors of [28] consider that the resource constraint of most IoT devices is a limitation to use traditional solutions based on private keys stored in the device's memory and computationally heavy cryptographic algorithms will turn insecure, inefficient, or, directly, impossible to run. They propose a new mechanism to protect, authenticate and anonymize data in IoT systems supported by future 5G networks.

During the usage of a system, the user's data have a lifecycle and the user should be the center of the process controlling. At the beginning of the system usage, the user must give consent for the system to collect personal data. Then, the user should be able to control his

data and at any time, the user should be able to see his data flow including the process. All the processes must be transparent.

Figure 2.3 shows the general data lifecycle.

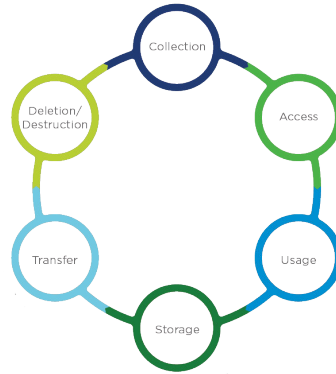


Figure 2.3: Data lifecycle

The system must be implemented considering privacy by default and privacy by design principles. Access control policies must be implemented to restrict data access. In this access control, the system needs to ensure that the control of personal data is done only by the data subject. The right of access ¹ includes information about the processing purpose, categories of personal data, the planned duration of storage, information about the rights of the data subject such as rectification, erasure, or restriction of processing, and any existence of an automated decision-making process including profiling. If personal data are transmitted to a third country without an adequate level of protection, data subjects must be informed of all appropriate safeguards which have been taken.

At the end of the system usage, the consent must be revoked to enhance the end of the permissions to collect data. The right to be forgotten ² is the right to erasure. Personal data must be erased immediately when the data are no longer needed for their original processing purpose or the data subject has withdrawn his consent and there is no other legal ground for the processing.

2.3.1 Consent

Processing personal data ³ is prohibited unless it is expressly allowed by law or the data subject has consented it. Consent must be freely given, specific, informed, and unambiguous. Freely given means that it must be given voluntarily and it is necessary to guarantee intentional sharing. This implies a real choice by the data subject.

For consent to be informed and specific, the data subject must be notified about the controller's identity, what kind of data will be processed, how it will be used, and the purpose of the processing operations. When relevant, the controller has to inform about the use of the data for automated decision-making and the possible risks of data transfers.

¹<https://gdpr-info.eu/issues/right-of-access/>

²<https://gdpr-info.eu/issues/right-to-be-forgotten/>

³<https://gdpr-info.eu/issues/consent/>

The consent can be applied to one or more purposes but all the purposes need to be explained. Finally, the consent must be unambiguous which means it requires either a statement or a clear affirmative act.

The authors of [29] propose a user-centric solution named ADvoCATE, that allows data subjects to easily control consents regarding access to their personal data in the IoT ecosystem and exercise their rights defined by GDPR.

Consent is the basis of any private practice. Consent is important for both organizations and users. Not only cannot individuals prove what they have accepted at any point in time, but also organizations are struggling with proving such consent was obtained leading to inefficiencies and non-compliance. The authors of [30] defend a different approach to how the web of personal information operates using personal data receipts which can protect both individuals and organizations. These receipts are applied to online actions, from registration to real-time usage, is preceded by valid consent, and is auditable and demonstrable at any moment by using secure protocol and locally stored.

It is useful to define a set of idealized consent requirements [31]:

- **Choice:** Maximize opportunities for individual authorization, mutual agreement, personal data sharing; minimize acquiescence to sharing and unconsented sharing.
- **Relevance:** Capture consent at a time and in a manner most relevant to and convenient for the individual
- **Granularity:** : Enable differentiation of the parameters of consent, including data sources, data items, receiving parties, and modification of consent parameters over time, including revocation, again in a manner most relevant to and convenient for the individual.
- **Scalability:** Enable consent interactions, processes, and systems to scale to accommodate the numbers of data sources, data items, and consent functions that individuals will realistically experience.
- **Automation:** Enable machine processing and recording of consent functions.
- **Reciprocity:** Capture the consent of the data-receiving party in dealing with the individual, along with capturing the consent of the individual to sharing data.

The receipt can be proof of the consent. Considering a simple case where the system does not have the complex case sharing personal data with third parties which is more complex due to revocation and re-consenting. The lifecycle of consent using a receipt is shown in Figure 2.4.

At the initial stage, the user is exposed to the privacy notice. The notice must be as stated above it means clear, specific, and unambiguous. Usability is a clear challenge in this stage because the interface must be user-friendly. Once consent is given, there are three possible actions. The first one is the user withdrawing consent, either wholly or in part. The second one is the controller making changes to the notice which requires a re-accept of the new conditions. The third one is when a disagreement between the user and the controller on how personal data was collected or used occurs.

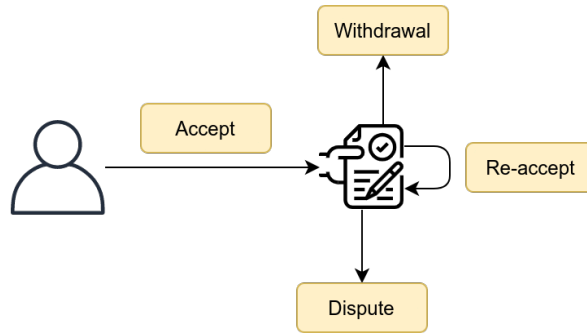


Figure 2.4: Consent lifecycle

A personal data receipt is similar to a traditional receipt and it is stored to possess evidence satisfying non-repudiation of who, what, and how was agreed.

Kantara Initiative has their specifications on consent receipts ⁴ which has been implemented by several organizations. The specifications create a JavaScript Object Notation (JSON) format for a receipt and in general, it contains the generic fields: the collection method, the jurisdiction, PI categories, and purposes. Kantara is the main proponent of User-Managed Access (UMA) [32] which creates a technology that enables users to manage their data with fine granularity. The emerging standard UMA allows apps to extend the power of consent.

2.3.2 Access Control

According to the data protection by design and by default, it is important to think about data practices. Access control can be a technical solution to protect access to personal data by design using the access control policies. To simplify the process of establishing these needed policies, Agile software development can help. It has dedicated tools to describe requirements such as user stories. Stories are concise and informal descriptions telling who, what and why something is required by users [33].

By understanding who has access to the data at every stage of the data lifecycle it is possible to apply for roles to ensure that only authorized individuals can see and modify specific data. With the constant evolution of the system, it can be appropriate to re-think the access to control to specific users.

User access control is a crucial requirement in any IoT deployment, as it allows one to provide authorization, authentication, and revocation of a registered legitimate user to access real-time information and/or service directly from the IoT devices [34]. On communications among several users, smart devices typically take place over insecure channels. There is a risk that these communications can be intercepted, deleted, or modified. Access control mechanisms can be implemented to help ensure that only authorized registered users are allowed access to the information and/or services.

An access control mechanism can monitor the access activities of resources and ensure that authorized users access information resources under legitimate conditions [35]. This technology can ensure information security and prevent the unauthorized flow of information.

⁴<https://kantarainitiative.org/confluence/display/infosharing/Consent+Receipt+Specification>

The authors of [36] propose a smart contract-based framework with multiple Access Control Contracts (ACC). Each ACC provides an access control method for a subject-object pair, and implements both static access right validation based on predefined policies and dynamic access right validation by checking the the behavior of the subject.

2.3.3 Delegation Problem

The enormous usage of computing systems such as smart devices or mobile devices allows all people to connect every day and anytime with other people. Mobile computing allows users to move freely without necessarily carrying their computing devices that are already available at any location.

Mutual collaboration between users of mobile devices is often required which means that the support for user-friendly delegation is important. For example, in hospitals when a senior doctor needs to delegate his permissions to another doctor when a patient is in a critical situation. In distributed systems, a user often needs to act on another user's behalf with some subset of his/her rights.

In common usage of a system, the quick solution is to share the credentials with another person to delegate the permissions. This is due to the short duration and frequent need for delegation, mutual trust of users, and the complex nature of existing delegation mechanisms that follow the principle of least privileges [37]. Most systems have attempted to resolve such delegation requirements with ad-hoc mechanisms by compromising existing disorganized policies or simply attaching additional components to their applications [38].

According to the [39] least privilege means that "Every program and every user of the system should operate using the least set of privileges necessary to complete the job". Delegation should grant only the right for performing a delegated task [37]. Fine-grained delegation allows the precise transfer of exactly those privileges that are necessary for a specific task.

The authors of [38] consider that delegation is the process how an active entity in a distributed environment allows another entity to access some resources. It is simply defined as temporarily granting a user permissions to perform a specific action [40]. Authentication verifies the user identity and enables authorization. An authorization policy mention what user identity is allowed to do. Therefore, authorization policies define what an individual identity or a group may access, and access control is the method to enforce such policies [41]. Delegation can be made at the authentication or identity level or authorization or access control level. The most common way is to provide delegation support at the access control level.

The decentralized architecture in IoT environments can be a problem for delegation. Access control with permission delegation mechanisms allows fine granular access to secure resources. There are architectures for permission delegation and access control that are either event-based or query-based. However, this assumes a single trusted delegation service, which is likely biased or fails to service. Additionally, they fail to allow users to verify delegation service operations. Thus it cannot be directly applied to IoT due to the low power, low

bandwidth, ad-hoc and decentralized nature [42].

The authors of [40] investigate an authorization and delegation model for the IoT cloud based on blockchain technology. The proposed approach enables the user to audit authorization operations and inspect how access control is performed.

2.4 INTERNET OF THINGS LANDSCAPE

Wireless technologies have been growing actively all around the world. Fifth-generation (5G) network has become an interesting topic in wireless research. 5G will make possible the creation of wireless architecture and smart services. The LTE (4G) will not be sufficient and efficient considering the multiple device connectivity and high data rate, more bandwidth, low latency quality of service, and low interference [43].

The IoT is an important topic in the scenario of modern wireless telecommunications. This concept represents the pervasive presence of a variety of things or objects around us. The main impact of this is the impact that it will have on several aspects of everyday life and the behavior of potential users [44]. The impact of IoT in privacy is detailed on subsection 2.4.1.

IoT transforms the real world objects into intelligent virtual objects. It aims to unify everything in our world with a common infrastructure providing the control of things around us and keeping us informed of the state of the environment and the things [45]. Figure 2.5 represents this representation of everything. Thus, IoT is a computing and communication paradigm where the objects are permanently connected to the Internet. Sensors and actuators are resources constrained devices and enables intelligent systems that obtain information from the physical world, process such information, and act on the physical world accordingly [46].

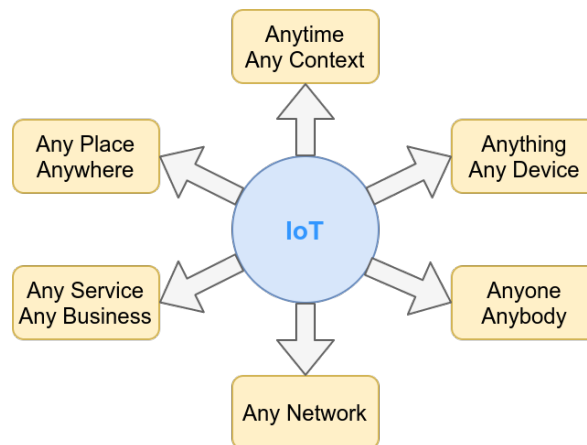


Figure 2.5: IoT: Generic idea

IoT is the network of physical devices, vehicles, home appliances, and a lot of other identifiable items embedded with electronics, software, devices (sensors and actuators), and network connectivity which enables to collect and exchange data across the Internet. It allows objects to be sensed or controlled remotely across existing network infrastructure creating opportunities for more direct integration of the physical world into computer-based systems

and resulting in improved efficiency, accuracy, and economic benefit with reduced human intervention [47].

A possible definition to IoT is about heterogeneous devices and the interconnection of these uniquely identifiable objects [48]. These objects can be sensors, devices or things, and it is cheap and ubiquitous [49]. The future of the Internet will consist of these heterogeneous devices connected which will further extend the borders of the world with physical entities and virtual components [50].

Smart devices usually are sensors or actuators characterized by a tiny microcontroller and a wireless communication interface. Individually, each one has low computational power, however, the combination of their capabilities allows us to develop complex IoT scenarios. It should have several differences [51] related to processing capabilities, communication, technologies, operating system, and implemented functions. The IoT concept is to use the increased connectivity provided by wireless communication technologies, the computing power, and the memory capacity of embedded devices to implement autonomous behavior, which can support the user every day [52].

According to the literature [53], IoT is “an interconnection of sensing and actuating devices providing the ability to share information across platforms through a unified framework, developing a common operating picture for enabling innovative applications. This is achieved by seamless ubiquitous sensing, data analytics and information representation with Cloud computing as the unifying framework”.

IoT connects the physical world to the cyber world using standard and interoperable communication protocols. According to the European Research Cluster on the Internet of Things (IERC) “physical and virtual “things” have identities, physical attributes, and virtual personalities and use intelligent interfaces, and are seamlessly integrated into the information network”⁵.

Several technologies are involved in implementing the idea of IoT such as Radio frequency Identification (RFID), Near Field Communication (NFC), Machine-to-Machine Communication (M2M) and Vehicle-to-Vehicle Communication (V2V) [54]. RFID is an automatic technology and aids machines or computers to identify objects, record metadata, or control individual targets through radio waves [55]. It has some applications such as smart parking, data collection, smart water supply (Wireless network system will enable to monitor the water supply and will help to ensure that there is the adequate water supply for the resident and business use), and smart home. NFC technology is similar to RFID configuration, and it is used in smart meters, patient monitors, agriculture, security devices, street lighting, and environmental sensors [56]. M2M refers to the communications between computers, embedded processors, smart sensors, actuators, and mobile devices. V2V communications involve a vehicle, which acts as a node in a network, and communication is done by the use of various sensors connected in an ad-hoc network. M2M and V2V have applications such as industrial maintenance, smart cars, smart grid, traveling, and health.

⁵http://www.internet-of-things-research.eu/about_iiot.htm

Smart wearables collect and analyze data, and they can make smart decisions and provide a response to the user. The authors of [57] classify the wearables into four major clusters: health, sports, and daily activity, tracking and location, and safety. The health wearable IoT device is mainly used for remote patient monitoring, treatment, and in some cases for rehabilitation purposes. The sports and daily activity group refers to during sports activities to record different metrics of the user/athlete activity to improve his/her performance. The tracking and location refers to find the position of the user, study the trajectory of a senior citizen in a care-home facility, analyze the movement of the people who are visiting an exhibition. Finally, the safety group refers to the wearables that are used to provide a safe environment for the users, such as a fatigue monitoring system that can alert the drivers who fall asleep at the wheel and notify the employers or an environmental condition monitoring.

The presence of smart devices (sensors and actuators) enables smart environments such as smart homes, smart health, smart cities, and smart factories. This document focus on smart home and it is detailed on subsection 2.4.2.

2.4.1 The impact of IoT in privacy

In general, any system needs to guarantee the ability to users selectively share and to determine how personal information is collected and used. Additionally, it needs to guarantee the right to be alone and reserved from others, the ability to control the degree to which one is identifiable when undertaking online or offline actions and the ability to control the data flow.

The IoT amplifies privacy challenges such as the opacity of data flows, and it creates new issues. The growth of IoT includes the increase of online data collection, decreases the private spaces, challenges to meaningful consent, and regulatory issues[20].

The traditional use of web pages, how long the users spend on each page and where they click on the screen is considered as online data collection. IoT brings a data collection considered offline. The IoT enables an increase on the monitoring of human activity with a great number of sensing devices and sensor types as well, with a greater proximity of sensing devices to people's bodies and intimate spaces such as smart homes. These characteristics can affect the users privacy and behavior. If a family stays on their living room watching TV and the TV is watching them, how can it affect the family behavior? How can we manage the consent for this? The traditional notions of privacy and boundaries are affected. In a traditional home, the user can close his door and feel that he controls his privacy. Network-connected devices in a private space can remove this sense of control and privacy. With the rise of ubiquitous data collection throughout the human environment, the notion of a private space may erode, and the ability to know who is observing may cease to exist.

Like described on subsection 2.3.1 the consent is an important requirement to guarantee the user privacy. Even if users consent the use of a device, whether they are knowingly consenting it is often unclear. Typically, consent for data collection by IoT devices occurs in the following way: users are presented with lengthy privacy policies up front and are given a binary choice to fully consent or not use the product. Users have a little to no opportunity to withdraw consent.

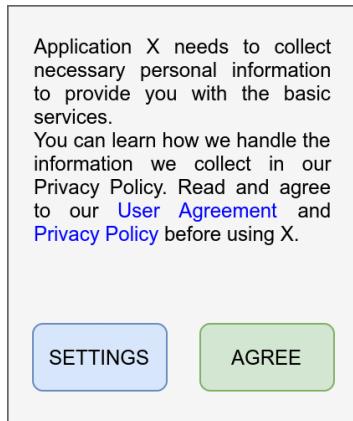


Figure 2.6: Example of a consent request

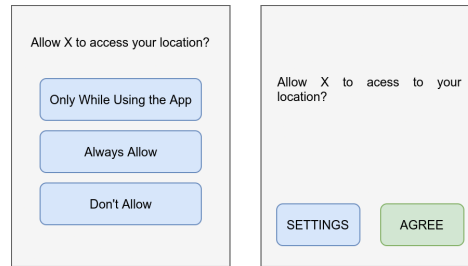


Figure 2.7: Example of a periodic notification

Normally, IoT devices have small screens or do not have it at all. In this way, users cannot easily change privacy settings or access details about what data they are sharing. Few devices include a privacy policy in their physical packaging. Instead, manufacturers provide links to websites where the privacy policies are often difficult to find or insufficiently address privacy issues related to the device.

Manufacturers of IoT devices can help improve privacy standards by adopting practices or adding features that give users greater control over the data collected about them. Companies should be transparent and not collect or use data in ways that violate peoples expectations. Companies should commit to protecting users privacy by only collecting data for which they have specific uses, versus hoarding it for some unknown, future use, and by deleting the data when it is no longer needed. In addition, users should be given more power to update their privacy settings during the pre-collection or post-collection phases.

To provide users with greater control, IoT products should build in “Do Not Collect” switches or permissions, which would allow users to turn off data collection. This is the case of devices such as microphones where the user can mute or use “wake word” or manually turn off activities collection.

It is possible to give users greater control by allowing them to withdraw consent to store data that has been previously collected. The GDPR requires that revoking consent must be as easy as granting it, an obligation that strongly supports user choice and control. Companies must also ensure that data are properly encrypted as they are transmitted and after they have been received and stored while giving users easy means to delete personal data.

At the first usage of an application, it is common for applications to ask for the consent with the policy and the links for more detailed information about it, as described before (similar to Figure 2.6). However, during the usage of a system or application the notifications are very important to inform the user about the permissions to collect data (similar to Figure 2.7).

These periodic notifications allow a greater user control, and it needs to be transparent and as useful as possible. There are different kinds of notifications with different purposes:

- Occurs when a user can decide in real time if he wants to agree to sharing certain data

- Regular reminders about ongoing data collection practices can allow users to reaffirm or cancel their consent at any time
- Notifications can be customized based on a users context
- This approach separates the granularity of notifications over time to give the user more information at the right time, and less when its likely to be glossed over.

It is not enough, however, for manufacturers to simply update the timing of their notifications. They must also ensure that consumers understand these notices.

There is a notion of identity in a system to control who can access to specific data, authorizations and log in information. Different users of the same devices should be able to create separate profiles with different privacy settings. Users should be able to easily switch between profiles and delete profiles that contain collected data. Devices with multiple users should separate profiles and their data collected from each user.

Selective sharing is an essential privacy topic for the IoT. Social networks allow people to share data provided from IoT devices. But people want to share this data selectively such as with friends, family and doctors. Privacy dashboards can allow users to see, understand, and control the use and sharing of their personal data.

2.4.2 Smart Homes

Nowadays, we are living in a smart environment and technology makes part of our lives. Technology makes our life easier, and it can be used at home to have automation, environment sensing, and actuators. Using devices such as desktop, laptop, tablet, or smartphone [58] we can see and manage the collected data and there are a huge number of devices to collect data such as light fans, air conditions, or smart security locks [59]. The term “smart” refers to an innovative technology that uses some degree of programmability, and it can acquire information from the surrounding environment and react accordingly by controlling the behaviour of the actuators. Smart technologies present in a house make it possible to monitor, control, and support residents, which can enhance the quality of life and promote independent living [60]. These smart Internet-connected devices provide safety, security, and convenience. Figure 2.8 depicts the idea of collecting data, processing, and actuating according to the data collected.

The smart devices available in a smart home can be controlled from outside the home or via the Internet. The main idea is to provide sophisticated information about the state of the home in any place and allow the user to control the connected devices also in any place. The authors of [61] describe Smart home technology as normally refers to a set of devices, appliances or systems, that is connected to a common network, which can be controlled independently or remotely.

The smart home can provide easy access to things across the home such as voice control, a simple touch of a button, or use a unique interface across the home. There are several simple and low-cost alternatives to implement from a turn off/on the lights or use security cameras. In the past, surveillance systems were limited to commercial enterprises. Nowadays it is possible to monitor the house using real-time and with high-definition cameras and it is possible doing this using a smartphone. In this context, there are devices such as smart door

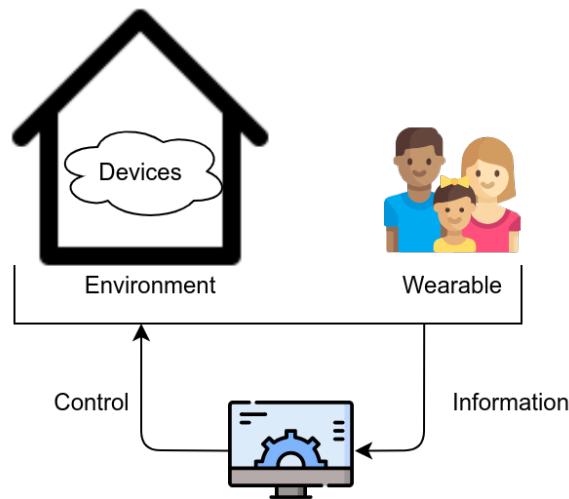


Figure 2.8: Generic smart Home working

locks, garage openers, video cameras, night vision, door and window sensors, and movement and fire sensors. Security systems can be self-monitored or monitored by a third party. A self-monitored system communicates with the user (communication between user and system) and the user and the data being collected can also be stored in the cloud.

The most common smart devices are the traditional sensors such as temperature and humidity sensors. But a smart home includes devices such as smart TV which is any television with the capacity to be connected to the Internet to access streaming media services and that can run entertainment applications such as web browsers or video-rental services. Smart TVs can be smart if they have an internal microprocessor and Internet capability or regular TVs made smart by being connected to a system to provide a way to enable Internet access and streaming. Smart tv can connect to many other devices wirelessly.

2.4.3 Privacy on a smart home

The huge number of smart homes and consequently the huge number of users' information on the Internet creates several challenges for preserving users' privacy [62]. External entities such as passive network observers can infer private activities inside the home by analyzing the traffic for different purposes. Additionally, the sensors can capture users offline information and send it outside the home. Thus, the devices can regard sensitive and private information and share it on the Internet.

The increasing use of smart home devices affects the privacy not only of device owners, but also of individuals who did not choose to deploy them, and may not even be aware of them [63]. The spread of these technologies causes a growing concern about what sensitive data is collected and what it is used for [64].

Smart home assistants, such as Amazon Echo or Google Home, have increasingly been present in consumer households in recent years due to their interactive digital services and user-friendly interfaces. The majority of it uses voice command devices using microphones. Users interact with virtual assistants primarily via a voice user interface that acquires information

via microphones and provides information through audio speakers [65].

Some devices, such as home security cameras are developed to be always on. Many others use microphones but are not necessarily always listening, recording, or even retaining information. The authors of [66] defined three general categories of devices with microphones: manually activated, speech activated, or always-on devices. Each category presents different privacy implications. Another key issue is whether the device is used for voice recognition, the biometric identification of an individual by the characteristics of his voice, or for speech recognition.

There are privacy implications of microphone-enabled devices. Many devices have a microphone to interact with users or simply to act as a sensor of the environment. The microphone could be manually activated using a button or other intentional physical action. Also, a microphone could be speech activated requiring a spoken “wake phrase” without transmitting or storing any information until it detects the word or phrase that triggers the device to begin actively recording. Devices had to be awake to process sensory input. When a modern smartphone is in a passive state the microphone can listen, buffering and re-recording every few seconds to try to detect the wake phrase. Thus, the microphone is just another environmental sensor and can detect when the user is in a noisy situation and adjust its ring volume accordingly without the need to record, transmit or save audio. Finally, it could be always-on devices. Always on devices are designed to constantly transmit data, including devices that “buffer” to allow the user to capture only the most recent period.

Cameras can also be used to monitor people and their activities within or near the home without people’s knowledge. The users have their daily activities and behaviors measured, recorded, and analyzed. Thus it is necessary for the developers and policy-makers to provide ways to inform consumers and citizens about who collects what kind of personal information, how it is stored, used, and disclosed to whom, and for what purposes. According to the privacy principles, the users should be able to keep control of their data and be able to opt-out of the smart environment without negative consequences. It is difficult to know what information is being collected, used, and disclosed by devices in a sensor network. Sometimes, it is also difficult to learn about the parties that benefit from the information collected with these devices. Moreover, it is important to mention that home is where the people spend more time, and it is considered the most private space.

Information collected by sensors creates a huge amount of data that can be combined and analyzed, potentially without adequate accountability, transparency, security, or meaningful consent. The motivation to track a device is to understand the behavior of the individual behind the device. Information as activities, movements, and preferences can be regarded.

These digital assistants and smart home devices allow companies access to the private home. It includes information such as sleep habits, listens in on dinner conversations, and track when users shower making complete profiles of their users to help them more accurately server targeted ads [67].

Another relevant information is related to metadata. Metadata is the data that provides information about other data which deserves privacy protection. The metadata can be

generated when the user makes a phone call, sends an email, on social networks, or in Internet browsing. For example, a Uniform Resource Locator (URL) that specifies the address of the web page an individual requested and it is metadata that indicates the content of the website. However, sometimes the metadata can reveal more than the content itself because it can be used on search engines to identify individuals and to reveal sensitive information about them.

2.4.4 Solutions for smart homes

Information and communication technologies are changing the life of a modern person in several aspects including, for example, the opportunity to communicate with technological appliances in natural language. [68] such as Amazon Echo/Alexa and Google Home/Assistant that have made our daily routines much more convenient [69]. There has been a huge growth in the number of smart devices for smart environment automation helping in the common routines [70]. Generally, the smart devices have the ability to dynamically adapt to the changing contexts and take actions based on their operating conditions, they should be self-configuring and interoperable, having unique identities and being able to communicate and exchange data with other devices and systems [71].

Smart home devices include sensors, actuators and surveillance cameras. Sensor is a device used for the conversion of physical events or characteristics into electrical signals. This is a hardware device that takes the input from environment and gives it to the system by converting it. The most common examples of sensors are the temperature, humidity and pressure sensors. The sensing of the environment makes possible to manage and improve the quality of the environment. Regarding information about the conditions it is possible to make decisions about it. Actuator is a device that converts electrical signals into physical events or characteristics. It takes the input from the system and gives output to the environment. Depending on the characteristics of the environment it is possible to decide which is better for it. Moreover, it is possible to have automations in the environment such as at a specific time of day the windows will be closed.

The unified control platform integrates and manages all smart devices also supporting automations. It is a system to control the internal and external factors of a smart environment such as thermostats, light bulbs, door locks, motion sensors, TV streaming devices and indoor/outdoor security cameras [72]. The unified control platform is able to set lighting, temperature, music, and TV programs differently depending on whether a person is at the smart environment and a users experience and preference [73]. Smart assistants, as an unified control platform, such as Amazon Echo, Amazon Echo Show, Google Home, Google Assistant, Microsoft Cortana, and others, interact and control appliances. Google Home and Amazon Echo include functionalities as music playing, web search, reminders and timers, and voice command controls for devices connected to it. Furthermore, there are commercially available smart home hubs, such as Samsung SmartThings, which provide centralized control for devices that may come from different manufacturers [69]. Additionally, devices and appliance manufactures have devices that can interact with digital assistants thus making it easier to control or send controls to them via commands [74].

There are several open-source technologies to use as a unified control platform to integrate all smart devices and manage the smart environment. Possible alternatives to use as the unified control platform are detailed in Table 2.1.

Table 2.1: Digital assistants alternatives

Platform	Open Source	Program language
openHAB ⁶	Y	Java
Home Assistant ⁷	Y	Python
IFTTT ⁸	N	ND
Domoticz ⁹	Y	C++
Home ¹⁰	N	ND
Jeedom ¹¹	Y	PHP
Control4 ¹²	N	ND
EVA ICS ¹³	Y	Python
Switchur ¹⁴	N	ND
Homify ¹⁵	Y	PHP
Gladys Home Assistant ¹⁶	Y	JavaScript
Mozilla WebThings Gateway ¹⁷	Y	JavaScript
Freedomotic ¹⁸	Y	Java
SEQUEmatic ¹⁹	N	ND

The most relevant open-source technologies applied to smart homes are the OpenHAB, Home Assistant and Domoticz. OpenHab is an open source home automation platform. It is vendor and technology agnostic resulting that if a device is popular, it will likely be supported by the platform. It is developed in Java, which provides endless devices where it is possible to run the system. The community is very well established and helpful and its architecture is based on bindings that bring support for different smart home devices. Home Assistant is a growing community with a very user-friendly approach. The tagline of this home automation software is simplicity. It is developed in Python and it is possible to extend its functionality by using plugins. Domoticz is very lightweight compared to OpenHab and Home Assistant while still delivering a decent number of features. The configuration is mostly done through a web interface and, as it happens in Home Assistant, it is possible to use plugins to extend the

⁶<https://www.openhab.org/>

⁷<https://www.home-assistant.io/>

⁸https://platform.ifttt.com/solutions/product_overview

⁹<https://www.domoticz.com/>

¹⁰<https://www.apple.com/ios/home/>

¹¹<https://www.jeedom.com/en/>

¹²<https://www.control4.com/solutions/control4-app>

¹³<https://www.eva-ics.com/>

¹⁴<https://switchur.com/>

¹⁵<https://github.com/markushaug/homify>

¹⁶<https://gladysassistant.com/en/>

¹⁷<https://iot.mozilla.org/gateway/>

¹⁸<https://freedomotic.github.io/>

¹⁹<https://sequematic.com/>

functionality.

Home Assistant is a free and open-source home automation software designed to be the central control system for smart home devices with focus on local control and privacy. Devices, services and IoT technologies are supported by integration components using a huge range of possible protocols such as MQTT, Bluetooth, or ZigBee. The front-end dashboard is called Lovelace, which offers different cards to display information and control devices. It is a fast and powerful way for users to manage their home using their mobiles and desktops. The interface is fully customizable using the integrated editor or by modifying the underlying YAML code. YAML is a human-readable language used mainly in configuration files. Additionally, it is used in applications where data is being stored or transmitted. Home Assistant acts as a central smart home controller hub by combining different devices and services in a single place and integrating them as entities. The provided rule-based system for automations allows creating custom routines based on a trigger event, conditions and actions, including scripts. These enable building automation, alarm management of security alarms, and video surveillance for home security systems as well as monitoring of energy measuring devices. In terms of security, there is no remote access enabled by default and data is stored solely on the device itself. User accounts can be secured with two-factor authentication to prevent access even if the user password is known by the attacker. Add-ons get a security rating based on their access to system resources. Home Assistant provides a user-friendly and intuitive interface to interact with the devices. Home Assistant is a complete and useful platform that provides a large number of important features to manage a smart environment such as:

- The home Assistant platform provides several different cards to place and configure;
- A dashboard editor that allows the management of the Lovelace dashboard including a live preview when editing cards;
- The configurations can be done fast;
- It is very customizable because cards have several options which help to configure the data as required, several themes and it provides the ability to override names and icons of entities.

Moreover, Home Assistant supports a huge number of possible integrations to facilitate the integrations of smart devices. Some examples of possible integrations are Google Assistant, Shelly, Amazon Alexa, MQTT, ZigBee, and others.

2.5 RELATED TECHNOLOGIES

A study developed by [75] intends to investigate the user's reasons for purchasing IoT devices, perceptions of smart home privacy risks, and actions taken to protect their privacy from those external to the home who create, manage, track, or regulate IoT devices and/or their data. The results of this study highlight some recurring themes: "First, users' desires for convenience and connectedness dictate their privacy-related behaviors for dealing with external entities, such as device manufacturers, Internet Service Providers, governments, and advertisers. Second, user opinions about external entities collecting smart home data depend on perceived benefit from

these entities. Third, users trust IoT device manufacturers to protect their privacy but do not verify that these protections are in place. Fourth, users are unaware of privacy risks from inference algorithms operating on data from non-audio/visual devices”. The authors of [76] study how people desire to protect their privacy in the smart home context. They identified the important factors such as data transparency and control, security, safety, usability and user experience, system intelligence, and system modality. Finally, they discuss how these factors can guide the design of smart home privacy mechanisms.

According to the literature, there are different proposed approaches trying to solve the existing problems when common users access to smart home devices. The authors of [77] propose a privacy-preserving scheme named PrivHome. This service provides data confidentiality as well as entity and data authentication to prevent an outsider from learning or modifying the data communicated between the devices, service provider, gateway, and the user. Moreover, there are some proposed approaches considering the complete data lifecycle since data generation, transfer, storage processing and sharing. The authors of [78] propose a solution for reliably concealing privacy and ensuring security for analytics of smart home sensor data. They replace the personal/quasi-identifiers of collected sensor data with hashed values before storing them into a de-identified storage.

Serious privacy concerns occur when smart devices are generating data that may reveal personal information about user’s inside the smart home mostly when this data is shared with the cloud which can include untrusted third party cloud services are accessing PII. The authors of [79] propose an architecture to integrate Attribute Based Encryption (ABE) schemes to the Openhab. They measure the performances and the overheads of the proposed solution.

The authors of [80] propose an approach to solve the challenges related to security, authentication and privacy created due to the internet connected devices, dynamic and heterogeneous nature of the smart home environment. This approach is based on the use of blockchain technology, which is called Smart Home based the IoT-Blockchain (SHIB) and describes an approach to data privacy in the smart home using blockchain. According to their results, the proposed approach solved the mentioned challenges in the smart home environment such as data privacy, trust access control, and the ability of extension allowing users to build the privacy policies in ACC, to be stored on Ethereum blockchain network, and only the creator of ACC can add a new policy, update or delete the policies in ACC.

According to the literature, there are other approaches trying to solve these privacy concerns. The authors of [81] developed a Python library to easily integrate privacy-preserving traffic shaping that replaces standard networking functions with versions that automatically obfuscate device traffic patterns. In addition, the authors of [82] measured the normal traffic patterns generated by the smart home devices and identified possible privacy vulnerabilities. Thus, they designed a smart home solution to obscure the real network traffic with synthetic traffic. The authors of [83] propose a dynamic method for switching the level of privacy in the environment based on the context and the situation in the environment ensuring the user’s safety and privacy. The idea was to decrease the invasiveness of the technology considering the purpose

of the system. The authors of [21] propose a study about how software defined networking technology can be used to dynamically block/quarantine devices, based on their network activity and on the context within the house such as time of the day or occupancy level. Their idea is to have an approach that can augment device-centric security for the emerging smart-home. The authors of [84] employ a smart home IoT architecture that enables users to interact with it through different devices that support smart house management, and they analyze different scenarios to identify possible security and privacy issues for users.

2.6 CONCLUSION

In this chapter we described the different definitions of privacy and, a provided a limited view of the relationship between privacy and security. We explored the concept of user-centric design and solutions, which includes the definition and usability aspects of consent, access control, and delegation. In addition, we focus the privacy concerns into IoT scenarios, which will be the target of our work. In this sense, we presented relevant considerations concerning the impact of IoT in privacy, presented important topics concerning a smart home as a smart environment, and presented possible solutions for these smart home scenarios. Finally, this chapter concludes with a description of the technologies that are related to the work developed in this dissertation.

CHAPTER 3

The CASSIOPEIA project

3.1 INTRODUCTION

The dissertation presented in this document was developed in the scope of the CASSIOPEIA project. This project intends to solve the SmartBnb problem where the user rents a smart home for a limited period of time. In this chapter, we describe the main important information about the CASSIOPEIA project such as the use cases, architecture, and the problem.

CASSIOPEIA project has a relevant contribution to increase the user's privacy in IoT environments. However, the scope of the project does not include a user control of his personal data. The project is about the privacy management and demonstrating a way to ensure the user's privacy during the stay.

This chapter is organized as follows: section 3.2 described the important topics about the CASSIOPEIA project (more details about the project are described in chapter A); the problem that was the base for this dissertation is described in section 3.3; and the proposed solution to the mentioned problem is detailed in section 3.4.

3.2 DESCRIPTION OF CASSIOPEIA PROJECT

CASSIOPEIA is a project that started in 2020 November and ended in 2021 April, funded by NGI Trust ¹ which aims to build a technical demonstration about how we can use open-source technologies to create usable and transparent architectures enabling device owners to selectively collect, share and retain data from users, while delegating control of device features to the users from whom data is being obtained.

The idea of the CASSIOPEIA project was to develop a demonstrator applied to the SmartBnb problem where a guest wants to rent an apartment for a limited period of time, containing IoT devices. This technical demonstration refers to selective sharing and feature delegation, granular consents, transparency, and non-repudiation. Smart homes include a range of smart devices for sensing and automation purposes. All of these smart devices collect data of different degrees of sensitivity.

The concerns about the SmartBnb problem are about the temporary usage of the smart environment. CASSIOPEIA demonstrates a way to give temporary access to all smart home devices to a guest. All these devices are controlled using a unified control platform. Moreover, this unified control platform is configured based on the privacy principles established to ensure the user's privacy.

In the CASSIOPEIA project, we can identify two different types of users. The Owner is the host of the smart environment. In this case, the Owner is the smart homeowner. Another user is the Renter. This user is who rents the smart environment for a limited period of time. In the project, the Renter is the user that rents the smart home for a stay in the smart home.

The renter should be able to select which devices are collecting data during his stay and he should give consent for it. In addition, it is essential to ensure that the owner has no access to the devices during the stay of the renter. CASSIOPEIA tries to solve these problems, developing a service capable to provide all the information to the renter before the stay,

¹<https://www.ngi.eu/ngi-projects/ngi-trust/>

configures the unified control platform according to the renter options, generates a receipt as a proof of consent that contains the relevant information about the stay, and ensures that during the stay the owner cannot access the unified control platform. As the owner cannot access the unified control platform, then the owner cannot control the devices neither can he see the data collected by the smart devices.

The generated guest data must have a lifecycle that includes a provable deletion. This implies non-repudiation of consent and consent revocation. All processes should be transparent, trusted, and cryptographically secure.

The owner gives degrees of access to the renter, for a limited period of time, using short-lived permissions. Selected data will be collected, with the renter's consent, stored for a limited period of time, and then deleted when the renter checks out. The renter can use the devices and access the data, but it cannot repudiate the permission he granted, and the owner notifies the renter when the data are deleted.

The human-centric view of IoT and smart homes means strong privacy which includes confidentiality and a flexible sharing arrangement that aligns with social norms for information sharing and user-centric design principles. Moreover, comprehensive privacy means having visibility into data flows. The Renter should feel safe and empowered about the data collected on him by other peoples devices.

In short, smart home device privacy and sharing arrangements currently are inconsistent, piecemeal, inconvenient, and/or controlled by large internet companies who have their own visions of how people should and should not be able to share their data. Today, IoT devices collect data without displaying how that data are treated in the background, nor when they have been deleted. The owner/renter identity relationship is absent from todays smart home architecture.

CASSIOPEIA relies on applying existing technology to enable a broad set of interactions and an expansive conception of privacy. CASSIOPEIAs architecture prevents the privacy characteristics of IoT data sharing without being tied to any particular companys product roadmap, or business model. Our proof-of-concept is necessary innovation to show developers, policymakers, and the public what privacy-by-default and -design looks like when united with the core NGI principles of an open, trustworthy, human-centric internet ².

During the stay, the smart devices presented in the smart home will collect the renter's data. In this way, the Renter should provide the permissions for data collection. As proof of this consent, the Owner requests for a receipt that is sent to the Renter to confirm the given consent. During the stay, it is required that the Owner cannot access the Renter data collected by the smart devices. At the end of the stay, the Renter should be able to request the data deletion. More details about the possible deletion are provided in the following sections of this document. The Figure 3.1 depicts the interaction between the two different users and the CASSIOPEIA project through the web interface.

The project has a component responsible for the smart devices controller and the access control to different views between Renter and Owner. As previously said, the smart home

²<https://www.cassiopeia.id/>

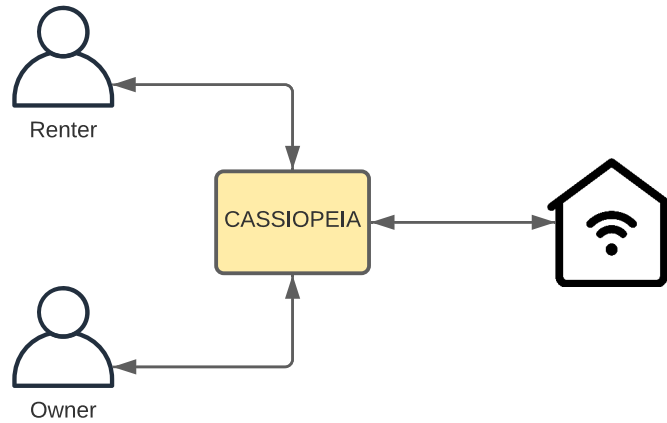


Figure 3.1: Interaction between different users, CASSIOPEIA project and smart home

is controlled by a unified control platform. We used the Home Assistant as open-source technology and developed an external module to manage all permissions (between renter and owner), receipts, and flow data control.

3.3 PROBLEM SCOPE

The focus of the CASSIOPEIA project is the user’s privacy management, which included the possibility of the user to select the devices that are allowed to collect data during the smart environment usage and the possibility to have a receipt as a proof of consent. In addition, the project’s demonstrator focus on guaranteeing that the user’s views are different from the Renter and the Owner ensuring renter’s privacy during the stay. Additionally, a service to data management was added to provide the user the ability to request the data deletion after the stay. This service is also responsible for notifying the user when the data is removed. However, there are some problems out of the scope of this project and they are relevant to ensure the user privacy in any smart environment usage. In this way, this dissertation focus on identifying these problems and proposing a solution for each of them.

The main problem faced during the project implementation was how can we ensure that privacy is being preserved? In the proposed demonstrator in the CASSIOPEIA project there is no way to confirm which devices are collecting data during the stay. In addition, the project provides the ability to request data deletion and notify the user when the data is removed but there is no way to confirm if the data were effectively removed. Moreover, the data subject has no way to access the information provided nor the consent information. Moreover, after the stay, the data subject has not control over his data.

Another problem faced is the missing of an option to export data considering the data portability legislation. According to the data portability legislation exposed in the GDPR ³ “The data subject shall have the right to receive the personal data concerning him or her, which he or she has provided to a controller, in a structured, commonly used and machine-readable

³<https://gdpr-info.eu/art-20-gdpr/>

format and has the right to transmit those data to another controller without hindrance from the controller to which the personal data have been provided". In addition, in this regulation, it is mentioned that 'the data subject shall have the right to have the personal data transmitted directly from one controller to another, where technically feasible.' In this way, it is important to provide a feature to export personal data.

The remaining problems found are about the receipts and the way how they are processed and stored because the receipts are stored in a decrypted way and without any signature. Storing the receipt in a decrypted way can compromise the user's privacy and it can be enough for an inference attack because the receipts has no sensitive data, but it has a large amount of stay and user information. Without any signature the receipt is not associated to any identity and in the project there is no way to ensure the receipt integrity such as a hash of the receipt. The receipt is associated with the user of the system, but it is impossible to validate the integrity of the receipt. Another problem with the integrity of the receipt is related to how we can ensure that after the user's consent the content in the receipt is the same if it is not signed? Within this dissertation, we need to ensure that the content of the receipt is immutable and the user's signature and respective validation contribute to this guarantee.

3.4 PROPOSED SOLUTION

In this dissertation, we focus on increasing the user's privacy and increasing the transparency about user's personal data processing. In addition, we consider including the ability to provide a simple and clear way for the user to manage his personal data. This way, we propose a solution that can be integrated in any smart environment, increasing the smart environment quality in terms of privacy. However, in this dissertation we consider the implementation of this solution in the SmartBnb problem where the smart environment is a smart home and the application occurs in the CASSIOPEIA project's architecture.

The solution proposed relies on a demonstrator of how we can provide a private smart environment where the data subject has a large level of data control and transparency about personal data processing. Moreover, the demonstrator focus on preserving the user's privacy and control during the stay and after the stay. This control includes the ability of the user to access his data, control which entities have access to his data and how these entities are involved in the personal data processing. In addition, the proposed solution ensures the user rights about his privacy including the ability to request data deletion, the request for data portability, and the notification about what happens with his personal data.

The consent concerns are a relevant topic to take into account in the proposed approach. The receipt works as a proof of the given consent before the smart environment usage. This way, this dissertation focus on the receipt generation and its storage including the receipt security and its integrity during the receipt storage process. Moreover, the approach proposed considers the possibility to add more entities to the system after the consent provided. In this case the demonstrator depicts how we can notify the user and request a new consent.

3.5 CONCLUSION

In this chapter, we briefly described the CASSIOPEIA project, which provided a basis for the work developed in this dissertation. The underlying mechanisms and components proposed in the project are also briefly described, as well as the broad strategy followed to solve the SmartBnb problem. Further chapters will consider how this effort can be improved in order to achieve better privacy and overall security.

Architecture and Specifications

4.1 INTRODUCTION

The dissertation was conducted in the CASSIOPEIA project and focus on providing a demonstration of how we can achieve a system that enables the data subject to manage his information transparently and reliably. The demonstrator should highlight the ability to manage all user's data accurately and only the data subject should be allowed to access his data. Namely, in our chosen scenario, the subject data is related to the data collected by the smart devices during the stay in the smart environment and stay's information such as stay identification, receipts as proof of the consent, and user's identification (name, email, and check-in and check-out dates). In this way, the *Data Manager Service* and the *Receipt Manager Service* was improved. Moreover, a new component was added to increase the reliability of the consent receipt.

In this chapter, we describe the architecture and the specifications of the system. Thus, the chapter is organized as follows: section 4.2 describes the use cases and the scenarios of the system; the established requirements are described in section 4.3; the defined architecture is presented in section 4.4 and the workflow that represents how the different components should communicate is described in section 4.5; section 4.6 describes how the persistence should be ensured in the system; and the section 4.7 describes how the system should behave in terms of permissions according to the different user roles.

4.2 USE CASES AND SCENARIOS

Assuming smart homes as the smart environment, in this section the use cases and the scenarios for this dissertation are described. These scenarios and use cases intend to improve the solution mentioned previously (chapter 3) to the SmartBnb problem described in chapter 1. SmartBnb problem is about the temporary usage of a smart environment and how the data subject can control his data. The idea is to establish how the user can use the system and how the system should behave.

Creating a new stay in the system is the first step of the temporary usage of the smart environment, in this case, the smart home. To create a new stay in the system, the future Renter should provide his identification such as name, email, and check-in and check-out dates in the smart environment. The Renter should accept the privacy policy of the system to use the system. Both information should be available to the data subject. The data subject should be able to access his stay's information at anytime. Table 4.1 describes the use case about this action. The privacy policies information should be available to the data subject. It is detailed in Table 4.9.

Personal data has a life period in the system established during the check-in and accepted by the data subject. However, the Renter should be able to request the data deletion before the end of this period increasing the user's control. While this life period does not end, the Owner can decide to approve the request for data deletion and delete the data or he can decide to reprove the request and not remove the user's personal data. When the lifetime of

Table 4.1: Use Case 1 - Create new stay

Actors	Renter and owner
Pre-requirements	Ability to create a new stay in the system. The user should be able to access the system.
Rationale	The renter should have access to the stay information.
Demonstrates; Distinction	Data control and transparency
Scenario	After the check-in the renter accesses the system and request for the stay information. The system returns information on the user identification such as email, the dates of the stay (check-in and check-out) the receipt and the type of the smart environment.

the data is over then the data should be deleted, but until then in case of a request by the Renter it may or may not be deleted. This use case is described in Table 4.2.

Table 4.2: Use Case 2 - Request the removal of renter’s personal data

Actors	Renter
Pre-requirements	Ability to access the system to request the data deletion. It includes the renter authentication verified by the service. Moreover, the renter should must have access to the email given during the check-in.
Rationale	The renter should be able to request the data deletion. The data collected during the stay belongs to the renter. In this way the renter should have control over his data.
Demonstrates; Distinction	Data control
Scenario	The renter and his family left the smart home. Using the mobile phone, the renter accesses the system and after the authentication, he requests the data deletion. After the request, the renter receives an email confirming the request.

It is required to have a notification system to notify the user whenever necessary. In this way, we can increase the user’s data control and transparency in the data process. As mentioned, the data subject can request data deletion. It is also mentioned that the data can follow one of the possible approaches:

- removed permanently from the system before their life period;
- removed only when the life period is over;
- anonymized

In all of these possible actions, the user should be notified and informed about what happens to his data. This use case is described in Table 4.3.

Data anonymization has been defined as a “process by which personal data is irreversibly altered in such a way that a data subject can no longer be identified directly or indirectly, either by the data controller alone or in collaboration with any other party”. Table 4.4 has the use case that refers to the anonymization ability of the system.

As mentioned above, the Owner can decide not to remove the data until their lifetime ends. In this case, the Owner can choose to anonymize the data. The anonymizing process must take some considerations into account to be done correctly. Only the data that cannot identify the data subject should be anonymized. The anonymization process includes dissociating the user’s identity from the data collected. After the anonymization process, the user’s identity cannot be inferred or identified.

The anonymization process requires some particular specifications to ensure that the user’s privacy is not compromised. It is possible to identify a pattern through all data collected

Table 4.3: Use Case 3 - Notification about data removed

Actors	Renter
Pre-requirements	Ability to access the system to check the notifications. It includes the renter authentication verified by the service. Moreover, the renter should have access to the email given during the check-in.
Rationale	Any action about his personal data should be known by the data subject. The renter should know what happens with his personal data. When the data are removed from the system, the renter should be notified to check the action.
Demonstrates; Distinction	Data control
Scenario	After the stay, the system notified the data subject about data deletion. The data subject receives a notification by email. This notification informs the user about how his data was removed and when it happened. The Renter accesses the system to check if this notification is true. The Renter requests for the state of his data and can confirm that the data were removed from the system. Furthermore, the Renter can request the data of the stay and verify that the result is null because the data were removed.

during a stay. For example, a temperature cannot provide a direct user identity but can provide a pattern of data subject behavior. This pattern can be combined with other information and makes possible the identification of the data subject identity. Moreover, if a hacker discovers the stay information he can associate the data in sequence with the data subject. In this way, we have two possible problems:

- Data cannot be anonymized because it is possible to build a pattern.
- Independently of the anonymization, the data sequence can be a problem in case of an attack of the system.

Considering these problems, we can have two possible solutions that can solve both problems. When it is possible to calculate the average of the smart device's data, it is possible to store only this average and not the real values. Thus, we can avoid the pattern identification and the problem of the data sequence. In the alternative, it is possible to shuffle the user's data that will be anonymized.

The first solution is easier to implement than the second one because the unique required action is to calculate the average of the values and store it. The second solution requires the selection of data that was previously shuffled and mix the new values and shuffle them again. In this way, this operation is also computationally less efficient but in terms of privacy is better and more appropriate.

The Renter should be notified about the anonymization process to ensure data control and transparency. Moreover, the Renter should be informed about how this data will be anonymized.

The Renter should be informed about the type of devices present in the smart environment. This list should contain the name of devices, characteristics, and features. In this way, we can ensure that the Renter is informed about the smart devices and can manage his privacy adequately. The Renter should be able to select which devices are interesting to be enabled during the smart environment usage. Only the accepted devices are allowed to collect and store data during the stay. The remaining devices should be disabled.

Table 4.4: Use Case 4 - Data anonymization

Actors	Owner
Pre-requirements	The owner should access the system.
Rationale	The owner can decide not to remove the renter's personal data when the renter requests data deletion. However, the owner does an anonymization.
Demonstrates; Distinction	Data transparency and data control
Scenario	After the renter's request or data deletion, the owner decides to remove all the data that identifies the renter and maintain the data for statistic purposes. Renter is notified about the anonymization process and he can verify which data were removed and which data were maintained without identification.

At any moment, the Renter should be able to access the list of the accepted devices through the system. Moreover, the Renter can request a list with the name of the entities that are collecting data. In this way, the Renter can confirm if the list of the entities that are collecting data is equal to the list of the accepted devices. In this way, the Renter has more control of his personal data. After the stay, this feature continues available. The Renter, when the stay ends, can access the system and request both lists. Thus, transparency is ensured if the user has access to the list of smart devices and can confirm which devices are effective regarding data. This use case intends to present the functionality to confirm the entities that are collecting data or that collected data. This use case is described in Table 4.5.

Table 4.5: Use Case 5 - Renter verify devices

Actors	Renter
Pre-requirements	A device capable to access the system; Ability to ask for the list of devices collecting data
Rationale	This is the basic action to give complete transparency to the user. The user can access the system and see effective which sensors are collecting data independently of the devices shown in the interface of the platform to unify control
Demonstrates; Distinction	Data flow; Transparency
Scenario	The renter is using the smart home and interacts with it. Smart devices are collecting personal data. At any moment the renter decides to check which devices are collecting data to check if these devices are only the accepted ones. Renter accesses the system and can ask for the list of the devices that are collecting data.

The system should provide a distributed and unique point of control of personal data. In this way, the user can have all the information about the stays in a unique platform accessible at any moment. Independently of the platform used to access the smart environment the system should provide all the information as long as it is integrated.

The Renter may request all the information about all stays. The system should return a list with the identification of the stay, data collected, devices that collected data, state of the devices, and the request about data deletion and receipts. This use case is described in Table 4.6.

At any moment, while the renter's personal data is not removed from the system, the renter should be able to export the personal data to a common file. This use case is described in Table 4.7.

The system should provide an efficient way to manage personal data from different smart environments. This system is a unique point of control in the user's perspective that allows the integration of different information provided from different kinds of smart environments.

Table 4.6: Use Case 6 - List all information regarded by all the stays

Actors	Renter
Pre-requirements	Ability to access the system to request all data associated with the stays. It includes the renter authentication verified by the service.
Rationale	The renter should be able to request the list of all the information about the stays and respective receipts.
Demonstrates; Distinction	Data control
Scenario	The renter decides to obtain all information about his stays including data collected by the devices and the receipts. To obtain this information, the renter accesses the system. The system returns a list with the identification of the different stays and the respective information such as the state of the receipts, information about the data deletion, and the type of the smart environment.

Table 4.7: Use Case 7 - Export personal data

Actors	Renter
Pre-requirements	A device capable to store a csv file
Rationale	At any moment, the user should be able to export his personal data to a common format
Demonstrates; Distinction	Data control
Scenario	The renter intends to export all the data collected during the stay. In this way, the renter accesses the system and requests for this action.

The user should be able to access the system to list all stays, and get all the information about all of them. The state of all personal data is controlled using a unique point of control.

At any moment and any place, the user should be able to access the system and see the state of the personal data. The state of the data indicates if the data were removed or not. Furthermore, the data subject should be able to get information about receipts of all stays and the consent information such as entities and privacy policies. In this way, the data subject can keep track of all entities that have been involved in the process of his data. This use case is described in Table 4.8.

Table 4.8: Use Case 8 - Access to unified control system to manage personal data

Actors	Renter
Pre-requirements	Always use the same email when checking-in in a new smart environment
Rationale	The user can use different smart environments such as different smart homes or another kind of smart environment.
Demonstrates; Distinction	Unified access control, management and visualisation of personal data
Scenario	During a year, the renter checked-in in several smart homes. At the end of the year, the renter accesses the system and can check for each stay, the state of his personal data, and the receipt.

Assuming that the system has integrated into a complex architecture with multiple entities involved in the data process, there are some pertinent considerations. In the beginning, all privacy policies are displayed to the user and the user can consent to them or not. The receipt is the proof of this consent. During the smart environment usage, some improvements can occur. From the user's privacy point of view, it can be problematic. These improvements can include the addition of a new component that will process personal data. Nevertheless, the data subject only consented to the previous entities to process his data. In this way, the data subject should be informed about it. The data subject should consent or not the usage of his data by the new component. Moreover, a receipt should be emitted as proof of this

consent. The system should manage all this information including the notification about the new service in the system as well as the consent request and the receipt.

Table 4.9: Use Case 9 - Privacy Policies of different entities that process data

Actors	Renter
Pre-requirements	Ability to access the system and the email.
Rationale	The renter should be informed about updates that can be involved with his personal data. In addition, consent should be requested whenever a new component processing personal data is added.
Demonstrates; Distinction	Transparency
Scenario	During the stay, the renter receives an email notifying him about the component added to the system that will process his personal data. Moreover, the renter is informed about the purpose of this new system integrated. In the same email, the consent to use the personal data in this new component is requested and its privacy policy provided. The renter decides to give the consent and signs the receipt.

In addition to the privacy policies associated with the entities, it should be also possible to list the identification of the entities involved in the data processing. We intend to give the user control and transparency about his personal data. The renter should know which entities are involved in his personal data process. This use case is described in Table 4.10.

Table 4.10: Use Case 10 - Control who can see/use renter's personal data

Actors	Renter
Pre-requirements	A device capable to access the system
Rationale	The renter should be informed about who can see and use his personal data.
Demonstrates; Distinction	Data control; Data flow
Scenario	The renter can access the system and check if other entities are accessing his personal data. The renter can contest this operation using the receipt.

4.3 REQUIREMENTS

Generally, most feature requirements are data transparency and data control. Data control is about the ability of the data subject to manage his data. Possible actions are to export his data to a file, acquire information about the data process at any moment, data collection consent, and request data deletion. In this way, the data subject should be able to request data deletion. As previously mentioned, the data subject's data should be removed when the data life period is over or when the data subject requests the data deletion and the Owner accepts this request. The other approach is to anonymize his data. Note that this life period should be equivalent to the duration while there is a purpose to maintain data. Independently of the process of data deletion, the data subject should be informed about it. The ability to notify is inserted in the data transparency notion.

Data transparency refers to how the information is presented to the data subject. It is about the ability to easily access and work with data no matter where they are located or what application created them. Moreover, data transparency is the assurance that data being reported are accurate and are coming from the official source. Therefore, the data subject should be informed about who, how, and when the data are processed.

Data control includes the user’s consent and the proof of this consent namely consent receipt. The user should be able to consent or not the data collection. In addition, consent is the first step to have data transparency. The consent should transparently contain all the information in a human-readable format (in this dissertation we considered plain text policies) and the privacy policies should be clear and understandable. Data transparency also includes all information about the smart environment such as the entities that are collecting data and the entities that are processing user’s data. The consent receipt should have all the information about the given consent, privacy policies accepted, user identification, and stay’s information. The process of personal data from the beginning until the end should only be available for the user providing a transparent data flow. Figure 4.1 depicts the most appropriate feature requirements described previously.

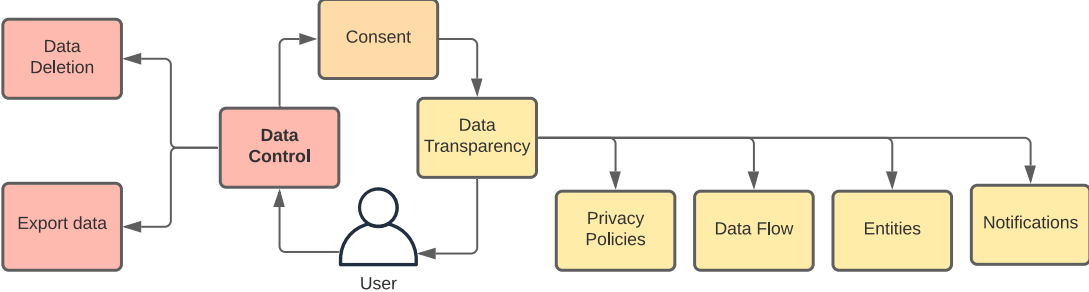


Figure 4.1: Feature requirements process

Before defining the architecture of the system, we need to define the list of requirements, features, and behaviors that the system needs to provide for correct execution. These requirements can be classified as functional or non-functional. Non-functional requirements describe how the system works, while functional requirements describe what the system should do. Thus, the functional requirements are described in Table 4.11 and the non-functional requirements are described in Table 4.12

Table 4.11: Functional Requirements

	Functional requirements
R1	The application needs to offer at least one authentication method. The data subject should be authenticated in the system to manage his data.
R2	Only the user should have access to all the information about him, including receipts, policies accepted, and data collected.
R3	The system should provide a way for the data subject to sign the receipt electronically.
R4	The system should ensure that the receipt is correct and validate the signature.
R5	The user should be able to see the receipts at any moment.
R6	The integrity of the receipt should be ensured.
R7	The system should provide a way to export a receipt in a usable format.
R8	The system should be able to associate a specific data set (either a database or other storing technology) with a user's stay.
R9	The data subject should have a way to verify if the devices that are collecting data are the same that were allowed by him. To increase transparency, the user can check which entities are available to control his personal data.
R10	The data subject should be able to request the deletion of his personal data.
R11	The data subject should be able to export his data to a file.
R12	The system should ensure that the data subject knows what happen with his data (transparency).
R13	The system should provide a list of all stays and a list of all the devices and respective policies.
R14	The data subject should have access to all his personal data collected.
R15	The Owner should be informed when the Renter requests for data deletion.
R16	The Owner should not be able to access the Renter's personal data.
R17	The data subject should be informed about his personal data flow. The data subject should be notified when his personal data were removed.
R18	The reference of the data should not be removed from the system. The data may be removed but the information about the stay should be in the system.
R19	At any moment, the data subject should be able to access the state of the personal data. The state corresponds to "Removed" or "Not Removed".
R20	While the personal data is not removed, the data subject should be able to export the data or just see them on the application.

Table 4.12: Non-functional Requirements

	Non-Functional requirements
R1	In terms of scalability, the system should be designed to handle high demands, from users and devices. It should also be capable of scaling vertically as needed.
R2	The system should not expose or share directly or indirectly with other entities, in any possible way, sensible information about the users. The user has a data flow control and a receipt as proof of granted consent. The owner cannot access the renter's personal data.
R3	The system should be prepared to be easier and fast to add more functionalities and features to it or make changes in the present functionalities. Moreover, it should be designed to be easy to reuse the different components in other applications.
R4	The renter should have a device capable of checking his email, interacting with the system and smart devices.
R5	The renter should be able to sign the receipt with digital signature.
R6	The house should have an interface to control the unique platform to control smart devices.
R7	Receipts and policies should be understandable by the user using a common language, accessible concepts, and only necessary content.
R8	The data collected during the stay should be securely stored . The personal data should be encrypted or stored in a disk encrypted to prevent malicious access to it.
R9	The data should be deleted after the time agreed upon during check-in. When the data are deleted, the renter should be notified.

4.4 SYSTEM ARCHITECTURE

This section describes the system architecture to the proposed solution following the main principles of a microservices' architecture. Thus, each component of the system is independently deployable and loosely coupled. This architecture also allows the system to scale vertically with ease. Finally, the microservice's architecture allows the system to interact with different technologies and different database implementations.

The high-level components that exist in the system are Figure 4.2: the component responsible for giving the data subject control over his data and complete transparency of data flow and entities in the process known as *Data Manager Service*; The component responsible for ensuring the privacy of the data subject during personal data collection known as *Privacy Manager Service* that may have a Web application to interact with the data subject and it is the point of contact when the data subject gives consent and signs the receipt; and the component responsible for receipt generation, and the storage of the receipt known as *Receipt Manager Service*. Moreover, the system requires an application to sign the receipt. Thus, the *Signature application* provides the ability to sign the receipt. In this way, the integrity of the receipt is ensured by the data subject's signature. To demonstrate the integration of this components in a smart home, it is required to have an *Unified Control Platform* which is the component where the data subject can control and update the state of the smart devices present in the smart space.

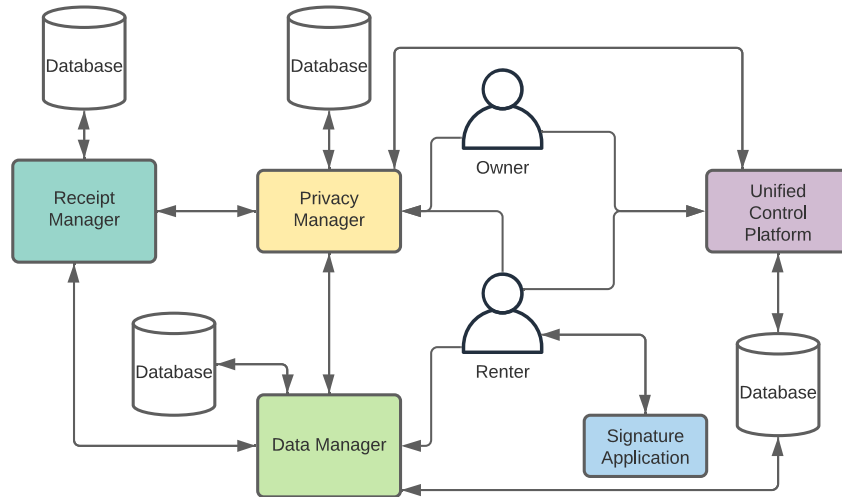


Figure 4.2: General architecture, the databases are placeholder for their implementation.

The first interaction point is the *Privacy Manager Service* where the user is able to create a new stay. However, to complete the stay creation it is required to sign a receipt as a proof of consent. Thus, the *Privacy Manager Service* requests the receipt generation to the *Receipt Manager Service* and this service returns a new receipt. Using the *Signature Application* the user can sign the receipt. The *Privacy Manager Service* send the stay information to the

Data Manager Service to be available to the user. Moreover, the *Privacy Manager Service* configures the *Unified Control Platform* that is available in the smart environment to control the smart devices. More details about the interaction between these components and the workflow are described in this section.

Interactions between all components are depicted in the Figure 4.3. In this way, the user can access the complete data flow since the beginning when consented to data collection until the end when the user can remove the data and revoke the consent. At the initial stage, the Renter interacts with the *Privacy Manager Service* to set up the stay in the smart environment. This service is responsible for requesting a receipt and provide it to the user. The user should sign the receipt and send it again to the *Privacy Manager Service* which forwards it to the *Receipt Manager Service* to store the signed receipt. In the context of this dissertation, the *Privacy Manager Service* is the component controlled by the Owner and it has no control over Renter's data. However, it is relevant to highlight that the *Data Manager Service* controls the renter's personal data and it makes part of the architecture to do the first interactions (create the stay), interacts with the *Receipt Manager Service* and provides the ability to the Owner decide to remove or anonymize the renter's personal data if the renter asks for it before the end of the data life period accorded during the consent. Moreover, this component is responsible for interacting with the *Signature Application* to request the Renter the receipt signature.

The security of the receipt should be ensured before the storage of the receipt. During the stay, the Renter controls the devices through the unified control platform. The unified control platform is the platform installed in the smart environment to control the smart devices such as check the state of the devices or to program the automations. At any moment, the Renter can use the *Data Manager Service* to control his data and to obtain all information transparently. At the end of the stay, the Renter can request data deletion using the *Data Manager Service*.

This architecture is prepared to integrate more components. In subsection 7.3.1 we propose an approach to deal with the delegation problem. In the future, a new component can be added to the architecture of the system to solve the delegation problem. The proposed solution is entirely compatible with the architecture proposed in this dissertation.

The following subsections describe specifically the components mentioned above. subsection 4.4.1 explains the features of the *Data Manager Service*; subsection 4.4.2 explains the features of the *Receipt Manager Service* including the receipt details; the *Privacy Manager Service* describes the functionality of this service into the presented architecture; and subsection 4.4.4 explains the *Signature Application* and the possible technologies to sign the receipt.

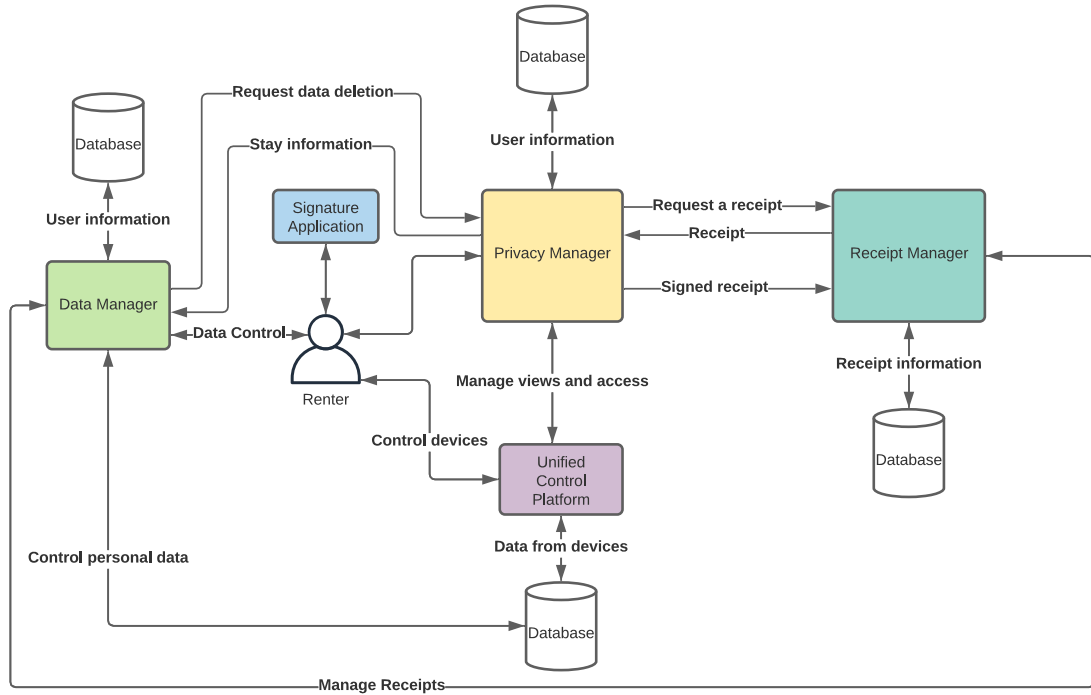


Figure 4.3: Interactions of the renter before or during the stay

4.4.1 Data Manager

Data Manager Service is the core of this dissertation. The idea of this service is to retain all information about all stays of the same user. In this way, the data subject can manage all personal data from different stays in smart environments including the consent receipts. The service should associate the user and his data securely and reliably to retain user's data from all stays in different smart environments. It is required that only the data subject can access to his personal data and consent information. Thus, this service works as a data control and it is the unique service in the architecture that should have direct access to the personal data collected.

Data Manager Service enables the data subject to manage his data transparently which includes information about which data is collected, where this data was collected, and by whom. The entities involved in the data process should only be the consented entities by the user before the smart environment usage.

The data subject can manage the devices that are collecting data during the stay. A suitable example of this control is about what happened in the CASSIOPEIA project where the user can select which devices are allowed to collect data. Independently of this selection, the data subject should be able to obtain information about the devices that are collecting his data. Thus, the *Data Manager Service* should provide a way to list the allowed devices. In addition, the user should be able to confirm which devices are collecting data and check if the admitted devices are the only devices collecting data. In addition, the consented entities

that are involved in data process should be made available to the data subject.

The described service should be able to export personal data to a file. In this way, the data subject can also confirm which data are collected. However, the most relevant point in this feature is the compliance of the right to portability mentioned in GDPR.

The notification system should be enabled to notify any update in the system, such as the addition of a new component or services in the system that are processing the user's personal data. Moreover, the data subject should be notified about any update to his requests. The most common request is the request for data deletion. *Data Manager Service* should provide a simple and intuitive way to request data deletion. The response to the request must be the anonymization process. In any case, a notification should be sent to the data subject referring the process to remove his data: removed permanently from the system or anonymized.

The consent information should be clear and understandable for any user. The consent receipt works as proof of the user's consent. The *Data Manager Service* should provide all information about the consent which includes the entities involved in the data process (system components and smart devices) and the privacy policies of each entity. More information about the consent receipt is provided in subsection 4.4.2.

4.4.2 Receipt Manager

The *Receipt Manager Service* should generate the receipt and store it ensuring the integrity of the receipt. Thus, the receipt should be signed before storage.

The receipt should contain relevant information about the given consent. In addition, the information present in the receipt should be clear and understandable for any user. Any user should understand the content of the receipt easily. In this way, the structure of the receipt should have the following fields:

- **User identification:** This identification should be the email of the user. The email should be provided during the check-in to the smart environment. Moreover, this field can be filled with any identification of the user. It can be the email or the name.
- **Devices:** the receipt should have the list of the accepted devices. In this way, before the consent, the user can check one more time the accepted devices. In addition, the privacy policy that corresponds to the device should be associated to it.
- **Entities involved in the personal data process:** All components that process the user's data should be identified in the receipt. In addition, the privacy policy that corresponds to the entity should be associated to it.

From the point of view of the user's privacy perspective, these fields are the most important to provide complete control and transparency to the user. According to the Kantara initiative¹ there are other important fields to add to the receipt. The following list presents these fields:

- **Version:** The version of this specification to which a receipt conforms.
- **Jurisdiction:** The jurisdiction(s) applicable to this transaction.
- **Consent Timestamp:** Date and time of the consent transaction.

¹<https://kantarainitiative.org/>

- **Collection Method:** A description of the method by which consent was obtained.
- **Consent Receipt ID:** A unique number for each Consent Receipt.
- **Language:** Language in which the consent was obtained.

Considering the specifications mentioned in the receipt according to the work requirements, the receipt should have the fields presented in Table 4.13.

Table 4.13: Receipt structure and possible option to fill the fields

Fields	Description and Guidance
Receipt version	CASSIOPEIA-202102
Receipt Timestamp	Unix timestamp milliseconds
Receipt ID	UUID v4
Language	English
Self-service point	Local where the receipt will be stored. For example, http://cassiopeia.id/receipts
Consent Status	"Consent given" "Consent rejected"
Legal Jurisdiction	"Europe"
Legal Justification	"Consent"
Method of Collection	"Online web action"
Receipt fingerprint	Digest of the whole receipt up to here (SHA256)
Digest of method of collection	Fingerprint (SHA256) of javascript + HTML of the page where the user clicks "I agree"
Organization	NGI CASSIOPEIA Demonstrator
User identification	Email of the user. For example, c.alexandracorreia@ua.pt
Devices	List of the accepted devices and respective privacy policy identification. For example, [temperature:2, humidity:4, motion:8]
Entities	List of the entities involved in the personal data process and respective privacy policy identification. For example, [data management:2, ML processing:1]
Other information	test123 .. extra information
Receipt Fingerprint	Fingerprint of the receipt

The present structure is similar to the receipt structure used in the CASSIOPEIA project. In the scope of this dissertation, we considered adding more fields to the receipt structure to increase user control and transparency to the data subject. Moreover, an appropriate difference for the receipt structure mentioned in the CASSIOPEIA project is the last field of the Table 4.13.

4.4.3 Privacy Manager

Privacy Manager Service is the service responsible for managing the user stay information from the Owner point of view.

It is the service that has an active part on the platform, the major use cases are triggered when the Owner creates a new record for a Renter's stay. The service enables the Owner to

manage the stay information, and consent receipts from the Renters. The service notifies the Owner whenever the Renter requests a data deletion and allows him to anonymize the data before deleting the raw version. It also provides the necessary methods to allow the *Data Manager Service* to download a copy of the personal data, or even request its deletion.

This service is for the Owner, what the *Data Manager Service* is for the Renter.

4.4.4 Signature Application

The receipt should be signed to guarantee its integrity. Additionally, if the user signs the receipt we can associate the identity of the user to the consent. It is important to validate the user's identity when consent is given. The *Signature Application* should provide the ability to sign the receipt. It is important to guarantee greater transparency, efficiency, security, improved traceability, and integrity.

Figure 4.4 depicts this signature process. The receipt is provided to the user and using the *Signature Application* the user should sign the receipt. The signed receipt is stored in the *Receipt Manager Service*.

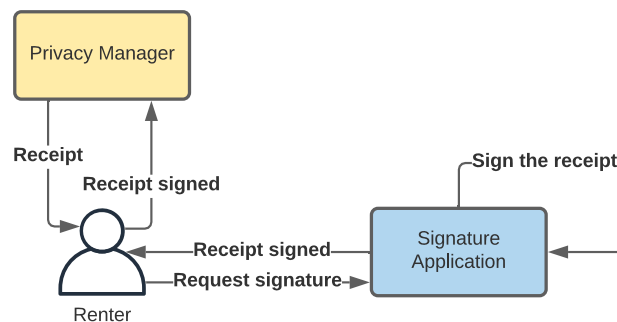


Figure 4.4: Signature process

The *Signature Application* requires a way to prove the user's identity. There are different options to assign the user's identity to the receipt. In terms of usability, we can consider the use of an embedded Subscriber Identification Module (eSIM) as a possible option to sign the receipt [8].

Up until recently, a physical SIM card was mandatory to make calls, send messages, and access the Internet using hardware modems, but a physical card has some limitations, especially when it is used within IoT devices deployed in a wide area. eSIM is an embedded alternative of the traditional physical SIM cards, providing the same usability, privacy, and security, but also minimizing some disadvantages of the traditional SIM card. Using an eSIM, the user could receive the receipt (from an application or email) and sign with a private key stored on the eSIM profile. The profile of the user can guarantee the validity of this signature independently of the mobile device. It is also possible to use a traditional SIM, but in this case, the user is dependent on the SIM card and not the profile that can run on other devices. During the initial definitions mentioned in this chapter, we consider using the eSIM to sign

the receipt. Consequently, we did a study about the impact of the eSIM in IoT scenarios and its advantages. The idea was to explore how the eSIM can be applied in the context of this dissertation. Moreover, the eSIM can be useful to integrate into IoT scenarios to be used in smart devices such as smart homes or eHealth scenarios. The mentioned study was accepted as a conference paper [8] that presents an overview of eSIM, discusses its main features, compares it to the physical SIM card, and additionally specifies the main characteristics of each vertical market. This paper also analyses the impact of eSIM in vertical markets. For the study present in this document, only the study about the usage of eSIM is directly applicable.

The traditional way of changing a mobile operator is by swapping SIM cards, which becomes problematic and expensive when we consider massive IoT deployments [85]. Every company operating in this field strongly avoids such operations, with the result of being tied to contracts with operators, losing flexibility, and frequently opting for synchronizing changes to product releases, entirely replacing old devices. To address this problem, the eSIM standard was developed by GSMA to fulfill the scalability, interoperability, and Over-the-Air (OTA) connectivity requirements.

SIM cards allow the connection of the devices to the cellular network by providing the required configuration and base authentication material. In practical terms, a SIM card is used to identify and authenticate a user within a Mobile Network Operator (MNO) to gain access to it. Therefore, it defines the relationship between the issuer (MNO) and consumers (user) and is part of a contract. SIM cards securely store the international mobile subscriber identity, which is the number that is used to identify the card identity into the network, and the authentication key. In addition, it can securely store data and applications [86].

eSIM is an SIM that is soldered directly into the device, and it allows the separation of the operators profile and the physical chipset. The trend is for it to be further merged with devices, becoming a section of silicon in the system on a chip or even functionality in an enclave. The device can have a default profile, and new profiles can be downloaded through a process called Remote SIM Provisioning, where the profiles contain the same data that normally would have been put in a regular SIM card [87].

One of the main focuses behind the eSIM development was to facilitate cellular M2M communication [88], and the management associated with the installation and renewal of contracts. However, it is also evident that the operation will not need to preallocate and manage a pool of standby SIM cards. In that sense, flexibility is greatly increased as operators may provision cards instantly and on-demand, without any physical collateral. From the perspective of new IoT deployments, several works are already exploring the concept, and eSIM solutions have been proposed in the IoT domain to facilitate secure M2M communication between devices [89]. One example [88] proposes an approach using eSIM for secured identification of devices. The idea is to distinguish the eSIM used for IoT services from those used for traditional services. Thus, MNO subscriptions associated with IoT services are managed more securely and efficiently.

Another approach to sign the receipt could be the use of the fingerprint sensor as a biometrical security mechanism to access the eSIM mobile key pair that is used to digitally

sign the file. A Public Key Infrastructure (PKI) provides the strongest known method of security. The authors of [90] consider that “Biometric sensor interoperability refers to the ability of a system to compensate for the variability introduced in the biometric data of an individual due to the deployment of different sensors”. Biometry provides us with a userfriendly method for the user identification and provides advantages for electronic transactions [91]. The authors of [92] propose a solution that involves the use of a biometric authentication mechanism. His proposed solution assumes that the fingerprint template would be captured on the phone and compared against a stored template on a database server. The fingerprint is captured and processed at run time, logins and passwords are also gathered at run time. The fingerprint template and authentication information is transmitted encrypted, using asymmetric keys. This approach ensures an end to end encryption of the communication between all parties. Additional fingerprint templates are never stored or compared in their raw state, but as hashes. Similarly fingerprints are not stored on the database in their raw form but as hashes. The fingerprint is never stored on the phone or server to be processed but is directly captured from the scanner, processed and hashed. In this way, it is important to highlight the most relevant disadvantage of this approach, which is a result of storing biometric data on a server, as it creates important disclosure risks. Also, users cannot replace fingerprints or templates after they are compromised [93].

Finally, another approach is to use the Portuguese Citizen Card (CC). This smartcard allows the citizens authentication and the digital signature of documents with legal value. It is a Portuguese digital national identification smartcard with strong authentication and digital signing capabilities. The CC is a unique identification card, in the sense that it substitutes most of the previous existing identification cards: Health System Card, Identity Card, Fiscal Identification Card, Social Security Card, and Electoral Identification Card. The card can be used both for physical and electronic recognition [94]. The CC is a versatile and secure card, with the latest in encryption and tamper resistance technologies [95].

4.5 DEMONSTRATOR WORKFLOW

The architecture defined in section 4.4 assumes the integration in a smart environment of the services developed in this dissertation. The services developed in this dissertation can be integrated into any smart environment. In this section we describe the workflow considering the integration of these services and the signature application into the CASSIOPEIA project which includes the collaboration with the *Privacy Manager Service* developed in the project. It is important to mention that this workflow describes the workflow of the services developed in this dissertation and not considering the features of services implemented in the scope of the CASSIOPEIA project. The services developed in the dissertation scope are improved versions of the services described in the project. In addition, the services proposed are represented using different colors: yellow boxes are about the *Privacy Manager Service*; green boxes are about *Data Manager Service*; marine boxes are about the *Receipt Manager Service*; and Blue boxes are about the *Signature Application*.

The initial workflow (Figure 4.5) results from the interaction between the services proposed named *Data Manager Service* and *Receipt Manager Service* as well as the *Signature Application*. This workflow is about the user register in the system and the consent for personal data collection. The data subject should be registered in the *Data Manager Service* to take advantage of the functionalities to increase privacy and data transparency using this service. Thus, the first needed action is to register the user in the *Data Manager Service*. Nevertheless, if the user is already in the system, then the instance should not be created and it should only be associated a new stay to this user. Before the user registration in the *Data Manager Service*, this service replies with a stay identification that should be provided to the user. It is important to mention that the user's identification processing should only occur when the user accepts the privacy policy associated to the *Data Manager Service*. The user's identification and the stay's information should be advanced to register the user in the *Data Manager Service*. The stay's information should include the check-in and check-out dates and the type of the smart environment. The user identification should be a unique identifier such as the user's email.

The consent for personal data collection should be finalized requesting a new receipt. Thus, the *Receipt Manager Service* should generate a new receipt containing the information about the data collection including the entities involved in the data process, their privacy policies and general legislation's information. After the receipt generation, the *Receipt Manager Service* sends it to the service responsible to provide the receipt to the data subject. The data subject should confirm the content of the receipt and he should sign the receipt through the *Signature Application*. The data subject can decide not to accept the consent and not sign the receipt neither use the smart environment. After this, the receipt signed should be stored taking into account some considerations to ensure the integrity of the receipts. It includes the signature of the receipt and the way how the receipt is stored (section 4.6). In addition, the receipt should be encrypted to increase its security. The receipt does not include direct personal data of the data subject but it includes information that can help to infer the identification to access the personal data. In this way, it is important to ensure the security of the receipt. The receipt's signature should be achieved using the *Signature Application* and the *Receipt Manager Service* should validate this signature. The Renter should request the receipt's signature to the *Signature Application* and this should reply with the signed receipt. The Renter should be able to upload or send the signed receipt to the entity that requests the signature of the receipt. *Data Manager Service* receives a reference to the receipt signed.

During the check-in phase in the smart environment, the interactions to request, generate and store the receipt occurs as follows (Figure 4.6). The receipt is requested to the *Receipt Manager Service* and after that, this service forwards the receipt to the service that requested it. The user should sign the receipt using the *Signature Application* and send the receipt again to the service that requested the signature. The signed receipt should be sent to the *Receipt Manager Service* that should store the receipt ensuring its security. After the storage a receipt reference should be sent, such as the receipt identification, to the *Data Manager Service*. During the Renter check-in, the unified control platform should be configured considering

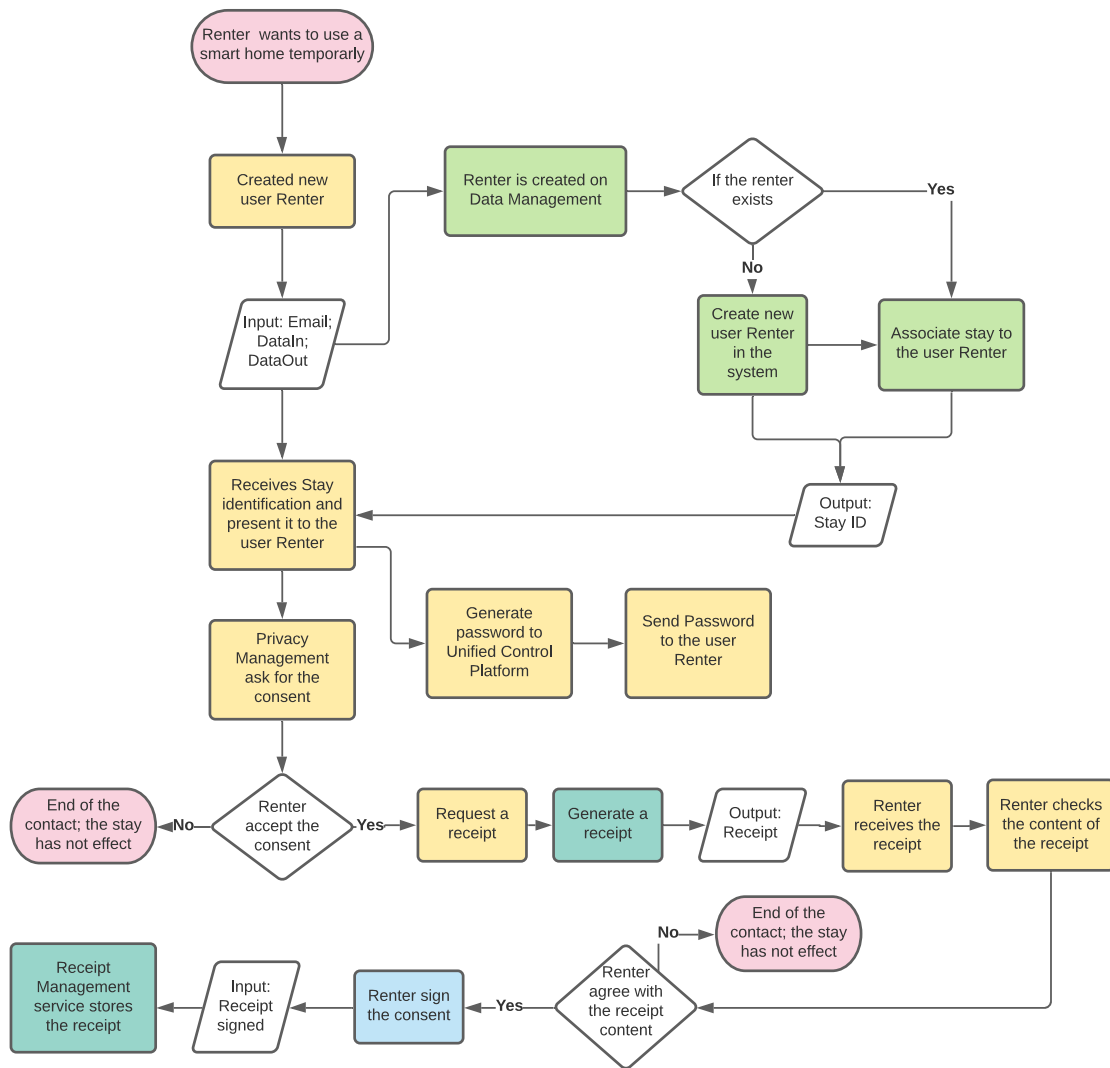


Figure 4.5: Initial phase in the demonstrator usage

the user preferences. In the CASSIOPEIA project this configuration is about the selection of the allowed smart devices by the Renter and its activation. To use the proposed services in this dissertation it is not mandatory to have this feature but it is required to provide the list of the smart devices that are being used during the stay. This list of devices that are collecting personal data will also be in the receipt. In addition, the unified control platform should be configured to show the Renter only the smart devices allowed. In the situation where this selection does not happen, then the Renter should be informed about all the smart devices, and they can only be used if the Renter accepts them. Additionally, a new password to access the unified control platform should be created to the new Renter and the unified control platform views, presented to the Owner and Renter should be different, with more restrictions to the Owner. The Owner should not have access to the Renter’s views neither the smart devices state that are being used by the Renter, promoting the Renter’s privacy.

Data Manager Service should be always available to the data subject and will include

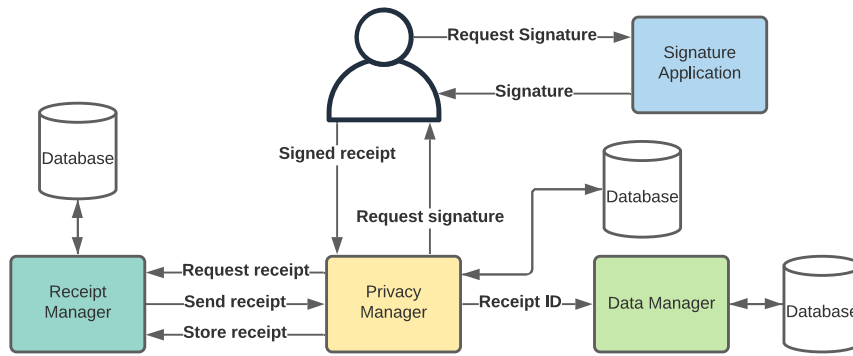


Figure 4.6: Receipt generation, signature and storage.

information about all stays registered in the system, which means that the personal data can be managed during the stay and after the stay. Through the *Data Manager Service* (Figure 4.7) the user can see the accepted policies, get receipt information, check devices entities, export data to a file, and control his personal data. If the Renter uses this service in different stays including different smart environments, all the information should be combined in this system and associated with the Renter. The services proposed are represented using different colors: purple boxes are related to the unified control platform; green boxes are related to *Data Manager Service*; and marine boxes are related to the *Receipt Manager Service*

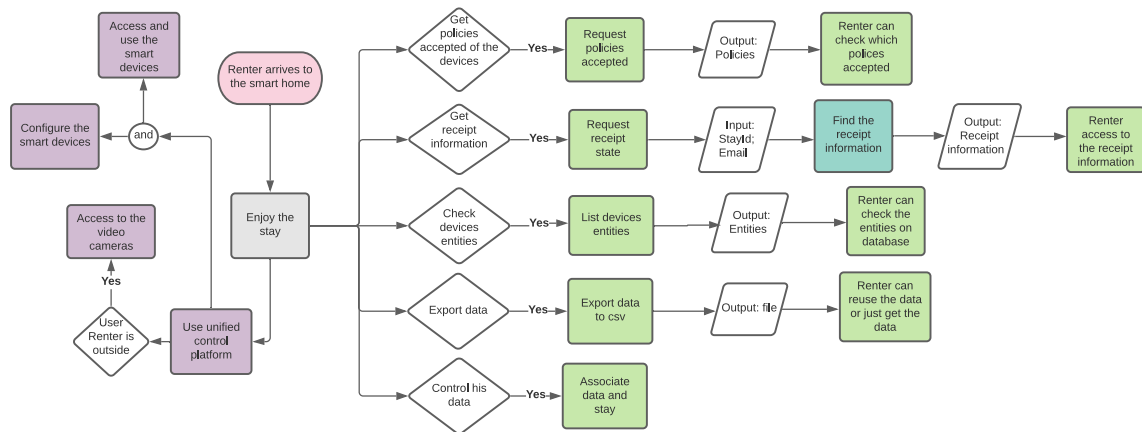


Figure 4.7: *Data Manager Service* interactions.

Although the personal data are associated with some retention policy, at the end of the stay, the *Data Manager Service* should provide a way for the user to request deletion of the data associated with a specific stay registered in the system. As mentioned, the personal data should only be kept in the system during the period accorded during the consent. However, the data subject should be able to request the data deletion sooner. In this case, the Owner can decide to remove or anonymize the personal data stored. Independently of the Owner

decision, the data subject should be notified about any action over his data and he should be able to access the state of his request through the *Data Manager Service*. After the data is deleted, the state associated with the receipt should be updated to a revoked state. In this way, the receipt should not be used to request data deletion again, and the lifecycle of data processing is over. Moreover, the state of user's data should also be updated to the removed state. Thus, the data subject should be able to access to the *Data Manager Service*, and check the state of the receipt and the state of his data. In addition, the data subject should be able to issue a request for his data, and the result should indicate that his data was removed from the system. It is an important feature because the request for his data should be done directly to the database that contains the data collected by the smart devices and if the data were removed then the response should be empty. In this way, the user can validate the information shown in the service interface that can be manipulated by other entities. By requesting the data directly to the source, the data subject can confirm what is presented in the service. This workflow is depicted in Figure 4.8. The services proposed are represented using different colors: Green boxes are related to *Data Manager Service*; and marine boxes are related to the *Receipt Manager Service*.

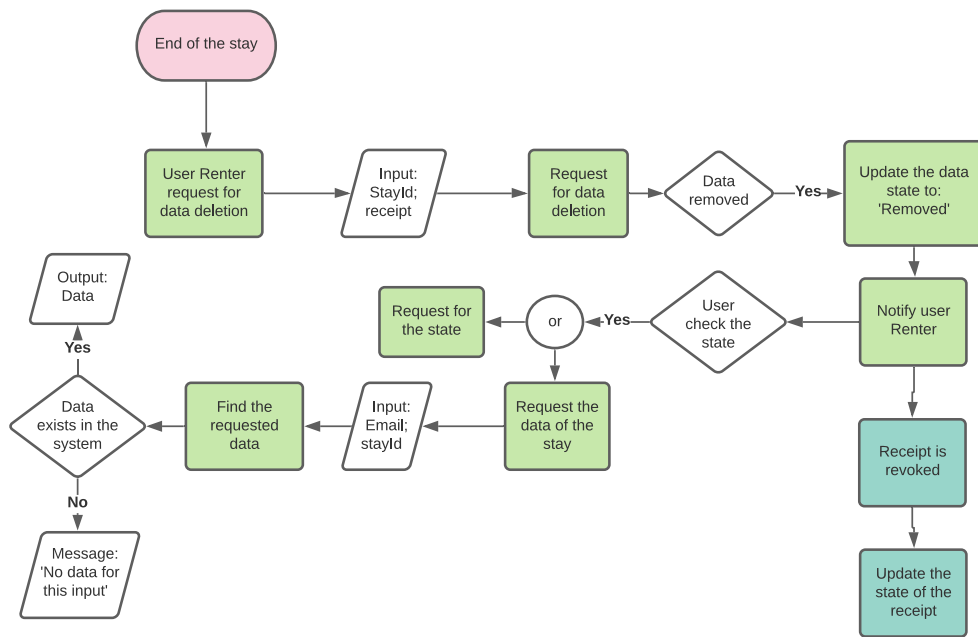


Figure 4.8: Data deletion request after the stay and through the *Data Manager Service*.

Typically, when the devices collect data, there is no notion of the identity of the user. The smart environment users may have an account in the unified control platform to use it but in general, this platform does not have information about whom the collected data belongs to. In addition, users registered in the unified control platform can use the smart environment directly. The authentication in the smart environment only allows the user to control the devices, and the data and the subject are not associated.

The entire smart devices present in a smart environment can build a complete user's profiles. Combining all user's information available, it is possible to infer the user identity. In this project, the idea is to have an association between users and data but with privacy guarantees. The data is managed by the user and not by third parties or any other system.

The *Data Manager Service* provides a unique point of control of personal data and, the controller of the data is the owner of this data without other additional entities.

4.6 PERSISTENCE

As it is depicted in the architecture diagram (Figure 4.2), the services proposed need to storing specific data persistently. Thus, the services have different requirements regarding the way to store and access this data. The *Data Manager Service* has to store stay's data including user identification and user consent. This information is static and well structured, the typical solution for this type of data is to use a relational database since it offers integrity, good performance, and a flexible querying system. However, this service needs to communicate with the database that stores the sensors data, gathered by the sensors in the smart house. The unified control platform should be configured to store data from sensors in a time series database [96]. Sensor data is better modelled as a sequence of values than as static data in a relational database. The *Receipt Manager Service* should store the receipt ensuring its integrity and security. There are two common ways to store this type of critical information: i) DHT and ii) blockchain. DHT is a distributed storage solution that uses a lookup service similar to a hash table. The items are stored in a corresponding node given by the hash of the key. The main advantages of this solution are the high level of scalability, good performance, and fault tolerance. A blockchain is a distributed ledger managed by a peer-to-peer network, where nodes collectively adhere to a protocol to communicate and validate new blocks. Blockchain records are secure by design since the network validates each record before adding them to the ledge. Although the blockchain appears to be the ideal storage solution for the *Receipt Manager Service* since the receipts contain critical information, the DHT is a better fit. However, there are some privacy concerns in Peer-to-Peer (P2P) systems. The authors of [97] propose privacy service (PriServ) which prevents privacy violation by prohibiting malicious data access. They focus on DHT due to its performance guarantees. In this dissertation, the receipts are secure by definition (signed by the users, and have internal integrity validations), the performance impact of the blockchain limits the usability of the storage solution in our scenario, on the other hand, the performance and fault tolerance of the DHT makes them ideal for our scenario.

4.7 PERMISSIONS

Assuming the presented architecture, we need to consider the permissions in three different parts of the data flow. The initial user's consent for data collection should be done and only the allowed entities by the Renter should process his data but the privacy policies should also be clear and understandable. The unified control platform should be configured to restrict

the content between the Owner and the Renter during the stay. In addition, the Renter should only have access to this platform during the stay. The *Data Manager Service* should be controlled only by the data subject and not depend of other components.

Privacy policies are the core of the permissions in the system as well the consent receipt as the proof of the consent of the mentioned policies. All components should be mentioned in the privacy policy and can only be used if the user consents to it. After the consent given, if the system requires a new component, it should be mandatory to request a new user's consent.

The access to the system should take into account the different user's roles. In the same smart environment, we can have the Renter, the Owner, and the system's administrator. The Owner should have complete control over the smart environment if there is not any stay occurring. In this case, the Owner should be able to control all devices. The Renter exists while there is a stay occurring. During the stay, the Owner's permissions should be restricted. Only the consented devices should collect data and only these devices should be controlled by the Renter. In addition, the Owner should only access the remaining devices (the devices are not allowed to this Renter because it is not collecting renter's data). The controller of the interfaces restrict what is displayed according to the user's role, considering an important topic in the privacy and data control contexts. Moreover, the management service should provide a way to check the validity of these views' restrictions.

The *Data Manager Service* should not depend on other components of the system. This service should only be controlled by the data subject and the information provided to the user should be reliable. In this way, this component should have permission to connect directly to other components to request user's information in an accurate way to minimize the possibility of third parties temper with the user's personal data.

4.8 CONCLUSION

In this chapter, we presented the use cases, scenarios, and requirements which frame the design of our set of solutions. We also describe a micro-service based architecture for a SmartBnb security enhancing solution, and the demonstrator workflow which includes the possible interactions with the system and the system behavior. Finally, this chapter described the main considerations regarding secure persistence, which will inherently result in a heterogeneous set of solutions, adequate to the different data types being persisted.

CHAPTER 5

Implementation

5.1 INTRODUCTION

With the architecture and the requirements defined, the next step is to implement the relevant component to validate the concept. This chapter addresses the in-depth description of the strategies followed to implement the key services: *Data Manager Service*, *Receipt Manager Service* and the *Signature Application*. Moreover, it explains the development of the *Privacy Manager Service* as the component controlled by the Owner. It also presents the strategies required to integrate the unified control platform in the demonstrator, and achieve a functional reference prototype.

This chapter is organized as follows: section 5.2 describes the architecture and the technologies used and the section 5.3 describes the implementation details considering this architecture; the details of Home Assistant integration are presented in section 5.4; and finally, the sequence diagrams are depicted and explained in section 5.5.

5.2 ARCHITECTURE

In terms of implementation, Figure 5.1 depicts the architecture considering the chosen technologies. The services developed are represented using different colors: yellow boxes are related to the *Privacy Manager Service*; green boxes are related to *Data Manager Service*; marine boxes are related to the *Receipt Manager Service*; and Blue boxes are related to the *Signature Application*

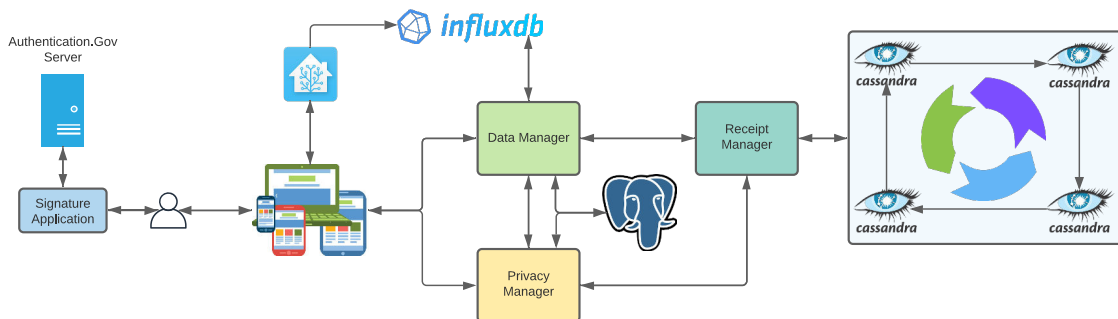


Figure 5.1: Architecture of the system

Home Assistant provides more advantages than the other open-source alternatives mentioned in the previous chapter. The documentation is very accessible and easy to understand, with an active community. However, the most relevant advantage is the large number of possible integrations that are available. In addition, Home Assistant provides a simple way to configure the interface, to integrate a new device and all other required configurations.

In this way, the PostgreSQL ¹ database was chosen as the relational database; the InfluxDB ² was chosen as the time series database; and the Cassandra ³ database was

¹<https://www.postgresql.org/>

²<https://www.influxdata.com/>

³<https://cassandra.apache.org/>

chosen as the DHT database. PostgreSQL, InfluxDB, and Cassandra were chosen since they are well-known databases, with good online documentation. Each one of them is typically recommended for their specific role.

Relational databases have been the universal standard for almost all the databases that existed. This kind of database is hugely useful to store tabular data [98]. Relational Database Management System (RDMS) must ensure four properties or characteristics in the transaction known as an Atomicity, Consistency, Isolation, Durability (ACID) [99]. PostgreSQL is a powerful, open-source object-relational database system. PostgreSQL has many features aimed to help developers build applications administrators to protect data integrity and build fault-tolerant environments, and help the management of the data no matter how big or small the dataset. In addition to being free and open-source, PostgreSQL is highly extensible. In the scope of this dissertation, the PostgreSQL database is used to store the user’s data such as email, name, and stay information such as check-in and check-out dates and the type of smart environment. Figure 5.2 depicts the data models established for the *Data Manager Service*.

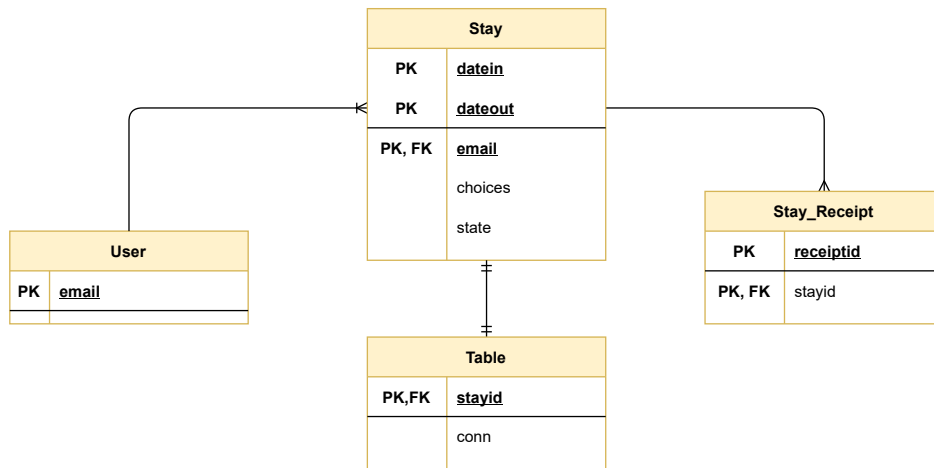


Figure 5.2: Relation database model of *Data Manager Service*

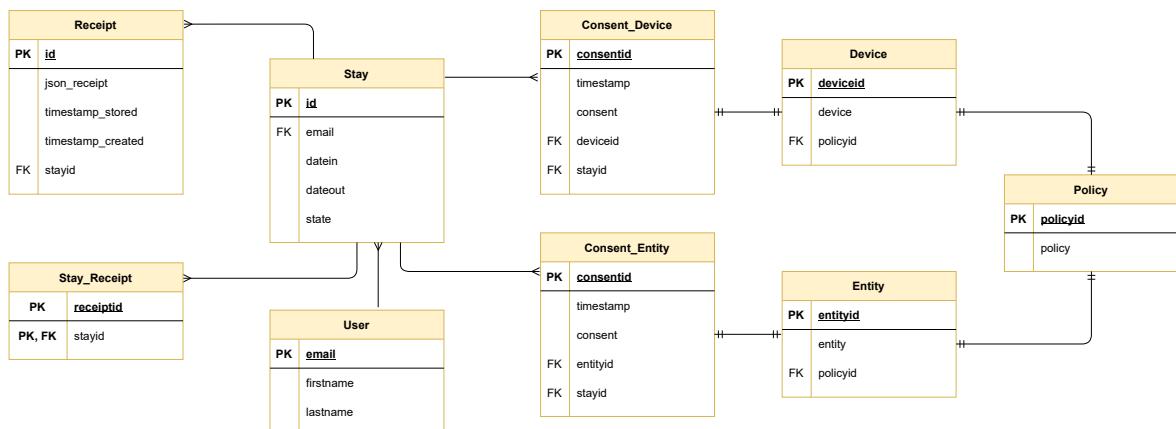


Figure 5.3: Relation database model of *Privacy Manager Service*

Sensor data is generated when a device detects and responds to some type of input from

the physical environment. Sensors can generate mass quantities of sensor data that may or may not be immediately valuable for decision-makers. The data points captured at a specific moment in time transform sensor data into time series data. There is a large volume of sensor data generated at each minute, making storing these unique points an increasingly difficult task. Traditionally, some of these sensor data points would be stored only for a limited period of time. With cloud-based storage solutions, it becomes more feasible that captured data points are stored over longer periods to allow for time-based analysis. The basis behind time series consists of repeated measurements of the parameters over time. Most times, the intervals are regular, but this is not a requirement. The use of measurements is employed when the data is not updated but rather accumulated over time, with new data being added for each measured item at a time point. The main purpose of a time series database is to store or log the sensor or other data over periods. A time-series database must handle a large number of database entries and should provide ways for filtering and analyzing data. The technologies for collecting and storing large-scale time series are used by more and more companies and people. In recent years, the volume of time series data has greatly expanded and the tools for handling time series data, at this scale, are more demanded [100]. InfluxDB provides a comprehensive platform to collect, store, analyze, and visualize data. To ensure the interaction with the data, InfluxDB offers an SQL-like query language named InfluxQL. Since the time-series databases can receive a large amount of data each second, InfluxDB automatically compacts the data to minimize the storage space.

Traditionally, the relational model and centralized architectures have been used mostly. However, with the growth of the Internet in recent decades, both in the number of users and in the amount of information, the use of decentralized architectures and alternative database models to the relational model has been extended, which receives the name of Not Only Structured Query Language (NoSQL) databases [101]. NoSQL databases are distributed, non-relational databases designed for large-scale data storage and parallel data processing across a large number of commodity servers. A DHT is a decentralized storage system that provides lookup and storage schemes similar to a hash table, storing key-value pairs. Each node in a DHT is responsible for keys along with the mapped values. Any node can efficiently retrieve the value associated with a given key. (key, value) pairs are stored in a DHT, and any participating node can efficiently retrieve the value associated with a given key. Just like in hash tables, values mapped against keys in a DHT can be any arbitrary form of data (Figure 5.4). The nodes in a DHT are connected through an overlay network in which neighboring nodes are connected (Figure 5.5). This network allows the nodes to find any given key in the keyspace.

A DHT works in a decentralized and autonomous way where the nodes form the system without any central authority. It is also fault tolerant and scalable. DHTs are initially conceived for efficient data lookup in large-scale wired networks. The main objective of this combination is to manage location-independent data and node identification. DHT mapping over WSN brings, however new challenges [102].

There are a few problems with this approach. Any sufficiently large distributed system

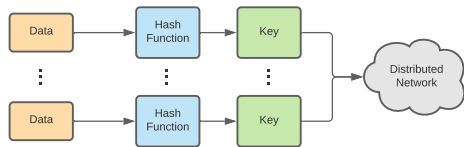


Figure 5.4: DHT work

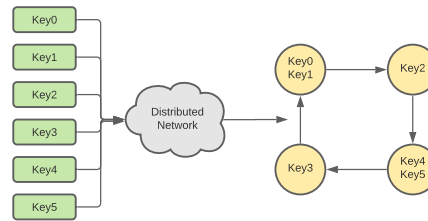


Figure 5.5: DHT Overlay Network

with a central coordinating node will experience bottlenecks at that node. A large amount of strain in terms of processing power and bandwidth can arise at the central node which can make the entire system appear unavailable. DHTs offer a scalable alternative to the central server lookup, which distributes lookup and storage over several peers with no central coordination required. Any read or written operation on a DHT must locate the node containing information about the key. This is done through a system of key-based routing. Each node participating in a DHT contains a range of keys stored along with information about the range of keys available at other nodes in the system. Any node contacted for information regarding a key forwards that request to the nearest node according to its lookup table. The more information each node maintains about its neighbors, the fewer hops are required to get to the correct node. While maintaining more state at each node means lower latency lookups due to a reduced number of hops, it also means nodes must exchange a greater amount of information about one another. The tradeoff between lookup latency and internal gossip between nodes is a fundamental driver behind DHT design. The model of the DHT is depicted in Figure 5.6.

Receipt	
PK	<u>email text NOT NULL</u>
PK	<u>id_receipt uuid NOT NULL</u>
	timestamp_now date
	json_receipt text
	state text

Figure 5.6: DHT model

The *Receipt Manager Service* is responsible for the generation and storage of the signed receipts. We need to ensure their security and their integrity. In this way, we select a DHT structure to store the receipts. To implement a DHT we must select the most appropriate database that supports it. Cassandra is an open-source distributed database management system designed to handle large amounts of data spread out across many commodity servers while providing a highly available service with no single point of failure. Cassandra is a NoSQL database that stores data in non-related tabular forms. NoSQL databases, similar to the Cassandra database, can handle huge amounts of data with easier management and lower cost

compared to SQL databases like Oracle and other relational databases [103]. Cassandra offers more flexibility regarding fault handling and managing a wide range of data. It can be done without compromising the performance of the system. Moreover, Cassandra can maintain availability and scalability [104]. Cassandra runs each query on the entire table because it has neither memory to remember the result of a previous query, nor supports VIEW or JOIN on tables (or keyspaces) [105]. In this dissertation, the Cassandra database was chosen to store the receipts in a DHT structure.

The DHT makes possible to have a receipt storage in a scalable way which can have a huge number of receipts and accepts different formats of receipts and can serve different services with different applications.

The three main services (Privacy, Data and Receipt Management) are developed as a Representational State Transfer (REST) API implemented in Python using the Django framework. Django is a high-level Python Web framework that encourages rapid development and clean, pragmatic design. The *Signature Application* was developed using the Flask ⁴ framework. This simple service works as a small plugin for the signature operations, for this reason we selected a lightweight microframework. Flask is a microframework web written in the python programming language. It is designed to make web applications quickly and easily, with the ability to improve complex applications [106]. Both the *Privacy Manager Service* and *Data Manager Service* have web graphical interfaces that allow the Owner/Renter to interact with the underlying services.

Before discussing the implementation detail, it is important to mention some points. The *Privacy Manager Service* is primarily a local service that is deployed in the Owner House. The *Data Manager Service* is a remote service that allows the Renter to have control over his personal data independently from his physical location. As such this service needs to be scalable, which is the reason for the lightweight data model depicted in Figure 5.2. However, this scalability is limited by the database. There are several ways to deal with this. The first one is to use a scalable storage solution, similar to the DHT of the *Receipt Manager Service*. We could even deploy a replica of the service per node of the DHT. The second approach would be developing the service to be stateless, meaning that all the necessary information would be inserted within the receipt. In this solution, there is no database, thus when the data is removed, then the receipt is revoked and it is created a new receipt where the state of the data is “Removed”. This is a disadvantage as the receipt will contain mutable information. This leads to the issue of developing a revoking and reissuing receipts on the fly. For this prototype we used a lightweight data model, that was supported by PostgreSQL. This database was selected since it is easy to develop and debug. However, in a real-world deployment, a DHT could be used to improve the scalability of the service.

Finally, it is important to mention that a fourth service could have been developed. This service could be used to transfer the personal data from the Owner’s smart environment database to the previously mentioned centralized service. This service would free resources on the Owner’s Smart Environment deployment, and allowed for greater control over the

⁴<https://flask.palletsprojects.com/en/2.0.x/>

personal data. However, the implementation of this type of service may require uniforming different types of data representation, into a single storage solution [107]. These issues are not within the scope of this work.

5.3 IMPLEMENTATION DETAILS

The Django views file has the methods (POST or GET) to implement the required functions. All of these methods are using the Cross Site Request Forgery (CSRF) protection (`@csrf_exempt`) that provides easy-to-use protection against CSRF attacks. This type of attack occurs when a malicious website contains a link, a form button, or some JavaScript that is intended to perform some action on your website, using the credentials of a logged-in user who visits the malicious site in their browser. CSRF attacks occur when a malicious website causes a users web browser to perform an unwanted action on a trusted site [108].

To instantiate the databases it was used the docker-compose tool. Docker is a tool designed to make it easier to create, deploy, and run applications by using containers. Containers allow to package up an application with all parts it needs, such as libraries and other dependencies, and deploy it as one package. With the container, it is possible to run on any other Linux machine regardless of any customized settings that the machine might have that could differ from the machine used for writing and testing the code. In a way, Docker is a bit like a virtual machine. But unlike a virtual machine, rather than creating a whole virtual operating system, Docker allows applications to use the same Linux kernel as the system that they are running on and it only requires applications to be shipped with things not already running on the host computer. This gives a significant performance boost and reduces the size of the application. Compose is a tool for defining and running multi-container Docker applications. To use these databases into the demonstrator it was required to integrate them with the Django framework.

In the data management Application Programming Interface (API) it was created an `InfluxDBClient` instance to communicate with the InfluxDB database. It was required because this system is responsible for returning the data collected during the stay when the user requests it. In addition, it is the service responsible for data deletion.

To use the Cassandra database in Django it was required to use the Django Cassandra Engine as the Cassandra backend for Django. To use PostgreSQL in Django is simpler than the previous configurations mentioned in the Cassandra database. We only need to configure the database in the `DATABASES` of the settings Django file with the right parameters. It is only required to fill the `databases` field with the correct identification of the database (IP address, port, database name, and credentials). Then, we can create the classes in the models and use them in the views Django file.

The authentication process is essential for controlling the access to various resources and facilities [109]. Authentication is the mechanism of associating an incoming request with a set of identifying credentials, such as the user the request came from, or the token that it was signed with. In the developed demonstrator, we can find three different kinds of authentication: user authentication in the demonstrator interface, Home Assistant authentication depending on the user's role, and service authentication.

The different components of the demonstrator are not running on the same server and the *Signature Application* runs on the client-side. Consequently, to have a secure communication between all the components it is required to have the API authentication. Figure 5.7 shows the service’s authentication in the proposed architecture.



Figure 5.7: Access through Token Authentication

To solve this problem, it is required to provide token authentication before any operation. All requests are allowed if the token was provided. This authentication scheme uses a simple token-based HTTP Authentication scheme. Token authentication is appropriate for client-server setups, such as native desktop and mobile clients.

5.3.1 Privacy Manager

The *Privacy Manager Service* has the interface to create a new stay in the system. The stay is associated to a new Renter. This Renter provides his identification, in this case the email, and provides information about the dates of the stay, check-in and check-out dates. After the stay created, the Renter can select which devices he consents to be enabled during the usage of the smart environment. In addition, he can consent which entities can be also enabled to process his data during the stay.

The final step of the stay creation is the receipt signature which works as a proof of consent. The system requests a new receipt, provides the receipt to the Renter and the Renter can sign it. After sending the signed receipt to the *Receipt Manager Service*, this service validates the signature using the certificate of the Portuguese Citizen Card.

Privacy Manager Service is controlled by the Owner and is the component responsible for the initial setup and for communicating with the remaining components in order to establish the initial privacy parameters. All the code of this service is available in a public repository in GitHub ⁵.

5.3.2 Data Manager

The focus of this work is to increase the privacy provided to the user responding to all privacy requirements based on regulation. *Data Manager Service* provides user control over his data and it is the point of data control. This service is available at any moment for the user to make

⁵<https://github.com/catarinaacsilva/cassiopeia>

possible permanent data management. Thus, the *Data Manager Service* provides features to increase data transparency, ensure portability and provides the possibility to confirm which data is collected, used by whom, and when.

The developed API is available in a public repository in GitHub ⁶. This API can be integrated with any other service in the privacy control in all smart environment types since the parameters to communicate are respected. This API provides the abilities mentioned above and responses to the requirements established in the previous chapter.

This service provides the required methods so the Renter can control his data ensuring that the Owner cannot access to the data and in this way; the Owner cannot change the information displayed to the Renter. In addition, this service accesses directly from the original data source avoiding the possibility to present wrong information to the Renter. In this way, this service provides the methods to store a new stay for a new user or a new stay for an existing user in the system. Moreover, the Renter can export his personal data to a CSV file, regarded by the smart devices and access to all the consent information which means that he can access the stay's receipt and the content of the details of the consent receipt including the accepted devices, accepted entities and accepted policies as well the remain information present in the receipt. At the end of the stay, the Renter continues to be able to access the service and can request for data deletion. The Owner can decide if the data is removed or anonymized. In both cases, the Renter is notified and if the Owner decides not to remove or anonymize his data then the Renter is notified about data deletion when the data life period ends.

5.3.3 Receipt Manager

As mentioned, the *Receipt Manager Service* has the ability to generate and store the signed receipt. The receipt works as proof of the consent given. The receipt contains information that can be useful to infer sensitive data about the Renter. Thus, there were some proper considerations to take into account during the implementation of this service. These considerations were:

- The receipt should be sent to the service that requested it. The receipt should only be stored after the signature process;
- The identity of the data subject should be validated to ensure its integrity;
- The receipt should be signed and stored in a distributed way using a DHT structure.

The receipt is a JSON file following the structure previously mentioned in Table 4.13. The service that requests the receipt generation should provide the information to fill the fields in the receipt structure. According to the proposed architecture, the *Privacy Manager Service* should invoke the method to generate a new receipt (GET request). Accordingly, some parameters have to be supplied. The fields about the version, organization, policies, devices, and entities should be supplied when the function is invoked. The fields about language and jurisdiction have default values to be used in case none are supplied when the function is

⁶<https://github.com/catarinaacsilva/cassiopeia-data-retention>

invoked. The consent field should also be supplied but it only has two possible values: consent or not consent. The timestamp field is filled with the current time of the receipt generation using the `DateTime` library. The receipt ID field is also calculated during the generation of the receipt using the Universally Unique Identifier (UUID) library. The self-service point field is filled with whatever value is passed as a variable in the code. That is, this value must always be the same.

Usually, the privacy policies describe what information the system collects, why this information is collected, and how the user can update, manage, export, and delete his information. The user should provide consent to data collection after understanding the content of the system privacy policy. Therefore, it is crucial to ensure the integrity of these policies. Accepting these policies means that the user consents the data collection in the way mentioned in the privacy policy. The content of these privacy policy must always remains the same as what the user accepted and not to be changed. In the implementation of the receipt generator, the policies must be associated to the devices or entities which means that the receipt has a field `device` with a dictionary where the key is the name of the device and the value is the policy; and a field `entity` with a dictionary where the key is the name of the entity and the value is the policy. In this way, we can have the different entities and different devices with different policies and easily relate the policy to the entity or device.

The Hash (SHA256) library was used to fingerprint the receipt. This fingerprint is stored in the `receiptFingerprint` field of the receipt. This is also important to ensure that the receipt is not changed after the consent. From a system point of view, it is also useful to have a way to prove that the renter has given permissions and the purpose of those permissions. The Renter associates his identity to the receipt signing the receipt with the *Signature Application*.

The *Data Manager Service* only stores a reference for the receipt using its identification. The *Receipt Manager Service* stores the receipt and responds to all the requests by other services involving the receipts. Moreover, this service can verify the signature validity. The signature is verified by the *Privacy Manager Service* before its storage in the *Receipt Manager Service* but at any moment it is possible to verify its integrity using the signature validator in the *Receipt Manager Service*.

The code of this service is available in a public repository in GitHub ⁷.

5.3.4 Signature Application

The receipt as proof of consent is only truly valid if the data subject signs it. In this way, we can associate an entity to the receipt, making possible the validation of the user's identity. The signature of the receipt is possible with the *Signature Application* developed using the publically available libraries of the Portuguese Citizen Card.

The data subject signs the receipt using the Portuguese Citizen Card known as “cartão de cidadão”. According to the documentation provided by the Portuguese governance, it was possible to develop an application to sign the receipt and verify the signature validity. The *Receipt Manager Service* uses the developed method to confirm the validity of the

⁷<https://github.com/catarinaacsilva/receipt-generator>

user’s signature. Thus, the *Receipt Manager Service* stores the receipt ensuring its integrity (fingerprint and user’s signature).

The *Signature Application* code is available on Github Repository ⁸. The application uses the *cryptography* library and *PyKCS11* as the most relevant libraries in this context. The certificates are loaded to validate the signature.

In a real-world deployment, the services could use the authentication services provided by AMA (“Agencia para a Modernização Administrativa”)⁹. This plugin was developed only for the prototype, as we only require the signature capabilities of the Card, and using a simple plugin eases the development and debugging of the services.

5.4 HOME ASSISTANT INTEGRATION

The integration of the smart devices in the Home Assistant requires some configurations as well as the integration of the Home Assistant in a smart environment architecture. However, there were some challenges during the integration of the Home Assistant in the system assuming our architecture. The Home Assistant is programmed to be installed and be used in any smart home but to correspond to our requirements some adaptations were needed.

As previously mentioned in this document, it is important to guarantee privacy between the Owner and the Renter. The system should ensure that the Owner cannot access the information produced by the Renter during the stay. Home Assistant does not provide any method to restrict the views depending on the user role. It was created a new group in *homeassistant/.storage/auth* file to solve the mentioned problem. This group is assigned to the Renter account and it only contains the entities’ names of the consented smart devices by the Renter.

The proposed approach to solve the problem is not absolute because the selection of the devices should be done manually in a configuration file of the Home Assistant which implies restarting the Home Assistant. However, it is a unique option to have different roles in the platform. According to the permissions mentioned in the previous chapter, we have three different roles. The administrator can see all the devices and update the configurations; during the renter’s stays, the Owner can only access the disabled devices; after the renter’s stay, the Owner can access all available devices in the smart home; during the renter’s stay the Renter has access to the consented smart devices; after the renter’s stay, the Renter cannot access the Home Assistant. The content for each role was described in the *homeassistant/.storage/auth* file. The interface for the Owner only contains the smart devices not accepted by the Renter. Thus, the group assigned to the Owner only contains these smart devices. On the contrary, the groups assigned to the Renter have the remaining devices that he accepted during the smart environment check-in. Both renter and owner cannot change the configurations of the Home Assistant (the button for this effect was removed from their interfaces).

Due to the limitation mentioned above, the renter is always identified by the Renter in Home Assistant. However, it is mandatory to change the password to different renters. When

⁸<https://github.com/catarinaacsilva/smartcard-application>

⁹<https://www.autenticacao.gov.pt/>

the renter leaves the house, it is required to change the password to guarantee that he cannot access the system. When a new renter arrives at the house, the new password should be given to the renter.

Home Assistant does not provide any method to change the password dynamically. To solve this problem a script was implemented to facilitate this task. Moreover, it was required to create two Home Assistant instances. The restricted Home Assistant is used for the renter and the home instance of Home assistant is used for the administrator and the Owner. The script assumes that the account present in the *secrets.txt* file is the unique account valid to enter in the restricted Home Assistant. For the Owner, the password is static.

This is possible because Renter and Owner are only known by these names in the Home Assistant platform. For the *Data Manager Service*, the users are identified by another key, in this case, the email. It is also possible because the instances of the services are deployed locally. At the same time, there can be different users Renter in different smart environments from the Home Assistant perspective as this works locally. For the *Data Manager Service*, there are different users identified by email in different smart environments. The dates of the stay (check-in and check-out) can be the same but the email makes it possible to identify the user.

Home Assistant uses the database to store events and parameters for history and tracking. The default database used is SQLite. However, other databases can be used. As previously referred, the selected smart devices by the Renter collect data during the stay in a particular format. Thus, the Home Assistant was configured to store the data in InfluxDB.

To test the demonstrator, added sensors were added to the Home Assistant in different phases. At an initial phase, sensors developing mock sensors were simulated : CO₂, light, smart lock, motion, smoke, temperature, and humidity. The code for these simulated sensors was developed in Python using the *paho-MQTT* library. The data for the simulated sensors is from different datasets. To deploy the sensors it is used a docker-compose file to make the deployment easier. In addition, it was developed a lock control. To simulate a real environment real sensors were added.

At the initial stage of the dissertation development, an experiment was done to analyze the impact of the security mechanisms in a WSN that contains smaller smart devices with restricts capacities. A WSN has relevant applications in smart environments such as monitoring and target tracking. This has been enabled by the availability of smart devices that are smaller, cheaper, and intelligent [110]. A Wireless Sensor Network (WSN) is a network of nodes that works in a cooperative way to sense and control the environment surrounding them. These nodes are linked via wireless media. Nodes use this connection to communicate with each other. A typical WSN architecture consists of the following three components: sensor nodes, gateway, and observer (user) [111]. WSNs are a collection of nodes, and these nodes are individual small computers. These tiny devices work cooperatively to form centralized network systems. WSNs have been used for various applications such as smart homes, agriculture, nuclear reactor control, security, and medical applications. In terms of design and requirements, it is necessary to guarantee some aspects. Reliability is the ability of a sensor to maintain its

network functionality without any interruption, and it is essential in specific kinds of devices and scenarios. The nodes' density affects the coverage area, reliability, and accuracy, adding a considerable load to the network. Network latency, capacity, and robustness are all affected by the sensor nodes' network topology and density. Furthermore, the complexity of routing data depends on network topology.

Sensor nodes are mainly battery-powered. Therefore, each node's lifetime depends on the battery's lifetime and the power consumption of the work performed in that device. The security aspects of WSNs are focused on the centralized communication approach. Therefore, there is a need to develop a distributed security approach for WSNs. WSNs need to be self-organized. Since sensor node failures can occur in any network, new sensor nodes may join the network to decrease these failures' harmful effects. In many applications of WSNs, the mobility of sensor nodes or the base station is essential, which means that sensor nodes are wandering around. This mobility raises many issues, from routing stability to energy and bandwidth consumption. For some applications, data delivery and latency have to be bounded, especially in real-time operations and monitoring. Sensing data after the desired bound is usually useless. We need to guarantee the connectivity of the sensor nodes with the Internet. Network connectivity is defined by a permanent connection between two different sensor nodes. These nodes are densely deployed in a sensor network.

Different smart devices have different characteristics including power consumption energy. To study which devices are better for our smart home simulation we study the power consumption. In addition, our experimental evaluation and main contribution explore the impact of security mechanisms on top of conventional WSN use-cases. We do so by measuring the three fundamental metrics (power consumption, message delay, and additional bytes) using state-of-the-art communications protocols (HTTP and MQTT) and representative embedded devices (the ESP8622, ESP32, RPi1, RPi2, and RPi3). It is important to mention that smart home security is not the focus of this dissertation. However, data protection also includes the security of the users. Thus, the security requirements ensured in this work are the basic to guarantee the *privacy by default* for this kind of architecture, and the encryption algorithms are included in these assumptions. In the process it helps to select smart devices and to study the impact of cryptographic algorithms in smart devices with limited capacities. The experiment was designed as depicted in Figure 5.8:

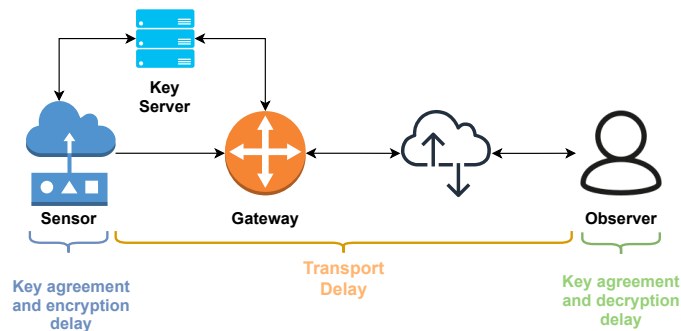


Figure 5.8: Delays measured on the experiments.

Figure 5.8 depicts the architecture of the experiment and the delays measured. The gateway is responsible for the communication between the observer and the system and the key server is responsible for sharing the public keys of the session. The idea of the experiment is to measure the delays: key agreement, transport delay, and encryption/decryption delay. The selected smart devices are described in Table 5.1.

Table 5.1: Devices used in the project

Device	Description	Brand
Smart light	Control the lighting at home	Shelly
Gas sensor	A smart gas sensor that can detect Natural Gas or Liquefied petroleum gas	Shelly
Smart plug	Control a wide range of devices and appliances that are connected to the smart plug	Shelly
Aqara Motion Sensor	Detect nearby motion	Xiaomi
Aqara Water Leak Sensor	Indicate water levels in areas	Xiaomi
Aqara Temperature and Humidity Sensor	Measure temperature, humidity, and pressure	Xiaomi
Aqara Door and Window Sensor	Detects open and close doors	Xiaomi
Co2 sensor	Measure Co2 gas	Virtual
Lock	Lock and unlock doors	Virtual

A video camera was placed in Instituto de Telecomunicações (IT) to simulate a kitchen. This camera records the IT kitchen for 36 hours. The FFmpeg¹⁰ integration allows other Home Assistant integrations to process video and audio streams. FFmpeg is a command-line tool to convert multimedia between different formats.

To manage which buttons are shown to the user in the Home Assistant sidebar, it was used an open-source project¹¹ to control the buttons.

5.5 SEQUENCE DIAGRAMS

The initial point of the data flow occurs when a new user rents the smart environment, in this study case, the smart home. In the proposed architecture, the *Privacy Manager Service* is responsible for providing the methods required for this action. Considering the prototype, it is necessary to have the Renter information, in this case the renter’s email, and the stay dates information (check-in and check-out dates). This stay information is stored in the databases of the *Privacy Manager Service* and it is sent to the *Data Manager Service*. This service also requests for the Renter consent, generating a receipt and sending it to the Renter. Using the *Signature Application* the Renter can sign the receipt with the Portuguese Citizen Card and the receipt is sent to the *Privacy Manager Service* which validates the signature. After completing this initial setup, the Renter can access the *Data Manager Service* and access the stay data, export personal data and check the information about the consent provided. In addition, through this service, the Renter can request for data deletion. This request is

¹⁰<https://ffmpeg.org/>

¹¹<https://github.com/Villhellm/custom-sidebar>

forwarded to the *Privacy Manager Service* to notify the Owner about the request from Renter. The Owner can decide to remove Renter’s personal data or anonymize this data. In both cases, the Renter is notified about what happened to his data and at any moment the Renter can check the state of his data. These interactions between the user and the services and between services are explained in this section including the messages exchanged between them.

The user’s Renter personal data can only persist in the system during the validity of the purpose. Thus, this life period should be referred in the consent request. For the implementation, we chose two years by default as the life period of the data in the system. When these two years are over, then the data should be removed from the system. In this way, the *Data Manager API* provides the required method to implement this functionality. However, the API provides additional methods to provide more data control and transparency to the user. At any moment, the data subject can access the state of his data and check if the data were removed or not. Moreover, the data subject can request the data deletion before the end of the data life period. In addition, this API provides a way for the data subject request the data deletion.

At the end of the data life period, the *Data Manager API* removes the data without any other inquiries to any user. However, the Renter can decide requesting the data deletion soon. In this way, the Renter requests the data deletion through *Data Manager Service* providing the stay identification because the service has information about all stays in different smart environments of the same data subject. The service notifies the owner about the request. The state of the data deletion request is updated to “Pending” when the request is received. In this way, the user can obtain more information about the state of the request and in terms of usability, it makes it possible to check if the request was made. Finally, the owner acts to solve the request. The Owner has the ability to decide if the data is removed or not. Thus, the API provides three possible alternatives to the Owner:

- Owner accepts the request for data deletion and the data is permanently removed from the system;
- Data can be anonymized;
- Owner does not accept the request for data deletion and nothing happens.

Figure 5.9 depicts a possible option to solve the data deletion request. The Owner can decide to anonymize the data as previously mentioned. As said above, this is a valid option but requires significant challenges to avoid the identification of the user through possibly created patterns. It is also important to mention that the implications and nuances of the anonymization task are not within the scope of the work done in this dissertation. Regardless we implemented a simple anonymization scheme that suited our scenario. The data gathered by the Home Assistant is modelled as a set of time series, stored in a time-series database (InfluxDB). For each time-series, we compute the average and standard deviation, for numerical values, and a frequency histogram for text values. This information is sent to the *Privacy Manager Service*, making it available to the Owner, before deleting the raw data.

However, the most simple and effective solution is to remove the data permanently. It is simpler than anonymization because the owner can only select the data deletion for the stay

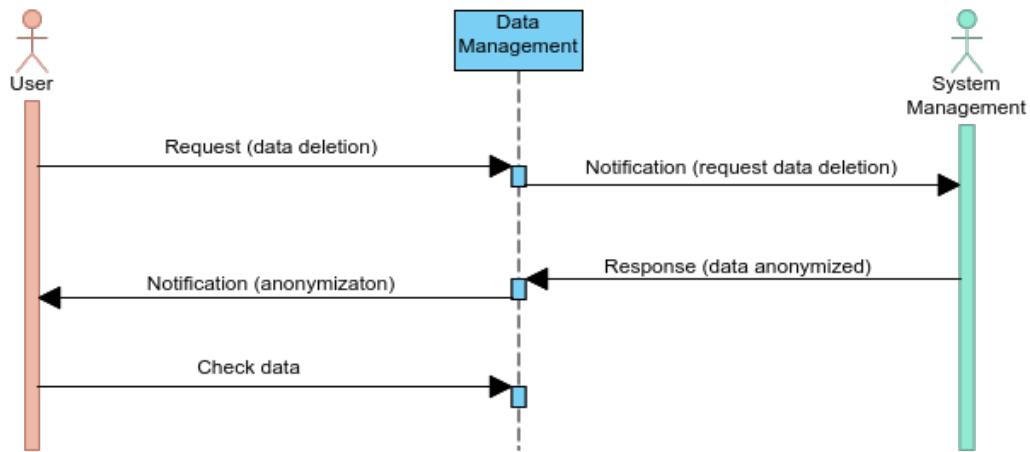


Figure 5.9: Personal data anonymization

provided. Figure 5.10 depicts this action. In this case, the personal data is removed directly from the database where the data was collected during the stay. Usually, the database that stores the personal data collected during the stay is the database integrated in the unified control platform. In addition, we assume that in the architecture described.

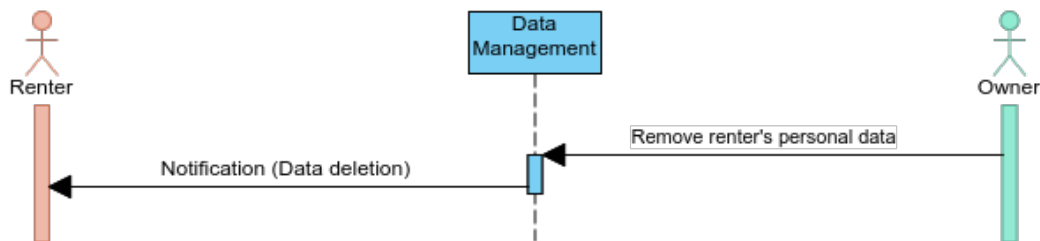


Figure 5.10: Remove personal data from the system

Data Manager Service provides the ability to request the entities involved in the data processing and accepted by the Renter during the check-in. In this way, for each stay, the Renter can request a list of the devices that are collecting data or that collected data during the stay. Thus, the user can check which devices were collecting data, confirming if the list is equal to the accepted devices. The Renter can also request the list of devices that he accepted to collect data. In this way, the data subject can compare and confirm the effect of the consent given. The diagram of this interaction is depicted in Figure 5.11. The Renter requests the list of the accepted devices and the *Data Manager Service* returns it. The Renter can request a complete list of all stays information through the *Data Manager Service*. The idea is to show all information about all stays in a unique request. This is depicted in Figure 5.12. The Renter can also request information about a specific stay or he can request specific information about a specific stay such as the list of the accepted devices mentioned above. However, the ability to request all data from all stays in a unique request is important in terms of data control and in terms of usability.

Data Manager Service receives the user's email as the user identification and the stay's

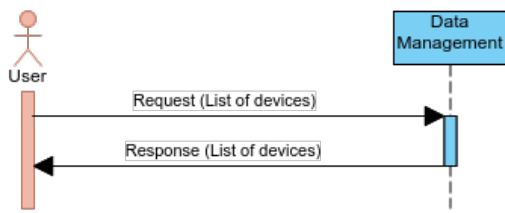


Figure 5.11: List of the accepted devices

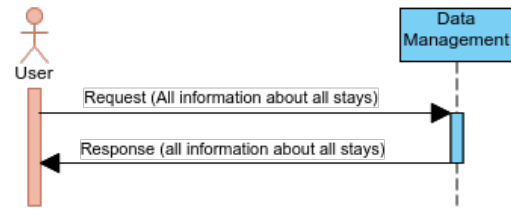


Figure 5.12: Get all information about all stays

information that includes check-in and check-out dates and the type of the smart environment. As response, this service sends the stay identification to the service that sends the user and stay information. This stay information should be provided to the Renter. However, it is not responsibility of the *Data Manager Service* but this information is always available through this service. Figure 5.13 depicts the first interaction.

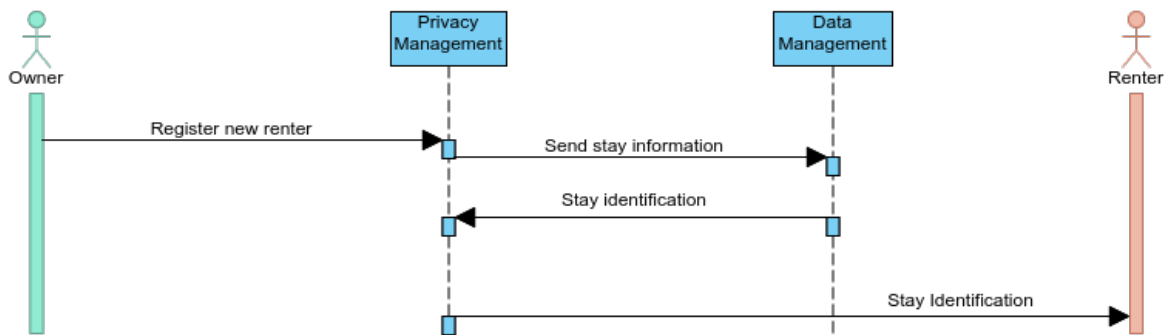


Figure 5.13: Register a new renter/stay in *Privacy Manager Service* and *Data Manager Service*

At any moment, the user has the right to request all data in a CSV file using the *Data Manager Service*. The data he produces during a stay is converted into a CSV file where the identification of the user is not presented. CSV is a text file where stored data are separated by commas. The file stores these tabular data (numbers and text) in plain text format. Each line of the file represents a data record. Each data record consists of one or more fields, separated by commas. CSV files are commonly used to store sensor data because of their easy use. In this way, the user can export the data and use it in another system. If the user provides this file to another system controlled by other entities, it is not the data management or privacy management systems responsibility. The request mentioned is depicted in Figure 5.14. The Renter can see all the information about a specific stay. For this purpose, the Renter should insert the stay identification. If the Renter does not know the identification of the stay, he can request for all the stays identifications and select the one he intends to see more information about. Inserting the stay identification, the Renter can get information about check-in and check-out dates, type of smart environment, the state of the receipts (revoked or not), privacy policies accepted, and the state of data. Figure 5.15 shows this possible action.

The privacy policies presented in the system are simple and understandable. *Data v Service* contains all the privacy policies accepted, including the privacy policies of all the

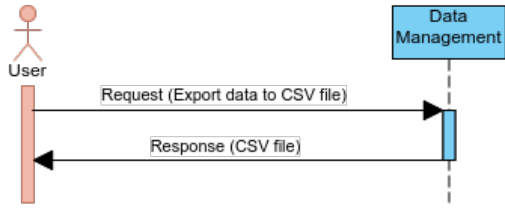


Figure 5.14: Export data to a CSV file

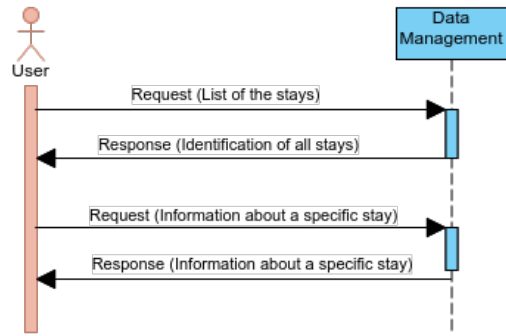


Figure 5.15: Information about a specific stay

devices that collected data during the stay and the services privacy policies. In privacy policies the external entities that will process the data are mentioned. However, in Figure 5.16 we intend to mention the possibility of the user to know which entities are processing his data.

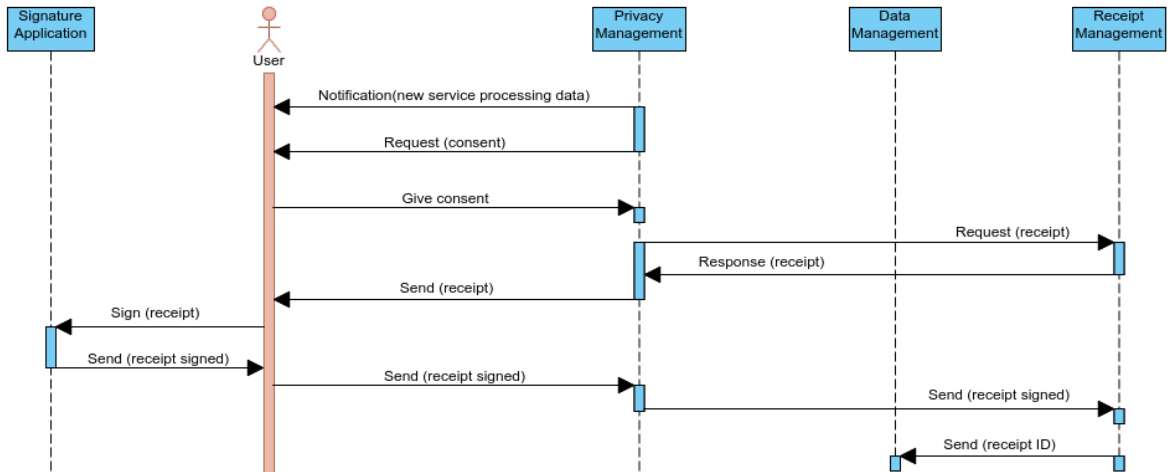


Figure 5.16: Manager consent provided and the entities involved in this consent

All the components added to the architecture are independently deployable. It is possible to add more components to the architecture depending on the main idea of the system created. For example, it is possible to add new components that process personal data, such as a component that uses machine learning algorithms. However, if this new component was added during the stay and after the given consent, then the Renter is notified about it and it will only be enabled if the Renter accepts it. Through the *Data Manager Service*, the user can request the name of the entities involved in the process. Each stay is associated with the policies accepted for each device or service present in the personal data process. In this way, the user can request which entities can access his data and he can also see the respective policies. In addition, it includes policies of the accepted devices. The consent given for each device allowed to collect data is the same as the one accepting the privacy policy. In this way, the policies of the accepted devices are also associated with the stay identification and available in the DEVICE/SERVICE field. It is depicted in Figure 5.17.

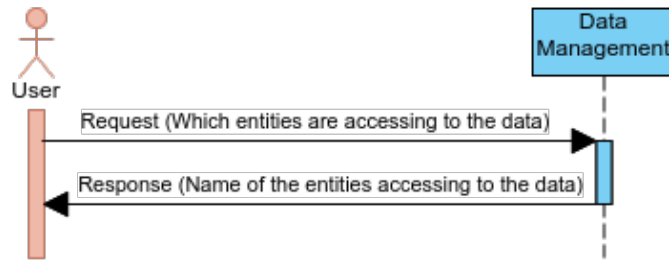


Figure 5.17: Get the entities involved in the personal data process

5.6 CONCLUSION

This chapter described the implementation details of our prototype, and interaction diagrams between the most relevant services of our architecture: *Data Manager Service*, *Privacy Manager Service*, and *Receipt Manager Service*. In addition, the description provides further details regarding support applications, which complement the architecture, by providing the means to cryptographically sign receipts and interact with the smart environment.

CHAPTER 6

Results and Evaluation

6.1 INTRODUCTION

After developing the proposed solution, it is important to mention that the proposed solution tries to accomplish the legislation about privacy and data protection and provides a prototype of how it is possible to have a system able to provide a user control over his personal data transparently and reliably.

In this section, we present the results achieved with the prototype developed. These results include, in a teorical way, how the prototype responds to the legislation requirements. In addition, these results include how the prototype answers the requirements and scenarios established in this document.

This chapter is organized as follows: section 6.2 describes how the proposed solution answers the legislation about privacy principles; section 6.3 explains the study about the impact of the cryptographic overhead on WSN as an initial study to select the sensors to integrate with Home Assistant; section 6.4 presents the results of the Home Assistant integration; and, the results obtained with the prototype are described in section 6.5.

6.2 PRIVACY PRINCIPLES RESPONSE

The main idea of this work is to ensure the principles relating to processing of personal data. According to the legislation, the personal data should be processed awfully, fairly, and in a transparent manner in relation to the data subject. Furthermore, personal data should be collected for specified, explicit and legitimate purposes and adequate, relevant, and limited to what is necessary in relation to the purposes for which they are processed. In addition, personal data should be accurate and, where necessary, kept up to date and kept in a form which permits the identification of data subjects for no longer than necessary for the purposes for which the personal data are processed. After this period, the data may persist in the system but without relation to the data subject. This action is called the anonymization of the personal data. It eliminates personal data so that data subjects can no longer be identified and this data is no longer personal data. Finally, personal data should be processed in a manner that ensures appropriate security of the personal data, including protection against unauthorized processing. Figure 6.1 intends to demonstrate the previously principles described.

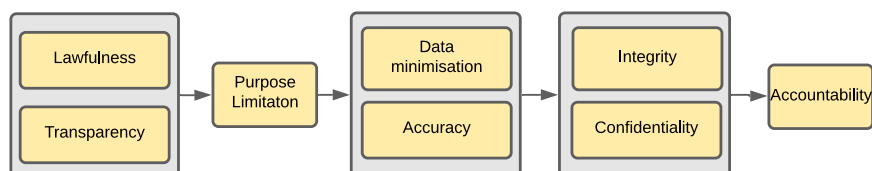


Figure 6.1: Principles relating to processing of personal data

Data manager Service was implemented corresponding to these principles related to the processing of personal data. This service is controlled by the data subject. The data subject

can verify which privacy policies are accepted in the privacy management system and control that his personal data is processed based in a legitimate way. In this way, lawfulness and transparency are guaranteed. Figure 6.2 depicts the bases for processing personal data.

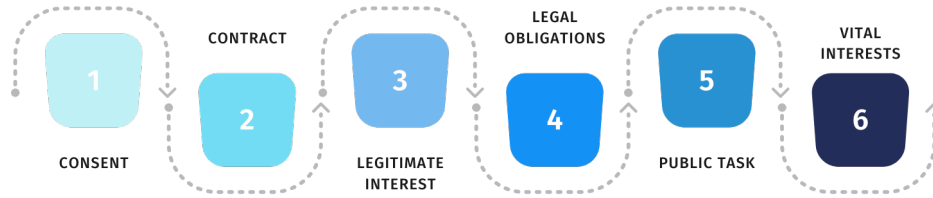


Figure 6.2: Bases for processing personal data

In the *Data manager Service*, the data subject may access the information about the consent of the individual, in this case his consent for data collection. As proof of the consent the data subject has the receipt signed by him when agreeing with the consent.

Moreover, the data subject can verify which data were collected. In this way the data subject can confirm if the purpose limitation was done as well as the data minimization. Through the *Data manager Service*, the data subject may update his identification data.

When the period of data retention is over, the *Data manager Service* is responsible for data deletion. If the data subject requests the data deletion before this period, then the request should be considered and when the statistic purposes are relevant the data is anonymized. When personal data is removed, the data subject is notified. The data subject can only see his personal data and his information ensuring the confidentiality. In this way, we can cover all topics mentioned above.

6.3 EVALUATION OF THE CRYPTOGRAPHIC OVERHEAD ON WIRELESS SENSOR NETWORKS

The focus of this work is based on the user’s privacy but considering the concept of privacy by default there are some important topics that we need to take into account. Thus, the security of the users should be ensured. In this way, to test the impact of the security mechanisms in the WSN several tests were performed. The results obtained were published in a conference paper [9] and the chapter B details the study done to evaluate the cryptographic overhead on WSNs. This study was developed in the initial phase of the dissertation work. During this phase, we had to select the necessary sensors to be integrated into the Home Assistant for the CASSIOPEIA demonstrator. Given the scope of the CASSIOPEIA project, we considered relevant a study that evaluates the impact of the cryptographic overhead on WSN mainly in terms of power consumption.

The security solutions choice considers typical use-cases of a WSN with a WiFi radio connection. We measured this impact by observing the security algorithms’ delays, the additional bytes that the communication protocol needed, and the power consumption recorded on the embedded device. While some of the previous works have already evaluated the delay added by the security algorithms thoroughly (see [112]–[115]), the power consumption and its relation with the measured overhead is still not considered in most studies. In our study,

we are mindful of this parameter and measured it. Our experimental evaluation and main contribution explore the impact of security mechanisms on top of conventional WSN use-cases. We do so by measuring the three fundamental metrics (power consumption, message delay, and additional bytes) using HTTP and MQTT and representative embedded devices (the ESP8622, ESP32, Raspberry Pi (RPI)1, RPI2, and RPI3). Each embedded device will be explored considering the HTTP and MQTT protocols; curve25519 and 521 ECDH curves; and Advanced Encryption Standard (AES)[128, 192, 256] and ChaCha20 cypher algorithms.

Each configuration runs for about one hour, summing a total of 60 hours of experiment (40 hours for HTTP and 20 for MQTT). For each configuration, we measured three different delays: the key agreement (in both parties), the transport delay, and the encryption/decryption delay (encryption on the sensor and decryption on the client). The transport delay was measured by the differences of timestamp by taking advantage of a local deployment of a NTP and the synchronization clocks between the devices. After running the experiments, we were able to gather the power consumption and delay added by the different security mechanisms. The results can be found in Table C.1 and Table C.2.

In the following subsections, we discuss the results from three different perspectives: the delay added by the security mechanisms, the measured overhead in bytes added, and the recorded power consumption. Without taking into account the transport delay between the HTTP and MQTT protocols, there are no major differences between their key agreement and encryption delays. For this reason, we are going to focus on the delays from the HTTP protocol. Furthermore, we are also only focusing on the delays of the sensor device and not the client, since the client was executed in a conventional computer.

6.3.1 Delay analysis

Before discussing the delay of the security mechanisms it is important to mention that the transport delay in the HTTP mechanism is not as reliable as with the MQTT protocols. Due to the Request/Reply nature of the HTTP protocol, both the sensor and the observer have to perform polling on the GET operations. The server is coded to return 400 BAD REQUEST when the data is not available, and the sensor/observer retries the request after some time. These polling operations lead to inconsistencies in the transport delay, the measured delay is the sum of the actual delay and the synchronization made with the polling operations.

This does not occur with the Publish/Subscribe nature of the MQTT protocol. After the initial key agreement algorithm, the observer subscribes the correct topic and awaits for the messages forwards by the broker.

In Figure 6.3 we can find the delays of the key agreement protocols for each board. As expected the usage of different cipher algorithms does not have any impact on the delay of the key agreement protocol. The fastest boards (RPI) are able to compute the shared with the smaller delay. Finally, the delay of P-521 is several times higher than the Curve-25519.

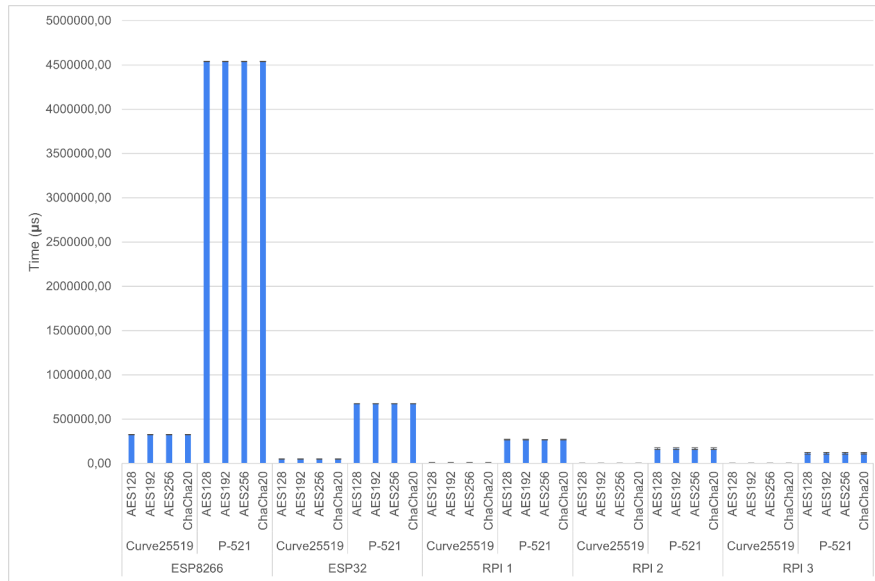


Figure 6.3: ECDH delay analysis

In Figure 6.4 we can find the delays from the cipher algorithm for each board. As expected, the stream cipher ChaCha20 is the fastest overall, as the block cipher AES has a delay proportional to its key size. Contrary to the ECDH the fastest boards are slower at the encryption of the data. There are some possible explanations for this. First, the code for the ESP devices is a highly optimized library developed especially for small boards, while the Python library used on the Raspberry Pi devices uses OpenSSL as the backend for cryptographic computations. To the best of our knowledge, the OpenSSL shipped with the OS is not optimized for embedded devices. Second, while the ESP devices only run a single program (do not have a OS), the Raspberry Pi has a full-fledged OS running with other processes.

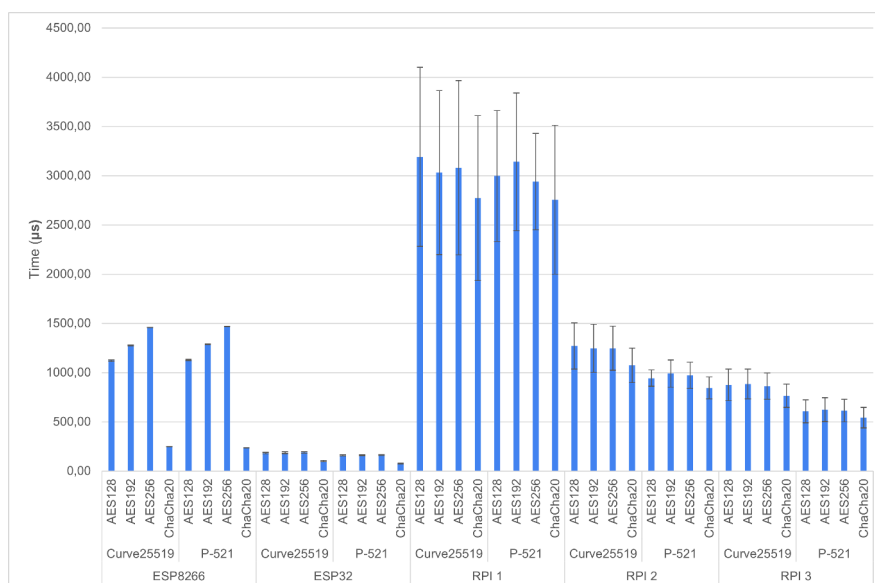


Figure 6.4: Cipher delay analysis

6.3.2 Size analysis

Another way to measure the impact of security methods is to compute the ratio of extra bytes shared between the entities. Let us assume that the normal message, without any cipher applied, is N bytes. All the cipher algorithm evaluated requires an initialization vector (IV), that needs to be sent to the other entity. Let us define the size of IV as I . For the ephemeral Diffie-Hellman it is also necessary to change the public key between the two entities. Let us define the size of the public keys as P .

The ratio for ephemeral Diffie-Hellman is given by

$$\frac{N + 2P + I}{N} \quad (6.1)$$

Where:

label= N : size of the original message (plain text)

label= P : size of the public key

label= I : size of the IV

Both P and I are constants during the session, we can simplify the ratio by replacing the $2P + I$ with k . This simplification generalizes the equation, making it usable for the static Diffie-Hellman scenario. In this scenario, the value k would be only the size of the IV: $k = I$. As the public keys are shared only once, at the beginning of the session.

The ratio simplifies to

$$\frac{N + k}{N} \quad (6.2)$$

Where:

label= N : size of the original message (plain text)

label= k : constant value that depends on the type of ECDH (static or ephemeral)

One important property of Equation 6.2 is that the limit of the ratio when the size of the data increases ($N \rightarrow \infty$), tends to 1:

$$\lim_{N \rightarrow \infty} \frac{N + k}{N} = 1 \quad (6.3)$$

The main aspect of this property is that, since the security bytes are finite in size, as the data part of the message increases, the extra bytes added by the security methods are negligible. In Table 6.1 you can find the ratios for the experiment done in this guide. The size of the raw message is 128 bytes, and the IV is 16 bytes, the size of the public keys are 32 and 132 bytes for the Curve25519 and P-521 respectively. As previously stated, for the static ECDH we do not consider the size of the public keys. The impact of security measures is significant when considering ephemeral Diffie-Hellman, and is minimized with the static Diffie-Hellman.

Table 6.1: Ratios for the experiment

Protocol	ECDH	Ratio
2*HTTP	Curve25519	163%
	P-521	322%
2*MQTT	Curve25519	113%
	P-521	113%

6.3.3 Power consumption analysis

We observed in Table C.1 and Table C.2 that the power consumption within embedded devices does not change significantly with each security configuration. It does vary considerably when taking into account the communication protocol, with HTTP consuming more (as it implements the Ephemeral Diffie-Hellman). A possible reason for this behavior is that the sensor only publishes a message per second, causing the CPU to be idle most of the time.

Although power consumption does not vary greatly in the same embedded device, the most powerful devices consume less power during the experiment. This statement appears counter-intuitive but it is explained by the previous observation. Most of the time, the CPU is idle, the fastest CPU spends more time in idle while the slower CPU spends more time in load, as seen in Figure 6.5.

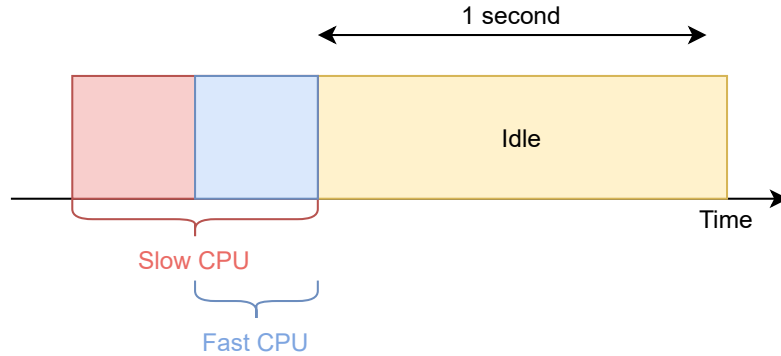


Figure 6.5: CPU processing time for a single message.

This means that, for this scenario, the ESP32 consumes less power than the ESP8266. Similarly, the RPi 3 consumes less power than the RPi 1 and 2. The direct comparison between ESP and Raspberry Pi is not interesting. The ESP embedded device is rather simple and is only able to run a single program. While the Raspberry Pi is a small computer that runs a full fledged OS and is capable of executing multiple processes.

6.4 HOME ASSISTANT INTEGRATION RESULTS

Home Assistant was configured to use the most significant features that are available. The interface has some different ways to depict the smart devices information. It offers access to a graphical floorplan where, instead of a list of devices, users see a representation of the space

similar to the real space in 2D (Figure 6.6a) or 3D (Figure 6.6b). The possible floorplan views are depicted in Figure 6.6.

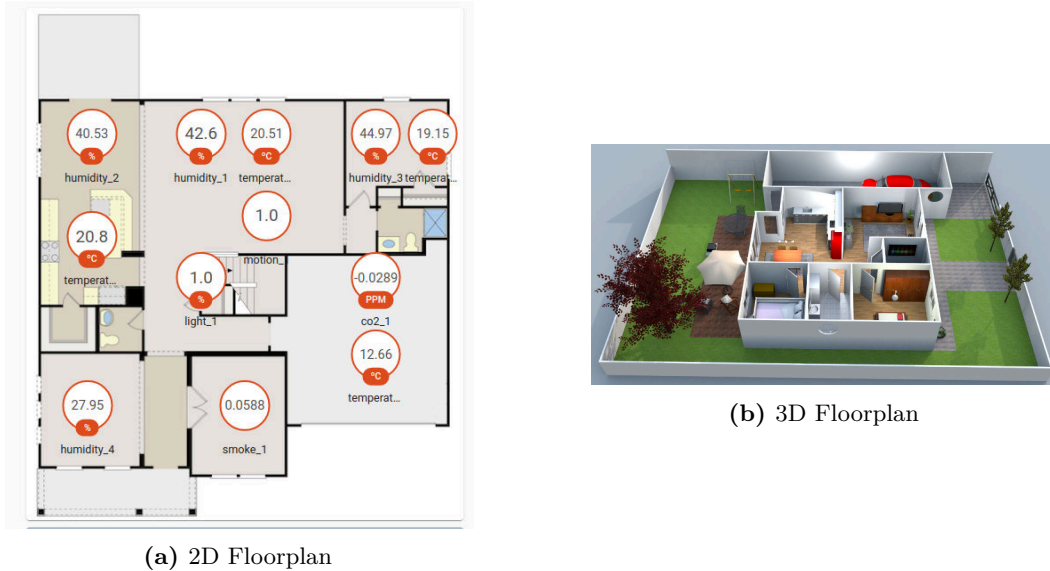


Figure 6.6: Home Assistant Home View using the 2D and 3D floorplans

In a real context the renter is able to select the allowed devices to regard data during the stay. However, for the demonstrator we assume that the Renter accepts all devices available and the interface of the views for the Renter and for the Owner are different. In this way, devices appear as unavailable in the Owner view as we can see in Figure 6.7. Moreover, the Owner cannot change the configurations which is important to ensure the Renter's privacy. In this way, we can ensure that the Owner cannot update the configurations and access to the devices allowed by the Renter which are collecting renter's personal data. On the contrary, the renter's view has all the devices available. The Renter can see and control the devices accepted.

It is also important to mention that an open-source plugin ¹ was added to the Home Assistant integration to control the Home Assistant sidebar. This plugin allows to control the options available in the sidebar according to the user. In this way, we can control which options are presented to the Renter and which options are presented to the Owner considering that the system Administrator has the complete control over the system. However, we require to have more control over the views according to the user roles which means that the content of the views should be different and based in the devices allowed during the stay.

To control the views content, two Home Assistant instances (Figure 6.8) were created to control the Renter's view and the Owner's view. Thus, the restricted instance of Home Assistance (Figure 6.8a) is to be used by the Renter and it is controlled by a developed script to change the passwords. Renter should fill the login fields with the credentials provided by email from the privacy management service. These credentials are randomly generated by the system and provided by email. The Home instance is to be used by the Owner (Figure 6.8b).

¹<https://github.com/Villhellm/custom-sidebar>

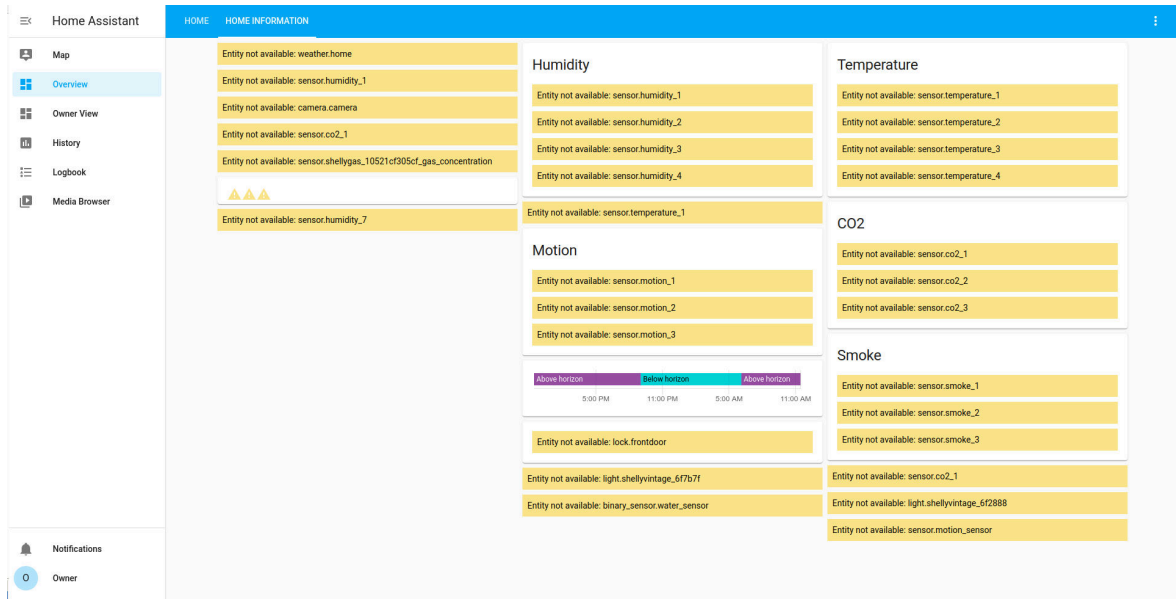


Figure 6.7: Owner view - Cannot see any device

The renter’s password to the restricted Home Assistant view is updated when the Renter leaves the house ensuring that previous renters have no access to the Home Assistant view during the new renter’s stay. In addition, the Owner has no access to the generated password to the restricted view.

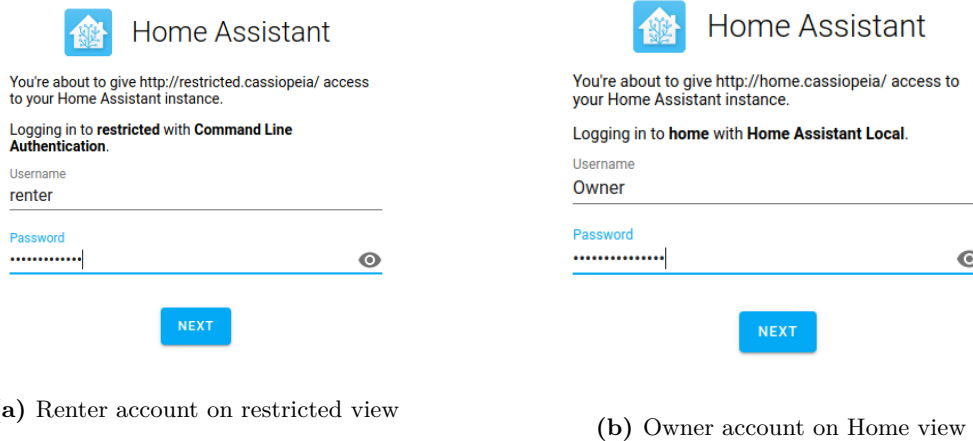


Figure 6.8: Home Assistant views for the Renter (Restricted) and for the Owner (Home)

6.5 PROTOTYPE RESULTS

In this section we associate the different requirements to the respective implementation solution and identify the obtained results. In this way, we try to answer the presented requirements and present an interface that responds to each of them and the messages exchanged between the different services and system components. The services are deployed in different virtual machines: the *Privacy Manager Service* and the *Data Manager Service* are in the same virtual

machine but in different virtual environments using Docker-Compose; The Home Assistant and the InfluxDB are in the same virtual machine to be independent of the remaining services and because this platform was also used in the CASSIOPEIA project; the *Receipt Manager Service* and the Cassandra database are in the same virtual machine due to the high required computational capacity to run the Cassandra database; and finally, the *Signature Application* should run in the user device to sign the receipt, thus in this case the application runs in the personal computer. The PostgreSQL runs in the same virtual machine than the *Privacy Manager Service* and the *Data Manager Service* because it is used by these services and it was also deployed using the Docker-Compose using a volume. For each of these services two PostgreSQL instantiations were used to have two different databases, one of them to the *Privacy Manager Service* and the other one to the *Data Manager Service*. Moreover, the Django services are also deployed using the same Docker-Compose file to the *Privacy Manager Service* and the *Data Manager Service*. *Receipt Manager Service* are also deployed using a Docker-Compose file.

Due to the specific characteristics of the smart environment where the *Privacy Manager Service* may be used, it is important to mention that this system should be deployed locally or adapted to the smart environment characteristics configuring the connection with the smart environment platform, (Home Assistant in our prototype). However, the *Data Manager Service* should be publically available as a remote service to respond to the Renter's requests. This allows the Renter to have full access to his data, even outside the smart environment. When a new stay is created, the *Data Manager Service* receives the information about the endpoint where the data is stored, allowing it to manage the data (either request its deletion, or download a copy of it).

The messages that result from the communication between the different API services are captured using Wireshark. Wireshark is the most popular protocol analyzer. Network professionals, security experts, developers, and educators use it regularly due to the powerful features that this tool has available. It is an open source tool and it is a free packet sniffer computer application [116].

The prototype intends to depict how we can provide a complete data control and transparency for the data subject. This section presents how we can interact with the prototype and the results. The interface was not improved to be used in production, it is only a prototype to exemplify how we can use the services. In this way, the interface is user friendly and it is a simple and intuitive way to use the prototype understanding how it is supposed to work in a real context.

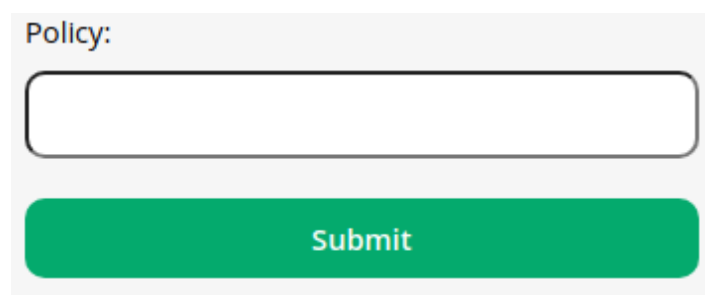
Due to the way the system was developed, there is scalability because the DHT has the potential to have the information in a distributed way and all the components were developed considering the horizontal and vertical needs of scalability. Moreover, it is possible to replace any of the services developed for any other or add more services to the architecture due to the APIs developed. It is only required to follow the methods parameters. In addition, it was ensured that the user's information is not shared between different users and only the services that require to access the personal data have access to them. Furthermore, all the content

provided to the user is easily understandable mostly the receipts and the policies. Finally, the data are stored in a secure way and removed when the consented life period ends or if the user requests it before and the Owner accepts it. Thus, the non-functional requirements are accomplished.

The initial interaction occurs using the *Privacy Manager Service* where it is possible to register the user, add new policies, and associate the created policies to new devices as depicted in Figure 6.13. In addition, it is possible to create the stay and after the stay the creation of a new receipt is requested. *Privacy Manager Service* presents the receipt to the user providing a way to sign it.

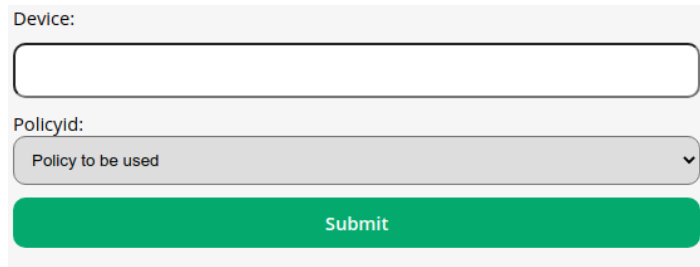
To the prototype, *Privacy Manager Service* provides a simple way to add new policies (Figure 6.9), add new devices (Figure 6.10) and add a new entity (Figure 6.11). This simple way is useful for the prototype but in a production environment these features can be improved to display only the devices and entities available in the system. However, for the prototype purpose we intend to highlight the concept beyond the action. In this way, we intend to show that only the existing entities and devices are available when a new stay is created, which implies to add the entities and the policies in the system. The policies should also be inserted to associate to the devices and entities.

In a production environment it is essential to improve the ability to add devices and entities. A possible option is to receive the name of the devices available in the smart environment directly from the Unified Control Platform, in this case, the Home Assistant. Thus, we can ensure that all the devices are inserted in the service responsible for the privacy manager, in this case the *Privacy Manager Service*. Moreover, in terms of entities it is required that only the entities that process personal data should be registered in the service. Thus, all the existing services should be announced to the *Privacy Manager Service* and the Owner should decide if this entity needs to be registered in the service. In this case, the Owner has the ability to decide (as depicted in Figure 6.9, Figure 6.10 and Figure 6.11)



The image shows a user interface for adding a new policy. It consists of a light gray background. At the top left, the word "Policy:" is written in a blue, sans-serif font. Below this label is a large, empty, rounded rectangular text input field with a thin black border. At the bottom of the form is a prominent green button with rounded corners and the word "Submit" written in white, bold, sans-serif text.

Figure 6.9: Add a new policy



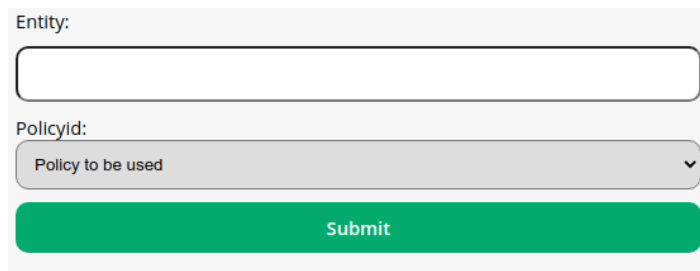
Device:

Policyid:

Policy to be used ▼

Submit

Figure 6.10: Add a new device



Entity:

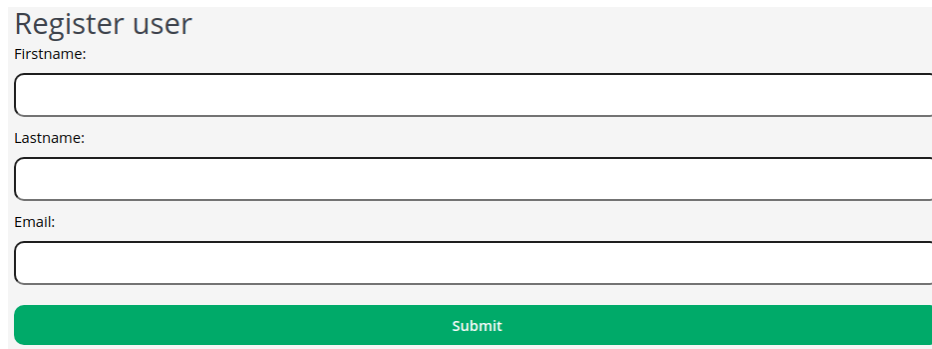
Policyid:

Policy to be used ▼

Submit

Figure 6.11: Add a new entity

The prototype provides the ability to register a new user (as depicted in Figure 6.12) which makes possible to identify the user in the system and consequently, to provide a possible way to ensure that only he can access his personal data. This feature appears as a response to the first and second functional requirements.



Register user

Firstname:

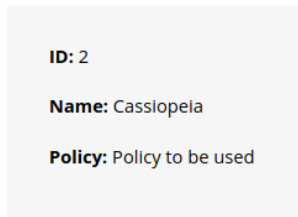
Lastname:

Email:

Submit

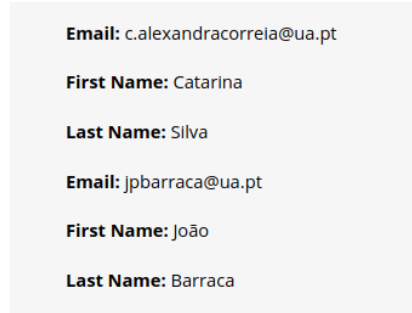
Figure 6.12: Register new user

The prototype also provides a simple way to list the entities available (Figure 6.13), list the users in the system (Figure 6.14), list the available policies (Figure 6.15) and list the available devices (Figure 6.16).



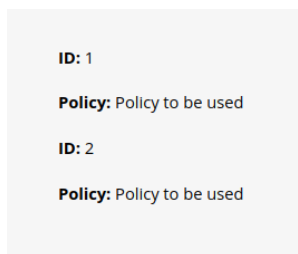
ID: 2
Name: Cassiopeia
Policy: Policy to be used

Figure 6.13: List the available entities



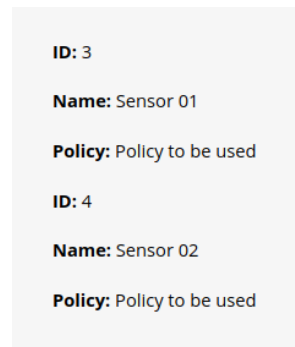
Email: c.alexandracorreia@ua.pt
First Name: Catarina
Last Name: Silva
Email: jpbarraca@ua.pt
First Name: João
Last Name: Barraca

Figure 6.14: List the users in the system



ID: 1
Policy: Policy to be used
ID: 2
Policy: Policy to be used

Figure 6.15: List the available policies



ID: 3
Name: Sensor 01
Policy: Policy to be used
ID: 4
Name: Sensor 02
Policy: Policy to be used

Figure 6.16: List the available devices

Any user registered in the system can be associated to a new stay. *Privacy Manager Service* provides a simple and intuitive way to create a new stay (Figure 6.17) associating one of the existing users to the stay. This stay requires the user identification, in this case, the email, and the dates of the temporary usage of the smart environment. In addition, the consent should be provided to the devices and entities that the Renter accepts to regard or process data during his stay. Thus, it is presented a list of the available devices and available entities and requested the consent for each of them. We decided to have a list of all available devices during the stay creation and provide the ability to the user consent or not consent the usage of it because in this way we can store the information about what devices are consented and what devices are not in the receipt. Thus, the receipt has the consent for all available devices instead of storing only the accepted ones. For the entities the same thing occurs. In this way, the functional requirements about the ability to create a new stay were accomplished.

Register stay

Email:

Datein:

Dateout:

Sensor 01:

Sensor 02:

Cassiopeia:

Figure 6.17: Create new stay

After the creation of the stay, the *Privacy Manager Service* provides the ability to present the receipt to the user in order to request his signature(Figure 6.18) listing the unsigned receipts for a specific user.

User:

User: c.alexandracorreia@ua.pt (1)

Receipt

```
{
  "Receipt Version": 1,
  "Receipt Timestamp": 1629997254.078739,
  "Receipt ID": "d9d7b8fc-3414-4dbe-986b-d4404a186275",
  "Language": "EN",
  "Self-service point": "cassiopeia.id",
  "userid": "c.alexandracorreia@ua.pt",
  "devices": [{"ID": "3", "Policy": 1, "Consent": false}, {"ID": "4", "Policy": 1, "Consent": true}],
  "entities": [{"ID": "2", "Policy": 1, "Consent": true}],
  "Legal Jurisdiction": "Europe",
  "Legal Justification": "consent",
  "Method of Collection": "online web action",
  "Other Information": "",
  "Receipt Fingerprint": "ePO7PTRMdwXcP/iI4it4zpwZsw5OpsA4xwgW+2E42k="}

```

Timestamp Stored: Aug. 26, 2021, 5 p.m.

Timestamp Created: Aug. 26, 2021, 5 p.m.

Stay ID 21

Figure 6.18: Present new receipt to sign

Using the *Signature Application* and the the Portuguese Citizen Card the user can sign the receipt (Figure 6.19). Moreover, the *Signature Application* is developed to verify the user signature in two different times: when the user signs the receipt during the check-in phase and before its storage. It is also possible to validate the signature at any moment but in these two phases it is mandatory to validate its integrity. In this way, the functional requirements identified by the R3, R4 and R6 are accomplished.

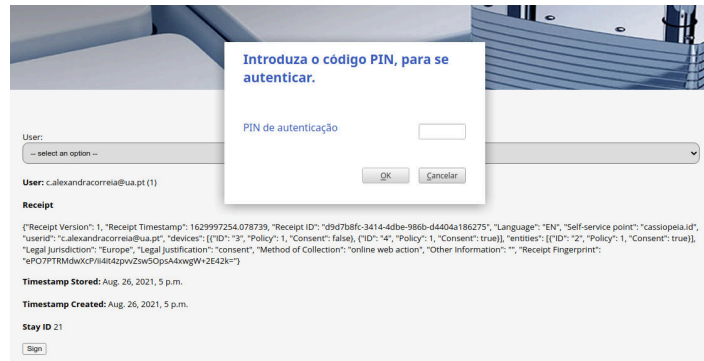


Figure 6.19: Receipt signature process

When the receipt is signed, it is possible to validate the signature using the Portuguese Citizen Card certificate (Figure 6.20).



Figure 6.20: Validation process of the receipt signature

Through the *Data Manager Service* the user can access the application (Figure 6.21) and check the stay identification, the stay information (check-in and check-out dates), the state of his data and the receipt. The receipt contains the information about the consent (entities, devices and policies), other relevant information and it is signed by the user. For each stay it is possible to validate the signature of the receipt, download it, export personal data to a xCSV file and request for data deletion. In this way, the functional requirements identified by R5, R7, R9, R13, R14 and R19 are accomplished.

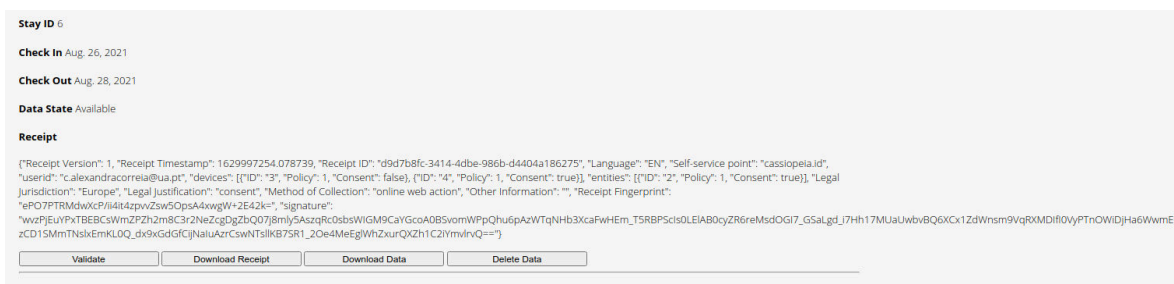


Figure 6.21: Check the stay in the *Data Manager Service*

The exchanged messages between the different services to create a new stay are depicted in Figure 6.22. In the figure, the Client represents the device that is accessing the service, in this case, is the Owner interface. Initially, the Client requests the registration of a new

stay. The *Privacy Manager Service* request a new Receipt from the *Receipt Manager Service*. After filling in the Receipt, the same is stored temporarily in the *Privacy Manager Service*, while it waits for the signature of the Renter. The signature process uses a local plugin, there is no arrow in the flow since the browser (localhost) is communicating with a local service, and the flow graph only draws communications between different IP addresses. After signing the receipt, it is removed from the temporary storage and sent to the *Receipt Manager Service* for permanent storage. Finally, *Privacy Manager Service* sends all the necessary stay's information to the *Data Manager Service*. It is important to mention that some messages are pages or images loading for the web interface.

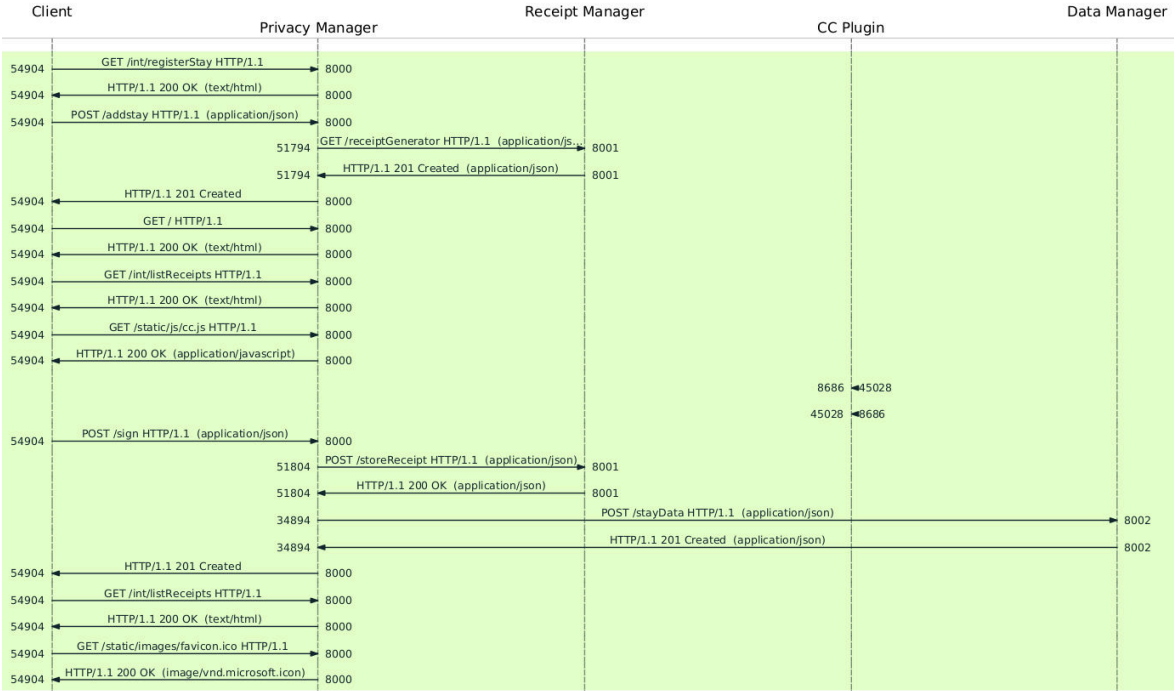


Figure 6.22: Wireshark capture about new register

During the stay, the Renter's personal data are collected and stored in InfluxDB time-serie database. Using the Data Explorer ² interface of the InfluxDB it is possible to access the graphic of the data. In this way, to show the integration of the InfluxDB the Figure 6.23 depicts the data from a temperature sensor considering that the only sensor chosen by the Renter was this sensor.

²<https://docs.influxdata.com/influxdb/cloud/query-data/execute-queries/data-explorer/>

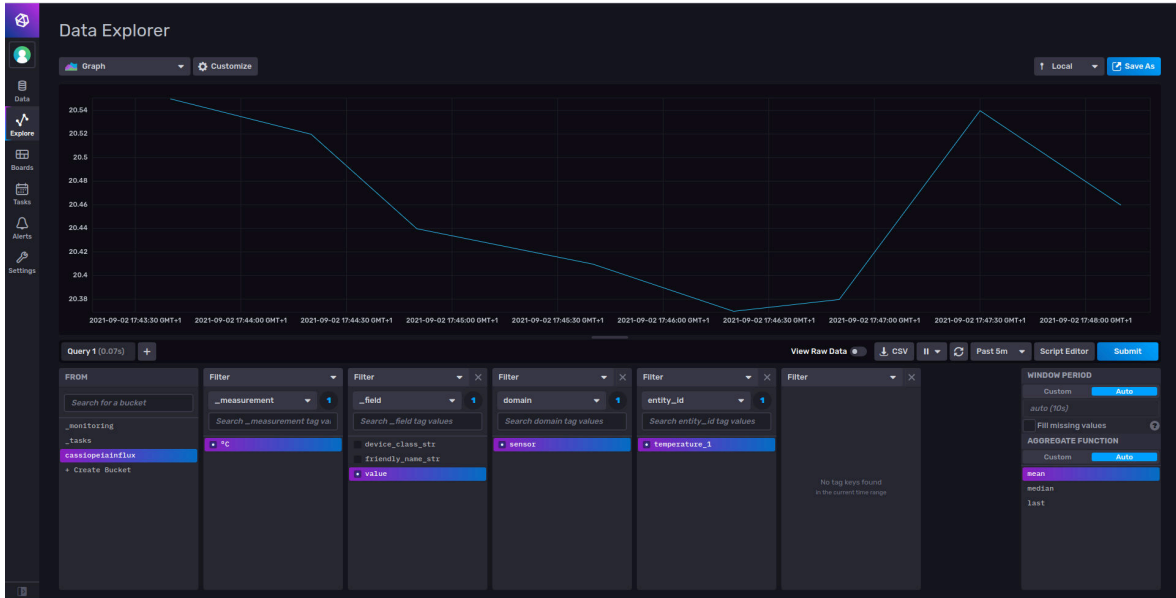


Figure 6.23: Data Explorer of the InfluxDB for the temperature sensor

The Renter uses the receipt to have access to the *Data Manager Service*, as depicted in Figure 6.24. The receipt contains the endpoint of the *Data Manager Service*, by accessing that endpoint the Renter can upload the receipt, the *Signature Application* validates the signature on it using the Portuguese Citizen Card. The Renter is granted access to the stay information and data if the signature is valid.

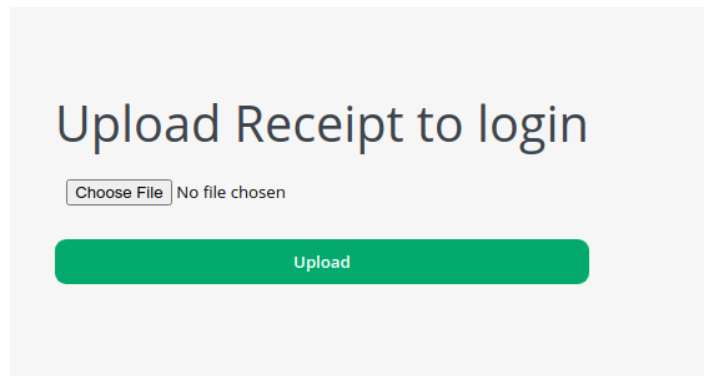


Figure 6.24: Data Manager access using the receipt

As we can see in Figure 6.21, the data state is “Available” which means that the data collected during the stay are available in the system and were not removed. However, the user can request the data deletion and when it happens the data state is updated to the state “Requested”(Figure 6.25). In this way, the requirement identified by R10 was accomplished. It is important to mention that it is possible to export personal data and see the information in the application which is related to the functional requirement R20.

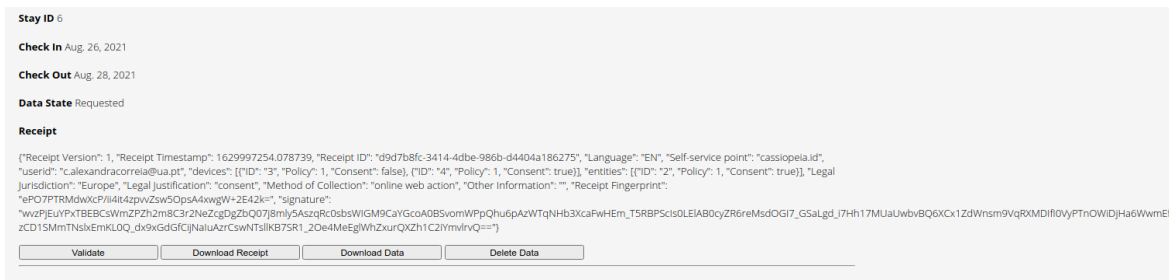


Figure 6.25: Data state updated to “Requested”

When the Renter requests a data deletion through the *Data Manager Service* then the Owner is notified through the *Privacy Manager Service* (Figure 6.26). In this way, the functional requirements identified by the R16 were accomplished.

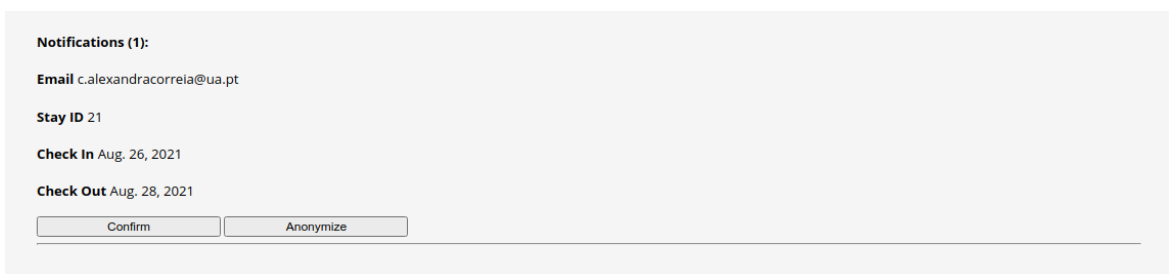


Figure 6.26: Notification about data deletion request

The Owner can ignore the request and the data are removed when the accepted life period ends or the Owner can decide remove or anonymize the Renter’s personal data. In all cases, when the data are removed from the system, the Renter is notified about it which is according to R17. In addition, the data state is updated to “Removed” (Figure 6.27).



Figure 6.27: Data state updated to “Removed”

As it is possible to check in the Figure 6.27 the data were removed and the state was updated. However, the stay reference continuous available which is according to the requirement R18.

The messages exchanged between the different services to request data deletion are depicted in Figure 6.28. The first messages are the validation of the signature of the receipt. the Renter is granted access to the *Data Manager Service* after the validation. The Renter request the deletion of his data, and that request is propagated to the *Privacy Manager Service*. The Owner confirms the data deletion request, and the *Data Manager Service* issues the

necessary methods to delete the data in the data endpoint (the InfluxDB database named home.cassiopeia in the flow). It is important to mention that the Client entity in the flow represents both the Renter and the Owner, both interfaces were open in the same browser.

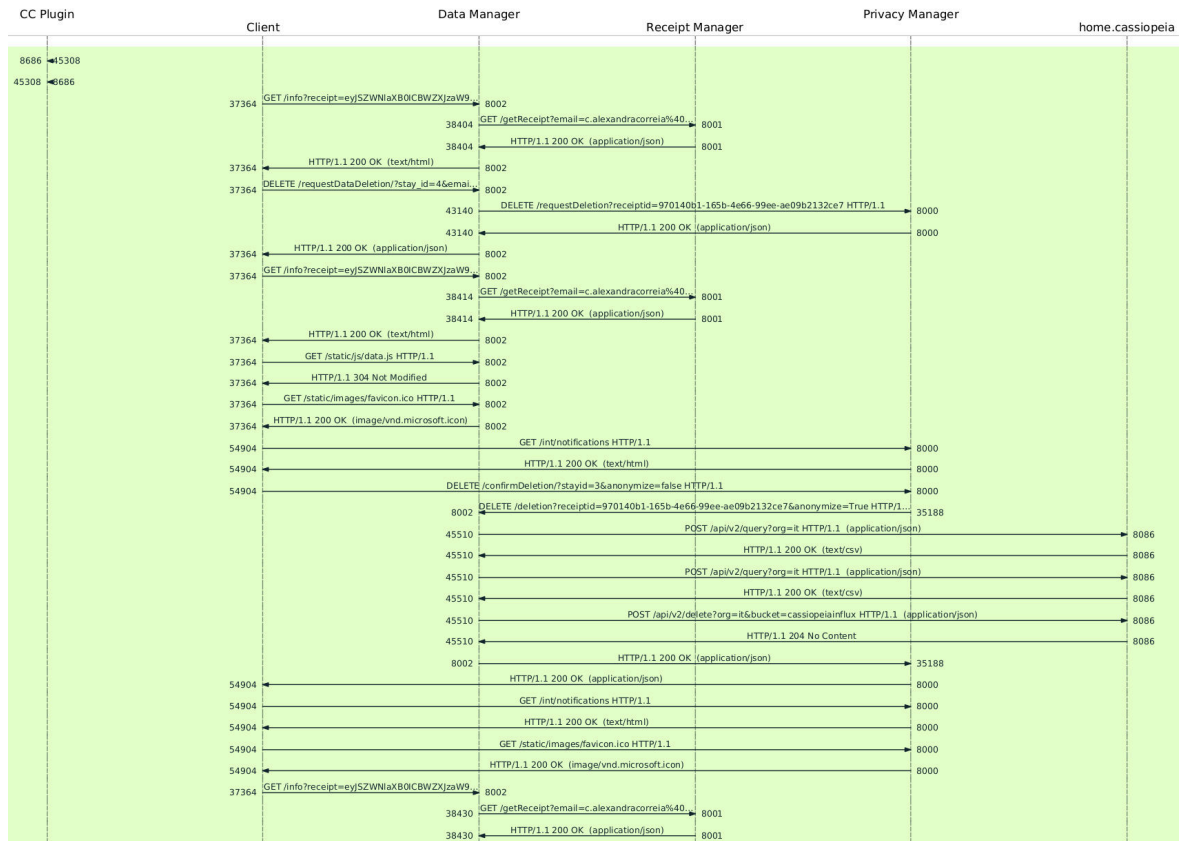


Figure 6.28: Wireshark capture about data deletion

In order to ensure that only the Renter can control his personal data, the *Data Manager Service* is the unique service that can access the database where the data from sensors are stored. The data deletion operation includes the interaction with personal data thus the *Data Manager Service* (data_manager) accesses the InfluxDB database (home.cassiopeia) to request data deletion and as we can see, is the unique service that interacts with the database where the personal data were stored. Moreover, the operation to export data to a CSV file also interacts directly between *Data Manager Service* and InfluxDB database.

The functional requirement identified by R12 is accomplished due to this guarantee about data control and transparency. In addition, this requirement is accomplished also due to the feature that made all the information about personal data available. The Renter has a complete access to his personal data and can manage them which includes a possible data deletion. Considering this, the system provides the ability to associate a user to his data which is related to the functional requirement R8.

The clear separation between the user roles and the services that are available to access the Renter's personal data and considering the restriction in the Home Assistant views, the functional requirement identified by R15 was accomplished.

6.6 CONCLUSION

In this chapter, we presented and discussed the evaluation of our prototype, under scope of a candidate solution to enhance, and demonstrate, privacy and security aspects of SmartBnb environments. The results demonstrate that the prototype complies with the proposed architectures, and demonstrate it's operation and the user interfaces developed. The results also demonstrate an evaluation of the impact of cryptography in low power devices, and the resulting demonstrator. Finally, this chapter also demonstrated the messages effectively exchanged between the different components, in their compliance to the proposed architecture. This evaluation is focused in the interactions triggered by users, and in particular when they exercise their rights over the data persisted (e.g delete data).

CHAPTER 7

Conclusion

7.1 INTRODUCTION

This chapter presents the most important conclusions of the work developed in this dissertation. A prototype was proposed as a solution for the selective sharing permission in IoT environments. The prototype intends to answer the problem found and it was developed according to the solution proposed and the requirements and scenarios established.

The delegation problem was discussed at the beginning of this dissertation, but the architecture proposed does not consider this problem. However, in this chapter we describe a possible solution for the delegation problem as a future work.

The chapter is organized as follows: section 7.2 conclude the dissertation; and the future work as described in section 7.3.

7.2 CONCLUSIONS

The document mentions an increasing problem that we are living nowadays with the constant usage of smart devices and smart environments. The temporary usage of smart devices could be a serious privacy problem mostly when the data subject has no control over those devices. The data subject may lose data control, sometimes he does not know what happened with his data and cannot have access to the entities involved in the personal data processing. The data subject should be informed about the data collection, should provide consent for this collection, and should have proof of this consent. Moreover, the data subject should have permanent access and control of his personal data. At any moment, the data subject should be able to request the data deletion and all updates and he should be sent news about his personal data. In this way, the data subject should have a total knowledge about his personal data.

The present work aimed at proposing a system that provides complete control and transparency for the user. The system intends to ensure user privacy and provides the methods needed to make it easier, understandable, and easier for the data subject to manage his personal data. The prototype developed intends to demonstrate how we can have a complete system to ensure user's privacy during a temporary usage of a smart environment, providing a complete control over his data and a separation between Owner control and Renter control. Thus, the Renter can control which entities and devices are available to access or process his personal data. Moreover, the Renter has a proof of his consent where the entities and devices are consented are detailed as well as the respective privacy policies of each. In addition, the Renter has the ability to request data deletion before their life period ends. In this case, the Owner is able to decide data deletion permanently or just anonymize these data. In both cases, the Renter is notified about the ending of the cycle of data control and transparency. In this way, the Renter can decide, has control over his data, and knows all the information about his personal data from the beginning of the process to the end of the data flow.

As a result of the prototype and considering the architecture developed, this dissertation presents three different services: *Privacy Manager Service*, *Data Manager Service* and *Receipt*

Manager Service. The first one is controlled only by the Owner and the second is controlled only by the Renter. The last one has not any interface is an API to deal with the devices and communicate with the other two services. The *Privacy Manager Service* has a API to manage the first interactions with the smart environment and setup the user's privacy. The *Data Manager Service* has an API to provide a high level of data control and transparency to the user. Both services have a simple, intuitive and user-friendly interface. Moreover, in this dissertation an application to sign the receipt is presented (*Signature Application*) developed to integrate the possibility to use the Portuguese Citizen Card. Finally, the dissertation considers the integration of a unified control platform (in this case, Home Assistant) as a platform to manage the smart devices in the smart environment such as sensors and actuators. Thus, this dissertation also includes the solutions developed to integrate the Home Assistant.

The proposed demonstrator can be used in any smart environment. We can increase the user's privacy following the architecture mentioned in this document. The solution proposed follows the legal regulations to provide an easier system in terms of usability and computationally efficiency.

The architecture proposed can be useful in the next network generations to solve the concerns about privacy in IoT environments. Several tests were made on the system to verify if the algorithms and measures applied to the system work as expected. The tests were positive and the tests showed that the complexity to deceive the system increased.

In conclusion, the work made can be positively evaluated. The objectives were accomplished as well as most of the requirements defined in this work.

7.3 FUTURE WORK

A system with the properties as the one presented in this dissertation should always be prepared to change and be able to adapt to new technologies and solutions. The future work consists of implementing new solutions or improving the developed ones to increase the complexity of deceiving the system even more. During the development of this dissertation, we explored different areas that can be combined to create a more complete platform. Due to the architecture implemented it is possible to add different components to optimize and continuously create a more powerful platform.

Beyond this, some important topics will be discovered during the implementation. The first consideration is about the delegation problem. This problem is out of the scope of the CASSIOPEIA project and in the dissertation it is also not considered. However, we propose a possible solution trying to solve the problem which can be integrated into the proposed architecture. In future work, the implementation of this new component responsible for the delegation problem will be implemented.

The receipt management proposed can be improved to add extra functionality. In this dissertation, receipt management can generate a new receipt for a new entity involved in the data process. However, in the scope of CASSIOPEIA project and this dissertation, the receipt is the proof of the consent given by the renter. Renter's family is not involved in the process of the consent but their data is also collected. Renter assumes the responsibility for

the other members. The receipt management can be improved to lead to the consent of other members. This possible problem is also related to the deletion problem and it is also explored in subsection 7.3.1.

As explained in previous chapters, eSIM can be a powerful option to integrate into the architecture. However, we require more equipment to test these technologies. In this way, the implementation proposed does not integrate this smartcard. In future work, the integration of the eSIM will be considered.

7.3.1 Proposed solution to delegation problem

The delegation problem is not considered in this work. Although we propose a solution to solve it. The idea is to provide the possibility to add a new component responsible for the delegation. In order to solve this problem in a smart environment, we propose the following approach. Considering the following scenarios:

1. A family has a smart home and receives visitors for a short time
2. A family of the renter and the renter are in the smart home (SmartBnB problem).

Both scenarios are about the temporary delegation of permissions. It is necessary to guarantee that at the end of the stay the delegated permissions are removed. In addition, it is necessary to ensure that there is a hierarchy in the delegation process. The owner needs to have the highest level on the hierarchy to ensure that he has control over the house and cannot lose it. For the second scenario, the renter is temporarily the controller of the house but at the end of the stay, the owner needs to obtain the control again.

For the second scenario, we will assume that the family has children and the parents are responsible for one of them and the other one is more than 16 years. Therefore, when the family rents the house the smart devices will collect data from all members of the family. One of the parents signs the receipt, accepts the privacy policy, and can ask for data deletion. However, how can other family members operate in this kind of scenario? The easy solution is to assume that there is a renter responsible for the family (or group of people). Another possible solution is for all the members to sign the receipts (with the same policy because it is a unique house). Considering the first option, if there is a unique renter responsible then there is a unique person capable of controlling the smart devices. We propose a solution based on a delegation process.

In this approach, the owner has complete control of the house when no renter is using the house. When the renter checks in, the owner delegates permission to make the renter capable to interact and control the smart home. At this moment, the user can choose if there is the possibility to delegate his permissions to other renters. If the renter accepts this ability, then he can delegate some permission to the other family members. When the renter accepts this feature in terms of implementation, a token is generated and stored in a database. When the user tries to delegate permissions to another family member, the token used is validated and used to check the specific renters permission. In other words, a renter can only delegate permissions if he has them.

In this approach, we assume that the renter can only delegate permission to another member if the other member is registered on the system. To simplify the process, the owner can delegate to the renter the permissions for creating a user account. To avoid multiple requests to delegate permission, we have decided that the renter can delegate some permissions to other members but not all permissions creating another level of the hierarchy.

When the renter delegates permissions, the token is used to validate the access to the receipt generator. The renter requests a receipt and for that he needs to provide a privacy policy. To simplify, this privacy policy must not be the typical privacy policy difficult to understand and write. The renter only needs to describe the permissions that he will delegate. Something that describes the smart devices included and the process. Then, the family member needs to sign the receipt. A good scenario would be the family member receiving the receipt in the mobile application and signing it using the fingerprint.

The biggest problem in the first scenario is how we can restrict the data collected by the smart devices in the smart home when the visitor enters the house. Smart devices are not selective about the users identity.

In these kinds of scenarios, we cannot have a large chain of delegations. But there are smart environments that may need several levels of delegation.

APPENDIX **A**

CASSIOPEIA project

A.1 CASSIOPEIA USE CASES AND SCENARIOS

The use cases and scenarios for the CASSIOPEIA project focus on solving the privacy concerns in a smart environment, specifically in a smart home. The main goal is to develop a demonstrator capable of managing the renter’s privacy during the temporary stay in the smart home. In this section, we describe the use cases and scenarios for this development.

The first interaction between the Owner and Renter is described in Table A.1. In terms of consent, this use case is very important because it represents the moment when the Renter selects the devices to be allowed to collect data during the stay and proof of the consent is provided to the Renter. The system provides a list of the available devices in the smart home and the Renter should be able to select the devices that he has interest in using during the stay. After this selection and before the smart environment usage, the Renter should provide his consent for this data collection.

Table A.1: Use Case 1 - Prior to arrival

Actors	Owner, Renter
Pre-requirements	Communication channel between owner and renter (email); known list of all smart devices being shared; ability to ask for and receive consent; ability to configure devices for guest use
Rationale	This is the initial ceremony between the Owner and the Renter to gain consent, and to set up the Renter as a sub-user of the smart devices; this formalizes the permission both to collect the Renters personal data, and to grant permission to use various device features.
Demonstrates; Distinction	Consent receipts; delegation; unified controls
Scenario	The Owner and the Renter interact. The Owner shares a list of smart devices in the house, including the features the Renter can control, and requests consent from the Renter for him and his family to have their personal data collected during the rental period. Consent is given, and the Renter gets a consent receipt. The Owner configures the devices for the Renter use, and grants him temporary access.

After the initial ‘ceremony’, the Renter should be able to use the smart environment. Only the allowed devices should be enabled during the stay. In addition, only the Renter should have access to the devices that are collecting data. The Owner should only access the not allowed devices. The delegation process occurs when the Owner allows that the Renter uses the devices providing the permissions to control it. The Renter enters in the house and should be able to set up the allowed devices and use the unified control platform. This use case is described in Table A.2.

Table A.2: Use Case 2 - Arrival and Setup

Actors	Renter
Pre-requirements	The Renter has been given delegated access to device features and data streams; single control surface is in the house
Rationale	The Renter begins the rental period with his family, and fully sets up to use the smart devices in the house.
Demonstrates; Distinction	delegation; use of unified control app
Scenario	The Renter arrives at the house for the first time and enters via the smartlock. The Renter authenticates to a centralized control app, and then downloads necessary apps to control/view devices, and confirms that he can see video, control devices, and get notifications. The Renter sets up other family members to use the devices.

As previously mentioned, the allowed devices should be only available for the Renter. The content of the views should be dynamically configured to ensure the Owner has no access to

the enabled devices in the smart home. The permissions to control the smart devices are delegated to the Renter. The Renter has complete control over the allowed devices. During the stay, the renter can use and access the smart devices. It is required to ensure that after the stay, the Renter loses this control to ensure the next smart home user’s privacy. This use case is described in Table A.3.

Table A.3: Use Case 3 - Lock, Doorbell, Lights

Actors	Renter, family members
Pre-requirements	Renter and family members have been set up as users on all devices
Rationale	This illustrates basic usage of various smart devices in the home
Demonstrates; Distinction	Easy delegation of basic smart device functions to temporary guests
Scenario	The Renter configures the smartlock to recognize him and his family members when they approach the house to unlock the door and turn on the entry lights. The Renter configures the smartlock to turn on the lights periodically when the house is empty. The Renter sets up a virtual assistant to show an activity feed of smartlock activity. The Renter sets up the lights to fade up at waking hours, and fade out for bedtime. The Renter receives notifications on his phone when the doorbell is rung.

The renter can use security cameras to control the environment around the house. The surveillance camera management is similar to the other smart devices. The surveillance cameras are available for the Renter if he accepts it. Therefore, if the Renter is using the surveillance cameras then the Owner cannot access these devices. This use case is described in Table A.4. The delegation of permissions between the owner and the renter about the surveillance cameras has a special note. The renter can live in the smart home for a temporary period. The period of the stay is mentioned at the check-in and is part of the initial contract. Thus, the stay has duration information. At the end of the stay, it is expected that the renter leaves the house, assuming that when the stay ends the owner can access the surveillance cameras to check if the renter leaves the house.

Table A.4: Use Case 4 - Cameras and Security

Actors	Renter, family members, Owner?
Pre-requirements	Renter and family members have been set up as users on all devices
Rationale	This illustrates basic usage of smart security devices in the home
Demonstrates; Distinction	Easy delegation of security systems to temporary guests; context-specific access by Owner?
Scenario	While away, Renter gets notification that someone has approached the property. The Renter looks at the video feed and sees that it is the postal carrier. The carrier has a package that must be signed for, so the Renter speaks through the external camera and asks the carrier to come back the next day. Renter receives notifications when a window is opened (security system). While away, the Renter gets notification that there is movement in the kitchen. He checks the video feed and sees his daughter getting cookies. He speaks through the camera and tells his daughter that she can have one cookie because dinner will happen soon. Renter receives an alert and sees video when a security camera in the child’s room detects sound when the child is supposed to be sleeping.

As said above in Table A.4, at the end of the stay, the Owner can use the surveillance cameras to verify if the Renter is not in the home. In addition, the Renter can request data deletion. When the Renter leaves the smart home then he should be able to request for data deletion using the proof of the consent provided during the check-in. The proof of the consent should have the necessary information to do this request. In addition, this proof of consent should have the period of data retention information. This period is also accepted by the

Renter during the check-in. After the data retention period, the data should be removed from the system. However, the ability to request the data deletion in advance should be provided to the Renter. The Renter requests the data deletion and the Owner can accept it or not. If the Owner accepts the request then he should be able to remove the data. However, if the Owner does not accept the data deletion in advance, then he can ignore the request. When the data is deleted from the system then the Renter is notified about the data deletion. This use case is described in Table A.5.

Table A.5: Use Case 5 - Departure and Post-rental

Actors	Renter, Owner
Pre-requirements	Consent receipt in place; ability to delete maximum amount of Renters personal data from all devices; delegation timeout or remote cancellation of guest access
Rationale	This is the closing ceremony where the Renter leaves physically and in terms of access control; maximum amount of Renters personal data should be deleted upon completion of the rental period; full control is returned to the owner
Demonstrates; Distinction	full lifecycle of consent receipts; conclusion of delegation; proof of deletion of a guests personal data
Scenario	The Renter and his family leave at the end of their rental. At the formal rental period end, all guest access to the devices is revoked, and all collected personal data is deleted where possible. The Owner is notified that the Renter has left, that guest permissions have been revoked, and that (most of) the Renters personal data has been deleted. The Owner remotely checks the internal and external security cameras to confirm that there is indeed no one in the property. The Renter is sent a consent receipt indicating that his consent for data collection is now revoked, and that (most of) his personal data has been deleted. The Owner sends a message indicating when the remaining personal data will be deleted.

The use cases presented describe all the interactions from the beginning of the stay until the end of the stay. Using the use cases, it is possible to define the most relevant requirements in the scope of the project. Considering the use cases previously mentioned, the feature requirements are the following:

1. Owner delegation: the owner delegates access to the home to a specific renter
2. renter consent: the renter signs the consent and can use it to prove the permissions given
3. notification to renter of data use: the renter should be informed about his personal data
4. renter authentications: for security aspects, the authentications should guaranteed
5. renter use of devices: the renter allows a specific set of devices to collect data and only these devices can collect data. the receipt should have information about it.
6. renter visibility of data flows: the renter should have knowledge about his personal data flow
7. owner restrictions on device features: the owner cannot access to the renter's personal data
8. limitations on the owner ability to surveil: the owner cannot use surveillance cameras to control the renter
9. proof to renter of lack of owner surveillance and personal data access
10. unified control surface: it should only exist a unified control platform of all devices in order to guarantee that unique devices present in the home are presented to the renter

The initial development preparation is done. The use cases, scenarios and requirements are essential to be done before the implementation. The architecture is the next step before the implementation. The architecture of the CASSIOPEIA project is presented in section A.2.

A.2 CASSIOPEIA ARCHITECTURE

The main goal of the CASSIOPEIA project is to provide a demonstrator trying to solve the SmartBnB problem. Temporary user’s privacy should be ensured and this includes the consent for data collection, the proof of this consent, the delegation of permissions, access control, and data deletion.

In the context of the CASSIOPEIA project, the *Privacy Manager Service* includes the integration of the Home Assistant and it is responsible for the request of a receipt. All the smart devices were integrated into Home Assistant to control the smart home. The generation and storage of the receipt is done by the Consent Receipts Management. To ensure data transparency and data control by the data subject it is added a *Data Manager Service* to the architecture. This system is controlled by the Renter and provides the ability to request data deletion.

Figure A.1 depicts the architecture of the CASSIOPEIA project.

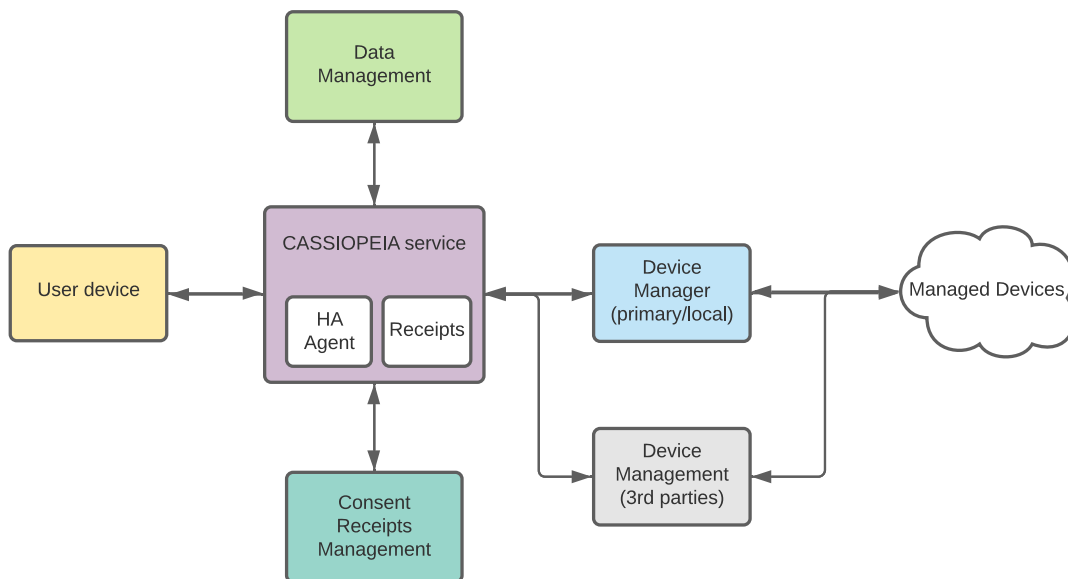


Figure A.1: Overview of the components in the Demonstrator

The user interface allows users (Owner and Renter) to interact with the platform all through the lifecycle of the service. The interface is web-based and accessible using a common and familiar web browser. Under the scope of CASSIOPEIA we have developed custom web interfaces to implement the demonstrator, and integrated the Home Assistant, which presents a single control interface.

The CASSIOPEIA service is the core of the CASSIOPEIA project scope. The demonstrator is based in a way to ensure the user’s privacy when controlling the smart devices that are allowed to collect data, controlling the views based on the user roles, and request the Renter’s consent. In this way, this component is responsible to control Home Assistant views, devices, and access control. In addition, it is responsible for requesting the generation of the receipt. The Figure A.2 intends to depict the interaction between the *Privacy Manager Service* with other components in a simple and basic usage.

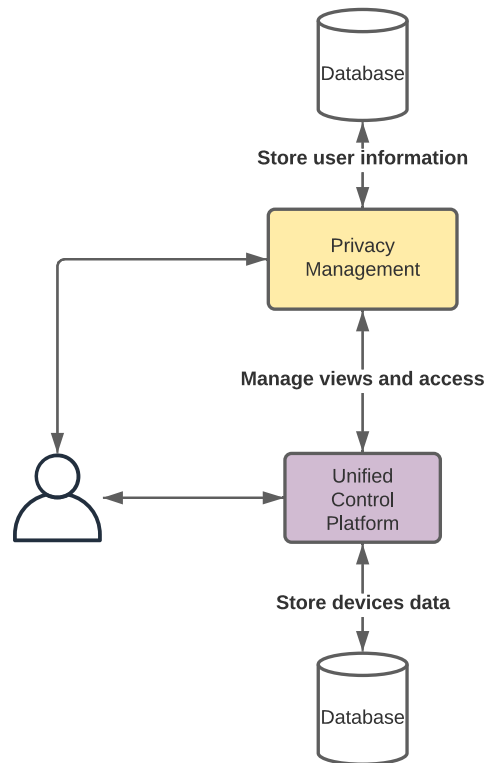


Figure A.2: Interactions of the *Privacy Manager Service* and the remaining services

Privacy Manager Service was developed to control the unified control platform, manage the stay data such as the check-in and check-out dates and the user identification; it is also responsible for interacting with the consent receipt management to request the generation of a consent receipt.

The Home Assistant, as the unified control platform, is a single point of device control. The user can access the platform to check or update the state of the devices. A possible action is to check the room temperature and turn on the lights in the living room. This interface shows all allowed devices and makes it possible to interact directly to the smart home.

The *Privacy Manager Service* is responsible for performing the initial configurations of the unified control platform present in the smart environment. The first configuration is to enable only the accepted devices by the renter. In addition, this configuration implies the

restriction in the content of the views (renter and owner). The Home Assistant was configured to dynamically change the content of the views depending on the user role.

Moreover, this system is responsible for controlling the authentication of the renter and changing the password at the end of the stay. After the stay, the renter loses access to the unified control platform and the *Privacy Manager Service*. We need to ensure that the password is changed and the renter loses access. It is extremely important because the smart environment can have more renters, and it is not expected that the other renters can see the information about the previous ones. Another important thing to guarantee is that the owner cannot have access to the password of the renter with access to his view. This password should be generated by the system and automatically sent to the renter without the interference of the owner.

The renter should only access his view and the owner should only access the owner's view. To control what is accessible for different users, the unified control platform has three different views. The renter view and the owner view are dynamically updated based on renter preferences. During the stay, the owner can only see the devices not allowed to the renter. In this way, the owner can access the devices that are not used by the renter. The renter views only make available the devices allowed to the renter at the check-in. The idea is to restrict the view of the owner during the stay and avoid that the owner can see renters' data. There is the administrator view for developers to manage the configurations, such as authentication configurations or add more devices in the system. The administrator view should not be accessible during the stay because all the information is available in it. As a consequence, the owner should not be the administrator. The administrator should be the developer or responsible for the maintenance of the system.

After the stay, the renter loses access to this platform and the owner gains access to the surveillance cameras to check if the renter has left the house. Meanwhile, the renter who left the house should signalize this action to inform the *Privacy Manager Service* that is necessary to reset the data in the unified control platform. It is important to guarantee that the future renters or the owner (after the stay) cannot access the personal data collected. This is different from removing data because the data is stored in other databases. We will only create the other instance of the database for the next renter.

Moreover, to manage the restrictions of the views, this platform is controlled by the *Privacy Manager Service*. Through the *Privacy Manager Service*, the owner can create a new renter in the house and give temporary access to the devices. The renter should choose which devices he wants to enable and collect data during the stay. Moreover, the owner can request a receipt from the consent receipt management. The renter receives the receipt and signs it. Then, the receipt is stored by the Receipt Management. All these interactions occur at the check-in and through the *Privacy Manager Service*. Through this component it is possible to register a user, request, a receipt, request to sign the receipt, see the signed receipt, ask for the consent, list devices, add policies, and list policies. For this demonstrator, *Privacy Manager Service* also represents the Web Interface for the users to interact with the system.

The renter can interact with the *Privacy Manager Service* through the web application in

order to accept the devices that will collect data, give consent and sign the receipt.

After the period of stay, the owner can use surveillance cameras to check if the renter is not in the smart home. Thus, the owner can access the unified control platform and use the surveillance cameras.

The owner can access the unified control platform, but with different views. After or before the stay, the owner can access everything in the smart home, such as smart devices and surveillance cameras. However, during the stay, this view is restricted. The owner can access smart devices not allowed by the renter as the smart devices not allowed by the renter are not collecting the renter's data.

The Device Manager is the interface between *Privacy Manager Service* and the actual physical devices. We used Home Assistant for this function, following the strategy of integrating well-known and openly available solutions. In a nutshell, it sends requests to direct interfaces provided by the devices to take data and request actions. These requests use RESTful APIs exposed by Home Assistant documentation ¹.

The consent receipt works as the proof of the consent given by the Renter during the check-in. At the beginning of the system usage, the Renter should accept the privacy policy of the system. In addition, as previously said, the Renter should select the allowed devices to collect data during the stay. After this selection, the Renter should provide the consent. The receipt is the proof of this consent that includes the privacy policy of the system and the privacy policies of the accepted devices.

The receipt structure was partially based on Kantara Initiative ² but with some differences. The used fields from this possible standard were the following:

- Version: The version of this specification to which a receipt conforms.
- Consent Timestamp: Date and time of the consent transaction.
- Collection Method: A description of the method by which consent was obtained.
- Consent Receipt ID: A unique number for each Consent Receipt.

In the scope of the project we consider important to add the following fields to the previously mentioned ones:

- Organization: Name of the organization.
- Product: Identification of the service provided.
- User identification: email of the data subject.
- Privacy Policies reference: URL to the privacy policies accepted.
- Other information: free fields that can be filled with any relevant information.

In general the structure of the receipt should have the fields presented in Table A.6.

The Renter can request the data deletion and it occurs at the end of the accepted retention period accepted at the check-in. This period makes part of the system privacy policy. However, the Renter can request the data deletion before the end of this period. The *Data Manager Service* is the component responsible for data deletion. When the data is removed from the

¹<https://developers.home-assistant.io/docs/api/rest/>

²<https://kantarainitiative.org/>

Table A.6: Receipt structure and possible option to fill the fields

Fields	Description and Guidance
Receipt version	CASSIOPEIA-202102
Receipt Timestamp	Unix timestamp milliseconds
Receipt ID	UUID v4
Method of Collection	"Online web action"
Organization	NGI CASSIOPEIA Demonstrator
Product	Homeassistant device identifier
User identification	Email of the user. For example, c.alexandracorreia@ua.pt
Privacy Policies identification	List with the privacy policies identifications. For example, [ID1, ID2,...]
Other information	test123 .. extra information

system, the *Data Manager Service* is also responsible for notifying the Renter about the data deletion. The focus of this document is to increase the potential of this component to ensure the user's privacy while are using the smart environment and after the smart environment usage.

APPENDIX **B**

**Evaluation of the cryptographic
overhead on wireless sensor
networks**

The IoT is an important research topic because it enables massive machine type communications (mMTC) and data acquisition from small sensors that may unlock solutions to today’s most pressing societal challenges, such as the Industry 4.0 revolution, smart grids, smart cities, smart crops, or Health. Privacy and security are among the most significant concerns of deploying IoT [117], [118]. Therefore, the desired functionality of WSN and the required security mechanisms are two important topics that must be researched together.

Figure B.1 depicts the architecture of the experiment and the delays measured. The gateway is responsible for the communication between the observer and the system and the key server is responsible for sharing the public keys of the session. The idea of the experiment is to measure the delays: key agreement, transport delay, and encryption/decryption delay.

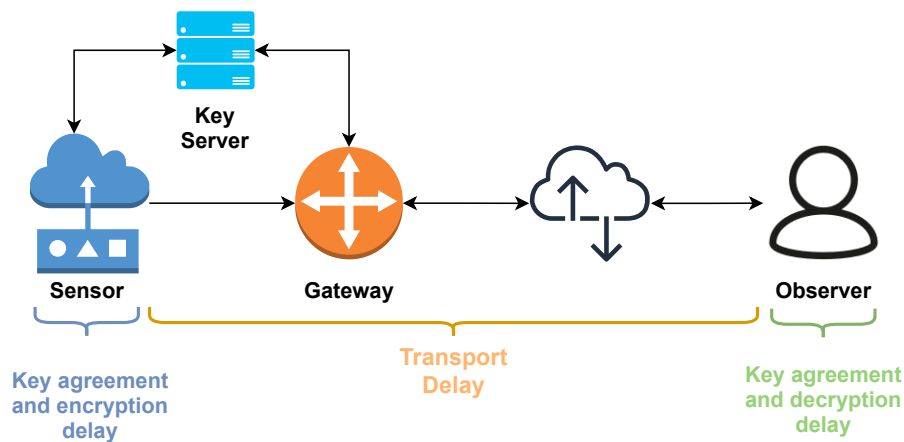


Figure B.1: Delays measured on the experiments.

Ephemeral Diffie-Hellman was implemented on top of HTTP, while the static Diffie-Hellman was deployed on top of MQTT.

When a key exchange uses Ephemeral Diffie-Hellman a temporary DH key is generated for every connection and thus the same key is never used twice. On the other hand, a static Diffie-Hellman the shared key is generated at the beginning of the session. The synchronous nature of HTTP makes it ideal for the ephemeral implementation of Diffie-Hellman, while the asynchronous nature of MQTT makes it ideal for static Diffie-Hellman. It is important to mention that even in a static Diffie-Hellman implementation the keys should be updated regularly (every n messages), however, that is not explored within this work.

Due to the constrained resources on the embedded devices, the key server was implemented as a method to share the keys between entities and share the data when the HTTP was used. It was possible to run an HTTP server on the embedded device and on the client, but that could have an impact on the security performance.

All the connections between the embedded device (sensor) and the servers (either the HTTP key exchange server or the MQTT server) were unencrypted. The arduino vanilla libraries for MQTT communication did not support TLS, it is possible to use with ESP specific code, but that could also have an impact on the measurement of the security performance.

In our experiment, the MQTT protocol uses static Diffie-Hellman, meaning that the session key is computed only once. For this reason, we did not vary the ECDH since it has minimal impact on the experiment.

B.0.1 Hardware

Five different embedded solutions were selected for this work: ESP8266, ESP32, Raspberry Pi 1, Raspberry Pi 2 and Raspberry Pi 3. It is important to mention that all boards, except for Raspberry Pi 1 and 2, have native wireless support. The remaining boards use an USB dongle¹ to have wireless capabilities. All embedded devices were connected using WiFi 802.11n on a dedicated access point (AP). On this AP were only connected embedded devices and the laptop that served as servers and the client. The main characteristics of these devices can be found in Table B.1.

The BME280² was selected as the sensing device for the embedded system. This device is an environmental sensor with temperature, barometric pressure and humidity.

Table B.1: Devices and characteristics

Device	Cores	Clock CPU	RAM	Wireless
ESP8266	1	160 Mhz	64 KB	802.11 b/g/n
ESP32	2	240 Mhz	520 KB	802.11 b/g/n; BLE
Raspberry Pi 1	1	700 Mhz	512 MB	WiPi dongle (b/g/n)
Raspberry Pi 2	4	900 Mhz	1 GB	WiPi dongle (b/g/n)
Raspberry Pi 3	4	1.2 GHz	1 GB	802.11n; BLE

Whenever possible the code for the work was written in Python, due to its isolation (through virtual environments), a vast collection of libraries (through the usage of the pip package manager), and the portability of the code. For the ESP boards, the code was written in C, using the Arduino IDE environment³.

For this work it is necessary to measure the power consumption of the devices. We have used an USB meter for that task, the *USB Safety Tester J7-t*⁴.

B.0.2 HTTP

As seen in Figure B.2, there is a single HTTP server. The same server has methods to store/retrieve the public keys (ECDH key exchange), and store/retrieve data (JSON document).

¹<https://thepihut.com/products/raspberry-pi-wipi-wireless-adapter>

²<https://www.adafruit.com/product/2652>

³<https://www.arduino.cc/>

⁴<https://budgetlightforum.com/node/48386>

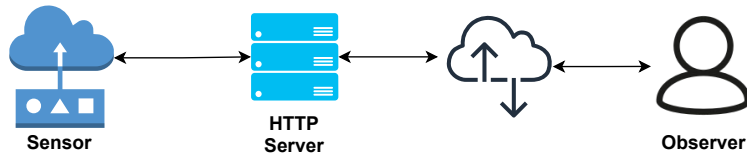


Figure B.2: WSN architecture for HTTP.

The typical sequence of messages is depicted in Figure B.3. Take into consideration that the observer and the sensor do not operate at the same speed and there should exist some retries in a normal execution (the message retries are not depicted within the diagram).

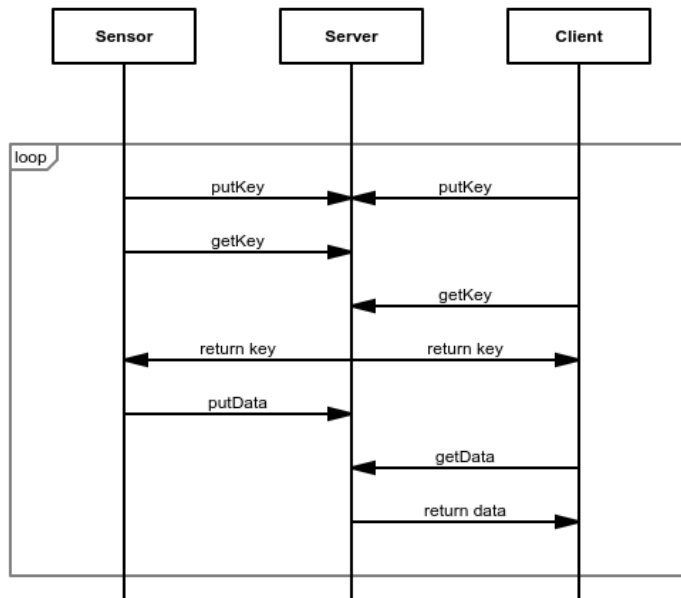


Figure B.3: HTTP messages exchange

B.0.3 MQTT

As seen in Figure B.4, there are two servers: the key server (same HTTP from the previous execution) and the MQTT server. The key exchange is performed at the beginning of the session, after that the observer subscribes the topic **sensor/bme280** and the sensor publish to that topic.

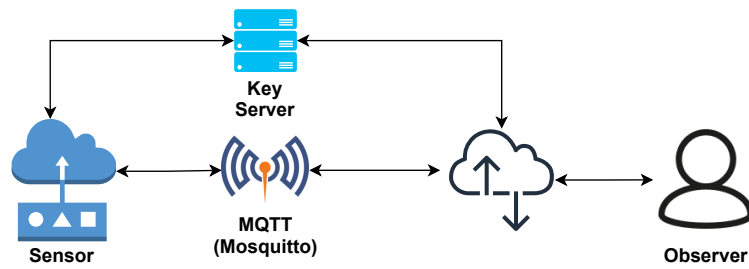


Figure B.4: WSN architecture for HTTP.

The typical sequence of messages is depicted in Figure B.5. Take into consideration that the observer and the sensor, do not operate at the same speed and there should exist some retries during the key exchange operation.

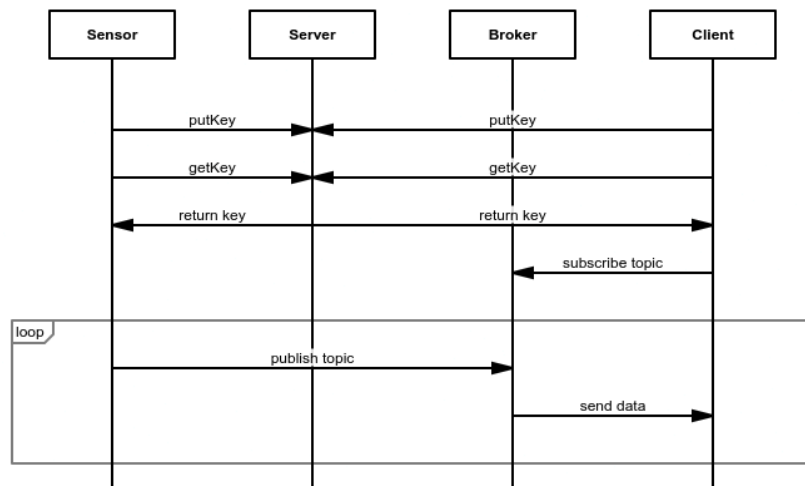


Figure B.5: MQTT messages exchange

It is not possible to select all possible combinations, but we have tried to select a representative approach. It is important to mention that the selection of secure algorithms was guided by the recommendations present within the official Arduino Cryptography Library⁵. ECDH [119] was selected as the method to securely exchange the secret key for that session, as suggested by the guidelines of the previously mentioned library. The library offers two different ECDH: curve25519 and P-521. The ESP devices support some hardware accelerator cryptographic operations, but they are more limited than the cryptographic library offered by Arduino. This library is also available on a larger set of embedded devices, not only Arduino and ESP.

⁵<https://rweather.github.io/arduinoilibs/crypto.html>

**Experiment Results - Evaluation of
the cryptographic overhead on
wireless sensor networks**

Device	ECDH	Cypher	Sensor Key (μ s)	Encrypt (μ s)	Transport (μ s)	Client Key (μ s)	Decrypt (μ s)	Wh	Ah
ESP8266	Curve25519	AES128	328562,84 ± 33,98	1120,59 ± 7,03	1021216,14 ± 563814,05	366,54 ± 1439,17	145,54 ± 10,97	0,241	0,052
		AES192	328571,03 ± 30,24	1275,68 ± 5,33	1012832,75 ± 565148,42	366,98 ± 1442,93	146,95 ± 9,27	0,241	0,052
		AES256	328570,79 ± 31,11	1456,80 ± 2,17	1022337,36 ± 557077,59	372,20 ± 1476,97	149,53 ± 19,47	0,244	0,053
		ChaCha20	328562,14 ± 31,95	245,76 ± 1,78	9923376,79 ± 567520,10	362,68 ± 1371,23	134,37 ± 10,40	0,241	0,052
		AES128	4538532,35 ± 720,12	1127,87 ± 5,99	1017741,15 ± 593570,26	1614,42 ± 2242,46	165,41 ± 12,58	0,238	0,052
		AES192	4538642,86 ± 785,22	1284,90 ± 5,52	1012657,58 ± 561245,13	1682,99 ± 3479,68	163,40 ± 8,24	0,240	0,052
P-521	Curve25519	AES256	4538579,54 ± 703,47	1466,50 ± 2,99	1012370,59 ± 562113,09	1544,99 ± 1618,86	154,17 ± 8,51	0,238	0,052
		ChaCha20	4538672,72 ± 657,60	235,12 ± 0,76	1000746,16 ± 551824,87	1640,99 ± 3442,16	141,56 ± 8,35	0,238	0,052
		AES128	52469,85 ± 185,16	181,69 ± 7,93	1021560,08 ± 552853,00	355,90 ± 1115,86	151,17 ± 7,86	0,122	0,026
		AES192	52473,80 ± 303,66	185,20 ± 8,95	1013823,71 ± 560445,54	380,09 ± 1419,94	155,85 ± 9,89	0,119	0,026
		AES256	52472,28 ± 302,33	188,19 ± 7,92	1058468,20 ± 853065,40	377,14 ± 1408,45	152,67 ± 9,54	0,116	0,026
		ChaCha20	52459,38 ± 236,85	99,66 ± 7,03	1022617,20 ± 560080,61	376,90 ± 1394,94	137,71 ± 12,70	0,121	0,027
ESP32	Curve25519	AES128	676896,17 ± 283,98	157,65 ± 5,62	1032991,16 ± 549276,57	1548,22 ± 1544,35	157,77 ± 18,02	0,135	0,030
		AES192	676901,10 ± 349,02	160,59 ± 4,78	1024080,45 ± 557776,50	1542,37 ± 1481,72	156,05 ± 11,56	0,134	0,029
		AES256	676897,11 ± 324,67	163,66 ± 4,69	1045246,19 ± 841745,54	1533,76 ± 1425,47	156,80 ± 11,35	0,136	0,029
		ChaCha20	676918,75 ± 273,26	76,13 ± 4,00	1021810,64 ± 557622,40	1558,35 ± 1544,31	148,77 ± 11,81	0,133	0,029
		AES128	10330,28 ± 864,52	3189,96 ± 910,34	1090885,59 ± 517115,84	392,37 ± 1909,78	155,34 ± 8,99	1,765	0,378
		AES192	10373,21 ± 871,76	3029,81 ± 833,22	1097316,15 ± 514032,84	350,36 ± 1311,35	156,87 ± 9,12	1,776	0,381
RPI 1	Curve25519	AES256	10205,97 ± 849,14	3080,56 ± 884,30	1100755,83 ± 508087,65	349,83 ± 1348,58	155,06 ± 9,20	1,773	0,379
		ChaCha20	10073,53 ± 890,71	2773,06 ± 837,46	1091973,55 ± 512154,12	369,61 ± 1347,99	144,49 ± 13,54	1,774	0,379
		AES128	270039,54 ± 4224,22	2996,39 ± 665,06	1035805,84 ± 521418,35	1855,33 ± 1270,63	155,99 ± 15,92	1,778	0,383
		AES192	269208,55 ± 2699,69	3139,63 ± 698,39	1041563,19 ± 525724,66	1590,42 ± 1280,90	154,05 ± 12,94	1,782	0,383
		AES256	268270,95 ± 2837,70	2939,13 ± 489,26	1032561,03 ± 525718,10	1563,79 ± 1260,25	156,11 ± 14,53	1,788	0,383
		ChaCha20	269403,18 ± 2615,00	2754,02 ± 755,38	1029914,41 ± 521869,06	1600,68 ± 2029,12	143,38 ± 13,38	1,785	0,383
RPI 2	Curve25519	AES128	5308,18 ± 408,47	1269,44 ± 234,67	1027858,12 ± 554080,67	364,06 ± 1835,65	157,67 ± 9,95	1,118	0,239
		AES192	5261,29 ± 407,49	1246,49 ± 243,91	1038463,43 ± 567412,46	349,92 ± 1283,08	155,68 ± 9,39	1,123	0,240
		AES256	5244,20 ± 415,64	1246,87 ± 225,10	1028524,72 ± 573442,20	350,77 ± 1403,18	155,82 ± 12,76	1,117	0,240
		ChaCha20	5271,91 ± 435,08	1072,85 ± 174,18	1034310,65 ± 570468,54	349,47 ± 1281,43	143,09 ± 15,17	1,121	0,240
		AES128	167443,44 ± 10162,30	942,94 ± 82,79	990104,56 ± 569967,35	1553,42 ± 1286,20	156,63 ± 20,27	1,128	0,242
		AES192	167841,92 ± 10125,03	990,49 ± 137,74	1003211,73 ± 569932,40	1584,25 ± 1340,26	154,63 ± 17,81	1,131	0,241
RPI 3	Curve25519	AES256	167461,08 ± 9895,18	972,11 ± 134,25	994980,04 ± 561239,06	1592,87 ± 1350,18	153,18 ± 12,45	1,132	0,242
		ChaCha20	167520,59 ± 9743,76	843,94 ± 111,02	996866,58 ± 567634,20	1583,58 ± 2151,70	143,31 ± 15,01	1,134	0,244
		AES128	4212,93 ± 390,06	874,43 ± 160,00	1043848,93 ± 571232,40	367,42 ± 1382,54	166,91 ± 13,02	0,946	0,202
		AES192	4200,14 ± 385,97	883,54 ± 152,65	1030993,07 ± 565133,03	373,24 ± 1423,73	172,06 ± 27,40	0,943	0,201
		AES256	4162,91 ± 372,77	862,46 ± 134,56	1042588,78 ± 576387,43	356,93 ± 1389,07	164,56 ± 15,75	0,943	0,201
		ChaCha20	4182,29 ± 382,24	764,38 ± 117,99	1034179,29 ± 573235,91	340,63 ± 924,46	150,34 ± 11,06	0,938	0,200
P-521	Curve25519	AES128	114047,24 ± 10663,40	606,61 ± 116,23	1050853,72 ± 571558,64	1622,43 ± 1480,62	167,12 ± 22,20	0,963	0,206
		AES192	114289,52 ± 10691,42	622,13 ± 120,41	1056427,11 ± 569460,77	1616,82 ± 1412,84	165,56 ± 13,09	0,971	0,206
		AES256	114365,56 ± 10037,19	613,38 ± 116,22	1062689,38 ± 563629,00	1615,12 ± 1438,33	165,96 ± 16,81	0,966	0,207
		ChaCha20	114915,53 ± 9528,70	541,87 ± 105,37	1067784,71 ± 564839,31	1612,49 ± 1485,99	150,01 ± 13,64	0,967	0,207

Table C.1: Results from the experiment HTTP.

Device	Cypher	Encrypt (μs)	Transport (μs)	Decrypt (μs)	Wh	Ah
ESP8266	AES128	1099,02 \pm 15,69	588079,85 \pm 447649,23	175,54 \pm 11,72	0,233	0,051
	AES192	1265,59 \pm 15,66	497408,47 \pm 471691,53	170,90 \pm 15,19	0,235	0,051
	AES256	1454,65 \pm 16,31	666822,48 \pm 436653,25	174,74 \pm 12,27	0,243	0,053
	ChaCha20	261,96 \pm 4,30	770039,26 \pm 386658,71	157,80 \pm 13,05	0,243	0,053
ESP32	AES128	141,34 \pm 6,80	471233,51 \pm 490600,43	176,90 \pm 19,10	0,107	0,023
	AES192	144,82 \pm 7,74	816345,09 \pm 370775,30	178,18 \pm 16,32	0,109	0,024
	AES256	147,23 \pm 7,68	797144,29 \pm 368438,65	178,16 \pm 23,00	0,110	0,024
	ChaCha20	69,15 \pm 5,81	965871,42 \pm 97463,14	160,02 \pm 16,10	0,107	0,024
RPI 1	AES128	2642,76 \pm 231,96	335339,04 \pm 465086,92	174,43 \pm 14,28	1,772	0,379
	AES192	2669,88 \pm 204,42	563632,35 \pm 477824,09	170,77 \pm 13,68	1,759	0,376
	AES256	2686,75 \pm 199,34	21188,47 \pm 3214,50	169,40 \pm 12,55	1,766	0,378
	ChaCha20	2410,02 \pm 198,65	18266,81 \pm 3698,13	156,93 \pm 11,19	1,772	0,379
RPI 2	AES128	1150,27 \pm 50,30	13507,98 \pm 57757,64	171,13 \pm 12,67	1,093	0,234
	AES192	1180,06 \pm 48,32	16632,30 \pm 28281,20	172,57 \pm 12,71	1,103	0,236
	AES256	1169,76 \pm 46,22	34044,58 \pm 161167,64	174,06 \pm 12,98	1,103	0,236
	ChaCha20	1055,62 \pm 66,21	11193,80 \pm 28012,58	156,55 \pm 10,34	1,104	0,237
RPI 3	AES128	876,62 \pm 17,13	21084,11 \pm 24397,87	169,12 \pm 11,28	0,946	0,203
	AES192	882,09 \pm 17,74	20193,67 \pm 14495,71	172,12 \pm 9,24	0,949	0,203
	AES256	895,93 \pm 17,76	16769,77 \pm 5991,42	167,77 \pm 12,88	0,944	0,202
	ChaCha20	794,51 \pm 17,28	19890,03 \pm 32270,03	154,18 \pm 11,96	0,946	0,203

Table C.2: Results from the experiment MQTT.

References

- [1] M. F. Elrawy, A. I. Awad, and H. F. Hamed, "Intrusion detection systems for iot-based smart environments: A survey," *Journal of Cloud Computing*, vol. 7, no. 1, pp. 1–20, 2018.
- [2] A. Sivanathan, H. H. Gharakheili, F. Loi, A. Radford, C. Wijenayake, A. Vishwanath, and V. Sivaraman, "Classifying iot devices in smart environments using network traffic characteristics," *IEEE Transactions on Mobile Computing*, vol. 18, no. 8, pp. 1745–1759, 2019. DOI: 10.1109/TMC.2018.2866249.
- [3] J. Wang, M. K. Lim, C. Wang, and M.-L. Tseng, "The evolution of the internet of things (iot) over the past 20 years," *Computers & Industrial Engineering*, vol. 155, p. 107174, 2021.
- [4] R. Sethi, B. Bhushan, N. Sharma, R. Kumar, and I. Kaushik, "Applicability of industrial iot in diversified sectors: Evolution, applications and challenges," in *Multimedia Technologies in the Internet of Things Environment*, Springer, 2021, pp. 45–67.
- [5] A. Alshamsi, Y. Anwar, M. Almulla, M. Aldohoori, N. Hamad, and M. Awad, "Monitoring pollution: Applying iot to create a smart environment," in *2017 International Conference on Electrical and Computing Technologies and Applications (ICECTA)*, 2017, pp. 1–4. DOI: 10.1109/ICECTA.2017.8251998.
- [6] C. Slobogin, "Public privacy: Camera surveillance of public places and the right to anonymity," *Miss. LJ*, vol. 72, p. 213, 2002.
- [7] V. Jesus, C. Silva, J. P. Barraca, G. Rosner, A. Nehme, M. Waqas, and R. L. Aguiar, "Permission and privacy challenges in alternate-tenant smart spaces," in *Open Identity Summit 2021*, H. RoSSnagel, C. H. Schunck, and S. Mödersheim, Eds., Bonn: Gesellschaft für Informatik e.V., 2021, pp. 217–222.
- [8] C. Silva, J. P. Barraca, and R. L. Aguiar, "Esim suitability for 5g and b5g enabled iot verticals," in *2021 8th International Conference on Future Internet of Things and Cloud (FiCloud)*, IEEE, 2021.
- [9] C. Silva, V. A. Cunha, J. P. Barraca, and R. L. Aguiar, "Hands-on evaluation of the cryptographic overhead on wireless sensor networks," in *2021 8th International Conference on Future Internet of Things and Cloud (FiCloud)*, IEEE, 2021.
- [10] G. Rosner and E. Kenneally, "Clearly opaque: Privacy risks of the internet of things," in *Rosner, Gilad and Kenneally, Erin, Clearly Opaque: Privacy Risks of the Internet of Things (May 1, 2018). IoT Privacy Forum*, 2018.
- [11] P. Bridges, "Eu and us privacy experts in search of transatlantic privacy solutions," *Amsterdam/Cambridge, September*, 2015.
- [12] M. Langheinrich, "Privacy in ubiquitous computing," in *Ubiquitous Computing*, CRC Press Boca Raton, FL, 2009, pp. 95–160.
- [13] A. Cavoukian, "Privacy by design," 2009.
- [14] P. Schaar, "Privacy by design," *Identity in the Information Society*, vol. 3, no. 2, pp. 267–274, 2010.
- [15] European Parliament and of the Council, "Directive 95/46/ec," Tech. Rep., 1995.
- [16] I. I. of Communications, "Personal data management: The user's perspective," 2012.
- [17] R. Gellert and S. Gutwirth, "The legal construction of privacy and data protection," *Computer Law & Security Review*, vol. 29, no. 5, pp. 522–530, 2013, ISSN: 0267-3649. DOI: <https://doi.org/10.1016/>

j.clsr.2013.07.005. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0267364913001325>.

- [18] E. Politou, E. Alepis, and C. Patsakis, “Forgetting personal data and revoking consent under the gdpr: Challenges and proposed solutions,” *Journal of Cybersecurity*, vol. 4, no. 1, ty001, 2018.
- [19] J. B. Earp, A. I. Anton, L. Aiman-Smith, and W. H. Stufflebeam, “Examining internet privacy policies within the context of user privacy values,” *IEEE Transactions on Engineering Management*, vol. 52, no. 2, pp. 227–237, 2005. DOI: 10.1109/TEM.2005.844927.
- [20] G. Rosner and E. Kenneally, “Privacy and the internet of things: Emerging frameworks for policy and design,” in *Rosner, Gilad and Kenneally, Erin, Privacy and the Internet of Things: Emerging Frameworks for Policy and Design (June 7, 2018). UC Berkeley Center for Long-Term Cybersecurity/Internet of Things Privacy Forum*, 2018.
- [21] V. Sivaraman, H. H. Gharakheili, A. Vishwanath, R. Boreli, and O. Mehani, “Network-level security and privacy control for smart-home iot devices,” in *2015 IEEE 11th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*, 2015, pp. 163–167. DOI: 10.1109/WiMOB.2015.7347956.
- [22] B. Könings and F. Schaub, “Territorial privacy in ubiquitous computing,” in *2011 Eighth International Conference on Wireless On-Demand Network Systems and Services*, IEEE, 2011, pp. 104–108.
- [23] B. Könings, F. Schaub, M. Weber, and F. Kargl, “Towards territorial privacy in smart environments,” in *AAAI 2010 Spring Symposium*, Universität Ulm, 2010.
- [24] M. Sobolewski, J. Mazur, and M. Paliski, “Gdpr: A step towards a user-centric internet?” *Intereconomics*, vol. 52, no. 4, pp. 207–213, 2017.
- [25] B. Priem, E. Kosta, A. Kuczerawy, J. Dumortier, and R. Leenes, “User-centric privacy-enhancing identity management,” in *Digital Privacy*, Springer, 2011, pp. 91–106.
- [26] M. M. Hassan Onik, N. Al-Zaben, J. Yang, N. Lee, and C. Kim, “Risk identification of personally identifiable information from collective mobile app data,” in *2018 International Conference on Computing, Electronics Communications Engineering (iCCECE)*, 2018, pp. 71–76. DOI: 10.1109/iCCECOME.2018.8659213.
- [27] C. Romero-Tris and D. Megías, “Protecting privacy in trajectories with a user-centric approach,” *ACM Trans. Knowl. Discov. Data*, vol. 12, no. 6, Aug. 2018, ISSN: 1556-4681. DOI: 10.1145/3233185. [Online]. Available: <https://doi.org/10.1145/3233185>.
- [28] B. Bordel, R. Alcarria, T. Robles, and M. S. Iglesias, “Data authentication and anonymization in iot scenarios and future 5g networks using chaotic digital watermarking,” *IEEE Access*, vol. 9, pp. 22 378–22 398, 2021. DOI: 10.1109/ACCESS.2021.3055771.
- [29] K. Rantos, G. Drosatos, K. Demertzis, C. Ilioudis, A. Papanikolaou, and A. Kritsas, “Advocate: A consent management platform for personal data processing in the iot using blockchain technology,” in *International Conference on Security for Information Technology and Communications*, Springer, 2018, pp. 300–313.
- [30] V. Jesus, “Towards an accountable web of personal information: The web-of-receipts,” *IEEE Access*, vol. 8, pp. 25 383–25 394, 2020. DOI: 10.1109/ACCESS.2020.2970270.
- [31] E. Maler, “Extending the power of consent with user-managed access: A standard architecture for asynchronous, centralizable, internet-scalable consent,” in *2015 IEEE Security and Privacy Workshops*, 2015, pp. 175–179. DOI: 10.1109/SPW.2015.34.
- [32] E. Maler, “Extending the power of consent with user-managed access: A standard architecture for asynchronous, centralizable, internet-scalable consent,” in *2015 IEEE Security and Privacy Workshops*, IEEE, 2015, pp. 175–179.
- [33] C. Bartolini, S. Daoudagh, G. Lenzini, and E. Marchetti, “Gdpr-based user stories in the access control perspective,” in *International Conference on the Quality of Information and Communications Technology*, Springer, 2019, pp. 3–17.

- [34] S. Mandal, B. Bera, A. K. Sutrala, A. K. Das, K. R. Choo, and Y. Park, "Certificateless-signcryption-based three-factor user access control scheme for iot environment," *IEEE Internet of Things Journal*, vol. 7, no. 4, pp. 3184–3197, 2020. DOI: 10.1109/JIOT.2020.2966242.
- [35] J. Qiu, Z. Tian, C. Du, Q. Zuo, S. Su, and B. Fang, "A survey on access control in the age of internet of things," *IEEE Internet of Things Journal*, vol. 7, no. 6, pp. 4682–4696, 2020. DOI: 10.1109/JIOT.2020.2969326.
- [36] Y. Zhang, S. Kasahara, Y. Shen, X. Jiang, and J. Wan, "Smart contract-based access control for the internet of things," *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 1594–1605, 2019. DOI: 10.1109/JIOT.2018.2847705.
- [37] N. Ahmed and C. D. Jensen, "A mechanism for identity delegation at authentication level," in *Identity and Privacy in the Internet Age*, Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 148–162, ISBN: 978-3-642-04766-4.
- [38] L. Zhang, G.-J. Ahn, and B.-T. Chu, "A rule-based framework for role-based delegation and revocation," *ACM Trans. Inf. Syst. Secur.*, vol. 6, no. 3, pp. 404–441, Aug. 2003, ISSN: 1094-9224. DOI: 10.1145/937527.937530. [Online]. Available: <https://doi.org/10.1145/937527.937530>.
- [39] F. B. Schneider, "Least privilege and more [computer security]," *IEEE Security Privacy*, vol. 1, no. 5, pp. 55–59, 2003. DOI: 10.1109/MSECP.2003.1236236.
- [40] N. Tapas, G. Merlino, and F. Longo, "Blockchain-based iot-cloud authorization and delegation," in *2018 IEEE International Conference on Smart Computing (SMARTCOMP)*, 2018, pp. 411–416. DOI: 10.1109/SMARTCOMP.2018.00038.
- [41] P. J. Lu, L. Yeh, and J. Huang, "An privacy-preserving cross-organizational authentication/authorization/accounting system using blockchain technology," in *2018 IEEE International Conference on Communications (ICC)*, 2018, pp. 1–6. DOI: 10.1109/ICC.2018.8422733.
- [42] G. Ali, N. Ahmad, Y. Cao, M. Asif, H. Cruickshank, and Q. E. Ali, "Blockchain based permission delegation and access control in internet of things (baci)," *Computers & Security*, vol. 86, pp. 318–334, 2019, ISSN: 0167-4048. DOI: <https://doi.org/10.1016/j.cose.2019.06.010>. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0167404819301208>.
- [43] L. Chettri and R. Bera, "A comprehensive survey on internet of things (iot) toward 5g wireless systems," *IEEE Internet of Things Journal*, vol. 7, no. 1, pp. 16–32, 2020. DOI: 10.1109/JIOT.2019.2948888.
- [44] L. Atzori, A. Iera, and G. Morabito, "The internet of things: A survey," *Computer Networks*, vol. 54, no. 15, pp. 2787–2805, 2010, ISSN: 1389-1286. DOI: <https://doi.org/10.1016/j.comnet.2010.05.010>. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S1389128610001568>.
- [45] S. Madakam, V. Lake, V. Lake, V. Lake, *et al.*, "Internet of things (iot): A literature review," *Journal of Computer and Communications*, vol. 3, no. 05, p. 164, 2015.
- [46] C. Gomez, S. Chessa, A. Fleury, G. Roussos, and D. Preuveneers, "Internet of things for enabling smart environments: A technology-centric perspective," *Journal of Ambient Intelligence and Smart Environments*, vol. 11, no. 1, pp. 23–43, 2019.
- [47] A. Tiwary, M. Mahato, A. Chidar, M. K. Chandrol, M. Shrivastava, and M. Tripathi, "Internet of things (iot): Research, architectures and applications," *International Journal on Future Revolution in Computer Science & Communication Engineering*, vol. 4, no. 3, pp. 23–27, 2018.
- [48] S. Deep, X. Zheng, A. Jolfaei, D. Yu, P. Ostovari, and A. Kashif Bashir, "A survey of security and privacy issues in the internet of things from the layered context," *Transactions on Emerging Telecommunications Technologies*, e3935, 2020.
- [49] Policy, R. Group, *et al.*, "The internet of things: An introduction to privacy issues with a focus on the retail and home environments," *Office of the Privacy Commissioner of Canada*, Feb, 2016.
- [50] S. Li, L. Da Xu, and S. Zhao, "The internet of things: A survey," *Information Systems Frontiers*, vol. 17, no. 2, pp. 243–259, 2015.
- [51] S. Cirani and M. Picone, "Effective authorization for the web of things," in *2015 IEEE 2nd World Forum on Internet of Things (WF-IoT)*, IEEE, 2015, pp. 316–320.

- [52] I. Kounelis, G. Baldini, R. Neisse, G. Steri, M. Tallacchini, and A. Guimaraes Pereira, "Building trust in the human?internet of things relationship," *IEEE Technology and Society Magazine*, vol. 33, no. 4, pp. 73–80, 2014. DOI: 10.1109/MTS.2014.2364020.
- [53] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of things (iot): A vision, architectural elements, and future directions," *Future generation computer systems*, vol. 29, no. 7, pp. 1645–1660, 2013.
- [54] S. H. Shah and I. Yaqoob, "A survey: Internet of things (iot) technologies, applications and challenges," in *2016 IEEE Smart Energy Grid Engineering (SEGE)*, 2016, pp. 381–385. DOI: 10.1109/SEGE.2016.7589556.
- [55] X. Jia, Q. Feng, T. Fan, and Q. Lei, "Rfid technology and its applications in internet of things (iot)," in *2012 2nd International Conference on Consumer Electronics, Communications and Networks (CECNet)*, 2012, pp. 1282–1285. DOI: 10.1109/CECNet.2012.6201508.
- [56] S. Al-Sarawi, M. Anbar, K. Alieyan, and M. Alzubaidi, "Internet of things (iot) communication protocols: Review," in *2017 8th International Conference on Information Technology (ICIT)*, 2017, pp. 685–690. DOI: 10.1109/ICITECH.2017.8079928.
- [57] F. John Dian, R. Vahidnia, and A. Rahmati, "Wearables and the internet of things (iot), applications, opportunities, and challenges: A survey," *IEEE Access*, vol. 8, pp. 69 200–69 211, 2020. DOI: 10.1109/ACCESS.2020.2986329.
- [58] A. Jain, P. Tanwar, and S. Mehra, "Home automation system using internet of things (iot)," in *2019 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COMITCon)*, IEEE, 2019, pp. 300–305.
- [59] T. Alam, A. A. Salem, A. O. Alsharif, and A. M. Alhujaili, "Smart home automation towards the development of smart cities," *Tanweer Alam. Abdulrahman A. Salem. Ahmad O. Alsharif. Abdulaziz M. Alhujaili. " Smart Home Automation Towards the Development of Smart Cities. ", Computer Science and Information Technologies*, vol. 1, no. 1, 2020.
- [60] D. Marikyan, S. Papagiannidis, and E. Alamanos, "A systematic review of the smart home literature: A user perspective," *Technological Forecasting and Social Change*, vol. 138, pp. 139–154, 2019, ISSN: 0040-1625. DOI: <https://doi.org/10.1016/j.techfore.2018.08.015>. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0040162517315676>.
- [61] M. Sundarramurthi, A. A. Sundari, and A. Giridharan, "A method of designing home automation control system (hacs) using virtual assistant and mobile application," in *2019 International Conference on contemporary Computing and Informatics (IC3I)*, IEEE, 2019, pp. 5–8.
- [62] N. Apthorpe, D. Y. Huang, D. Reisman, A. Narayanan, and N. Feamster, "Keeping the smart home private with smart (er) iot traffic shaping," *Proceedings on Privacy Enhancing Technologies*, vol. 2019, no. 3, pp. 128–148, 2019.
- [63] J. Bernd, R. Abu-Salma, and A. Frik, "Bystanders privacy: The perspectives of nannies on smart home surveillance," in *10th {USENIX} Workshop on Free and Open Communications on the Internet ({FOCI} 20)*, 2020.
- [64] N. Guhr, O. Werth, P. P. H. Blacha, and M. H. Breitner, "Privacy concerns in the smart home context," *SN Applied Sciences*, vol. 2, no. 2, pp. 1–12, 2020.
- [65] A. Benlian, J. Klumpe, and O. Hinz, "Mitigating the intrusive effects of smart home assistants by using anthropomorphic design features: A multimethod investigation," *Information Systems Journal*, vol. 30, no. 6, pp. 1010–1042, 2020.
- [66] S. GRAY, "Always on: Privacy implications of microphone-enabled devices," in *FUTURE OF PRIVACY FORUM*, 2016.
- [67] C. Watchdog, *Google, amazon patent filings reveal digital home assistant privacy problems*, 2017.
- [68] D. Bylieva, Z. Bekirogullari, V. Lobatyuk, and N. Anosova, "Home assistant of the future: What is it like?" In *Proceedings of the International Scientific Conference-Digital Transformation on Manufacturing, Infrastructure and Service*, 2020, pp. 1–8.

- [69] N. Abdi, K. M. Ramokapane, and J. M. Such, “More than smart speakers: Security and privacy perceptions of smart home personal assistants,” in *Fifteenth Symposium on Usable Privacy and Security ({SOUPS} 2019)*, 2019, pp. 451–466.
- [70] I.-I. Ptru, M. Caraba, M. Brbulescu, and L. Gheorghe, “Smart home iot system,” in *2016 15th RoEduNet Conference: Networking in Education and Research*, 2016, pp. 1–6. DOI: 10.1109/RoEduNet.2016.7753232.
- [71] M. Silverio-Fernández, S. Renukappa, and S. Suresh, “What is a smart device?-a conceptualisation within the paradigm of the internet of things,” *Visualization in Engineering*, vol. 6, no. 1, pp. 1–10, 2018.
- [72] C. Geeng and F. Roesner, “Who’s in control? interactions in multi-user smart homes,” in *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, 2019, pp. 1–13.
- [73] W. M. Kang, S. Y. Moon, and J. H. Park, “An enhanced security framework for home appliances in smart home,” *Human-centric Computing and Information Sciences*, vol. 7, no. 1, pp. 1–12, 2017.
- [74] P. Mtshali and F. Khubisa, “A smart home appliance control system for physically disabled people,” in *2019 Conference on Information Communications Technology and Society (ICTAS)*, IEEE, 2019, pp. 1–5.
- [75] S. Zheng, N. Apthorpe, M. Chetty, and N. Feamster, “User perceptions of smart home iot privacy,” *Proceedings of the ACM on Human-Computer Interaction*, vol. 2, no. CSCW, pp. 1–20, 2018.
- [76] Y. Yao, J. R. Basdeo, S. Kaushik, and Y. Wang, “Defending my castle: A co-design study of privacy mechanisms for smart homes,” in *Proceedings of the 2019 chi conference on human factors in computing systems*, 2019, pp. 1–12.
- [77] G. S. Poh, P. Gope, and J. Ning, “Privhome: Privacy-preserving authenticated communication in smart home environment,” *IEEE Transactions on Dependable and Secure Computing*, vol. 18, no. 3, pp. 1095–1107, 2021. DOI: 10.1109/TDSC.2019.2914911.
- [78] A. Chakravorty, T. Wlodarczyk, and C. Rong, “Privacy preserving data analytics for smart homes,” in *2013 IEEE Security and Privacy Workshops*, 2013, pp. 23–27. DOI: 10.1109/SPW.2013.22.
- [79] R. Chowdhury, H. Ould-Slimane, C. Talhi, and M. Cheriet, “Attribute-based encryption for preserving smart home data privacy,” in *International conference on smart homes and health telematics*, Springer, 2017, pp. 185–197.
- [80] T. L. N. Dang and M. S. Nguyen, “An approach to data privacy in smart home using blockchain technology,” in *2018 International Conference on Advanced Computing and Applications (ACOMP)*, 2018, pp. 58–64. DOI: 10.1109/ACOMP.2018.00017.
- [81] T. Datta, N. Apthorpe, and N. Feamster, “A developer-friendly library for smart home iot privacy-preserving traffic obfuscation,” in *Proceedings of the 2018 Workshop on IoT Security and Privacy*, 2018, pp. 43–48.
- [82] K. Yoshigoe, W. Dai, M. Abramson, and A. Jacobs, “Overcoming invasion of privacy in smart home environment with synthetic packet injection,” in *2015 TRON Symposium (TRONSHOW)*, 2015, pp. 1–7. DOI: 10.1109/TRONSHOW.2014.7396875.
- [83] S. Moncrieff, S. Venkatesh, and G. West, “Dynamic privacy in a smart house environment,” in *2007 IEEE International Conference on Multimedia and Expo*, 2007, pp. 2034–2037. DOI: 10.1109/ICME.2007.4285080.
- [84] D. Geneiatakis, I. Kounelis, R. Neisse, I. Nai-Fovino, G. Steri, and G. Baldini, “Security and privacy issues for an iot based smart home,” in *2017 40th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, 2017, pp. 1292–1297. DOI: 10.23919/MIPRO.2017.7973622.
- [85] A. Cherkaoui, L. Bossuet, L. Seitz, G. Selander, and R. Borgaonkar, “New paradigms for access control in constrained environments,” in *2014 9th International Symposium on Reconfigurable and Communication-Centric Systems-on-Chip (ReCoSoC)*, 2014, pp. 1–4. DOI: 10.1109/ReCoSoC.2014.6861362.

- [86] M. de Reuver and J. Ondrus, "When technological superiority is not enough: The struggle to impose the sim card as the nfc secure element for mobile payment platforms," *Telecommunications Policy*, vol. 41, no. 4, pp. 253–262, 2017, ISSN: 0308-5961. DOI: <https://doi.org/10.1016/j.telpol.2017.01.004>.
- [87] B. A. Abdou, "Commercializing esim for network operators," in *2019 IEEE 5th World Forum on Internet of Things (WF-IoT)*, 2019, pp. 616–621. DOI: 10.1109/WF-IoT.2019.8767260.
- [88] S. Chitroub, D. Blaid, H. Aouadia, and R. Laouar, "Securing mobile iot deployment using embedded sim: Concerns and solutions," in *2019 International Conference on Internet of Things, Embedded Systems and Communications (IINTEC)*, 2019, pp. 75–79. DOI: 10.1109/IINTEC48298.2019.9112138.
- [89] G. Hatzivasilis, O. Soultatos, S. Ioannidis, C. Verikoukis, G. Demetriou, and C. Tsatsoulis, "Review of security and privacy for the internet of medical things (iomt)," in *2019 15th International Conference on Distributed Computing in Sensor Systems (DCOSS)*, 2019, pp. 457–464. DOI: 10.1109/DCOSS.2019.00091.
- [90] A. Ross and R. Nadgir, "A calibration model for fingerprint sensor interoperability," in *Biometric Technology for Human Identification III*, International Society for Optics and Photonics, vol. 6202, 2006, 62020B.
- [91] T. Van der Putte and J. Keuning, "Biometrical fingerprint recognition: Dont get your fingers burned," in *Smart Card Research and Advanced Applications*, Springer, 2000, pp. 289–303.
- [92] M. Gordon and S. Sankaranarayanan, "Biometric security mechanism in mobile paymentts," in *2010 Seventh International Conference on Wireless and Optical Communications Networks - (WOCN)*, 2010, pp. 1–6. DOI: 10.1109/WOCN.2010.5587318.
- [93] M. Butt, O. Henniger, A. Nouak, and A. Kuijper, "Privacy protection of biometric templates," in *International Conference on Human-Computer Interaction*, Springer, 2014, pp. 153–158.
- [94] I. Cerqueira, V. J. Sá, and S. T. de Magalhães, "Study of the perception on the portuguese citizen card and electronic signature," in *Global Security, Safety and Sustainability & e-Democracy*, Springer, 2011, pp. 164–170.
- [95] R. Santos, M. E. Correia, and L. Antunes, "Securing a health information system with a government issued digital identification card," in *2008 42nd Annual IEEE International Carnahan Conference on Security Technology*, 2008, pp. 135–141. DOI: 10.1109/CCST.2008.4751292.
- [96] S. Rinaldi, F. Bonafini, P. Ferrari, A. Flammini, E. Sisinni, and D. Bianchini, "Impact of data model on performance of time series database for internet of things applications," in *2019 IEEE International Instrumentation and Measurement Technology Conference (I2MTC)*, 2019, pp. 1–6. DOI: 10.1109/I2MTC.2019.8827164.
- [97] M. Jawad, P. S. Alvarado, and P. Valdúriez, "Design of priserv, a privacy service for dhTs," in *Proceedings of the 2008 international workshop on Privacy and anonymity in information society*, 2008, pp. 21–25.
- [98] H. Vyawahare, P. Karde, and V. Thakare, "A hybrid database approach using graph and relational database," in *2018 International Conference on Research in Intelligent and Computing in Engineering (RICE)*, 2018, pp. 1–4. DOI: 10.1109/RICE.2018.8509057.
- [99] Y. Rasheed, M. Qutqut, and F. Almasalha, "Overview of the current status of nosql database," *Int. J. Comput. Sci. Netw. Secur.*, vol. 19, no. 4, pp. 47–53, 2019.
- [100] I. Petre, R. Boncea, C. Z. Radulescu, A. Zamfiroiu, and I. Sandu, "A time-series database analysis based on a multiattribute maturity model," *Studies in Informatics and Control*, vol. 2, no. 2, pp. 177–188, 2019.
- [101] A. S. R. Guzmán, D. Valdeolmillos, A. Rivas, A. G. Arrieta, and P. Chamoso, "Creation of a distributed nosql database with distributed hash tables," in *International Conference on Hybrid Artificial Intelligence Systems*, Springer, 2019, pp. 26–37.
- [102] G. Fersi, W. Louati, and M. B. Jemaa, "Distributed hash table-based routing and data management in wireless sensor networks: A survey," *Wireless networks*, vol. 19, no. 2, pp. 219–236, 2013.
- [103] S. Anand, P. Singh, and B. Sagar, "Working with cassandra database," in *Information and Decision Sciences*, Springer, 2018, pp. 531–538.

- [104] S. Kalid, A. Syed, A. Mohammad, and M. N. Halgamuge, “Big-data nosql databases: A comparison and analysis of big-table, dynamodb, and cassandra,” in *2017 IEEE 2nd International Conference on Big Data Analysis (ICBDA)*, 2017, pp. 89–93. DOI: 10.1109/ICBDA.2017.8078782.
- [105] S. K. Pandey and Sudhakar, “Context based cassandra query language,” in *2017 8th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*, 2017, pp. 1–7. DOI: 10.1109/ICCCNT.2017.8204142.
- [106] M. R. Mufid, A. Basofi, M. U. H. Al Rasyid, I. F. Rochimansyah, and A. rokhim, “Design an mvc model using python for flask framework development,” in *2019 International Electronics Symposium (IES)*, 2019, pp. 214–219. DOI: 10.1109/ELECSYM.2019.8901656.
- [107] N. C. Narendra, S. Nayak, and A. Shukla, “Managing large-scale transient data in iot systems,” in *2018 10th International Conference on Communication Systems Networks (COMSNETS)*, 2018, pp. 565–568. DOI: 10.1109/COMSNETS.2018.8328274.
- [108] W. Zeller and E. W. Felten, “Cross-site request forgeries: Exploitation and prevention,” *The New York Times*, pp. 1–13, 2008.
- [109] C. Braz and J.-M. Robert, “Security and usability: The case of the user authentication methods,” in *Proceedings of the 18th Conference on l’Interaction Homme-Machine*, 2006, pp. 199–203.
- [110] J. Yick, B. Mukherjee, and D. Ghosal, “Wireless sensor network survey,” *Computer Networks*, vol. 52, no. 12, pp. 2292–2330, 2008, issn: 1389-1286. DOI: <https://doi.org/10.1016/j.comnet.2008.04.002>. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S1389128608001254>.
- [111] M. Kocakulak and I. Butun, “An overview of wireless sensor networks towards internet of things,” in *2017 IEEE 7th Annual Computing and Communication Workshop and Conference (CCWC)*, 2017, pp. 1–6. DOI: 10.1109/CCWC.2017.7868374.
- [112] H. Noura, A. Chehab, L. Sleem, M. Noura, R. Couturier, and M. M. Mansour, “One round cipher algorithm for multimedia IoT devices,” *Multimedia Tools and Applications*, vol. 77, no. 14, pp. 18 383–18 413, Jan. 2018. DOI: 10.1007/s11042-018-5660-y.
- [113] D. A. F. Saraiva, V. R. Q. Leithardt, D. de Paula, A. S. Mendes, G. V. González, and P. Crocker, “PRISEC: Comparison of symmetric key algorithms for IoT devices,” *Sensors*, vol. 19, no. 19, p. 4312, Oct. 2019. DOI: 10.3390/s19194312.
- [114] M. Abu-Tair, S. Djahel, P. Perry, B. Scotney, U. Zia, J. M. Carracedo, and A. Sajjad, “Towards secure and privacy-preserving IoT enabled smart home: Architecture and experimental study,” *Sensors*, vol. 20, no. 21, p. 6131, Oct. 2020. DOI: 10.3390/s20216131.
- [115] A. Diro, H. Reda, N. Chilamkurti, A. Mahmood, N. Zaman, and Y. Nam, “Lightweight authenticated-encryption scheme for internet of things based on publish-subscribe communication,” *IEEE Access*, vol. 8, pp. 60 539–60 551, 2020. DOI: 10.1109/access.2020.2983117.
- [116] U. Banerjee, A. Vashishtha, and M. Saxena, “Evaluation of the capabilities of wireshark as a tool for intrusion detection,” *International Journal of computer applications*, vol. 6, no. 7, pp. 1–5, 2010.
- [117] F. A. Alaba, M. Othman, I. A. T. Hashem, and F. Alotaibi, “Internet of things security: A survey,” *Journal of Network and Computer Applications*, vol. 88, pp. 10–28, Jun. 2017. DOI: 10.1016/j.jnca.2017.04.002.
- [118] L. Tawalbeh, F. Muheidat, M. Tawalbeh, and M. Quwaidar, “IoT privacy and security: Challenges and solutions,” *Applied Sciences*, vol. 10, no. 12, p. 4102, Jun. 2020. DOI: 10.3390/app10124102.
- [119] T. K. Goyal and V. Sahula, “Lightweight security algorithm for low power IoT devices,” in *2016 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, IEEE, Sep. 2016. DOI: 10.1109/icaccci.2016.7732296.