TÉCNICO LISBOA

SurePresence: Location Proofs for Wearable and Kiosk Devices

Miguel Cordeiro Francisco

Thesis to obtain the Master of Science Degree in

Information Systems and Computer Engineering

Supervisor(s): Prof. Miguel Filipe Leitão Pardal
Prof. Hugo Miguel Aleixo Albuquerque Nicolau

Examination Committee

Chairperson: Prof. Pedro Tiago Gonçalves Monteiro
Supervisor: Prof. Miguel Filipe Leitão Pardal
Member of the Committee: Prof. António Manuel Raminhos Cordeiro Grilo

November 2021

"Great things come from hard work and perseverance. No excuses" - Kobe Bryant

Acknowledgments

I would like to start by thanking my advisor Miguel Pardal for the opportunity to work with him and in the SureThing project. I am very grateful for his guidance and confidence in me not only for this document but for all the work we developed together. I am also very grateful for the guidance provided by Samih Eisah in all aspects regarding my work and for Hugo Nicolau for accepting to be my co-advisor on such short notice. His help was undoubtedly crucial and I am very grateful for the confidence he put up in me. I would like to extend my gratitude to all members of the SureThing project for working with me in a harmonious and coordinated way.

I am also grateful for all the help and support from my friends, with whom I shared my struggles and my bugs. They always motivated me and never let me down. They contributed to my positive mental health and their support cannot be ignored. I would like to thank Marta Ramalho for the incredible drawings of the threats storyboards for the user study.

Last, but certainly not least, I would like to thank my family. I am very grateful for the opportunity my parents gave me to achieve a higher education in the field I love. Their constant support on everything I did was unmeasurable and incomparable. I am very grateful for the help my brother did not give me throughout my academic course which made me grow as a developer, as an engineer, and most of all as a person.

This work was supported by national funds through Fundação para a Ciência e a Tecnologia (FCT) with reference UID/CEC/50021/2019 (INESC-ID) and through the project with reference PTDC/CCI-COM/31440/2017 (SureThing).

Resumo

O mundo está a ir para além dos telemóveis e do paradigma da computação móvel, com mais dispositivos e ligações de rede presentes na vida quotidiana, de maneiras cada vez mais discretas. *Smartwatches* ligam cada vez mais pessoas no mundo e permitem novas aplicações baseadas no rastreio da localização do utilizador, como a monitorização da sua saúde. Geralmente, *wearables* permitem aceder a mais serviços, que requerem informação confiável acerca da localização do utilizador. Se o resultado do serviço for valioso, o utilizador poderá querer escapar ao processo de verificação da localização por parte da aplicação. Este é o motivo de muitas aplicações dependentes dessa informação necessitarem de proteção contra ataques de falsificação de localização. Uma possível abordagem é a utilização de certificados de localização, algo ainda não integrado com tecnologia ubíqua até hoje.

Neste trabalho introduzimos *SurePresence*, um sistema que verifica a localização dos seus utilizadores através de *wearables* e *kiosks*, com foco na experiência de utilizador e na segurança e privacidade das comunicações. Implementámos o nosso sistema no contexto de consultas médicas, onde um paciente verifica a sua presença num consultório através de três novas técnicas de provas de localização. A nossa avaliação, que inclui um estudo com utilizadores e uma avaliação comparativa de métricas de desempenho do nosso protocolo que garante tais comunicações seguras mostram que o nosso sistema é possível num mundo real, disponibilizando provas de localização através de dispositivos limitados, enquanto garantindo a privacidade de informação sensível do utilizador em comunicações baseadas em *Bluetooth Low-Energy*.

Palavras-chave: Internet das Coisas, Sistemas de Certificação de Localização, Experiência de Utilizador, Segurança, Privacidade

Abstract

The world is moving beyond the smartphones and the mobile computing paradigm, with more devices and network connections present in everyday life in more inconspicuous ways. Wearable devices, like smartwatches, are further connecting people to the world. New applications are possible such as activity and location tracking that allows health monitoring. More generally, wearables allow more access to services, which for many of them, the location information needs to be trusted. If the service output is valuable, the user may be tempted to bypass the application location verification. This is why many location-based applications need protection against location spoofing. One approach is to use a location certification solution, but so far they have not been integrated with ubiquitous technology. In this work, we introduce SurePresence, a system that allows people to verify their location using wearable and kiosk devices integrated into their everyday life, presenting a seamless user experience. The communications between such constrained devices are secured by an application-layer protocol providing end-to-end security. We present a specific implementation of our system to support a medical appointment use case, where a patient verifies his presence at a medical office through three novel kiosk-based location proof techniques. Our evaluation, which included a user study and a comparative performance assessment of our implemented application and underlying protocols showed that our system is feasible, providing verifiable location proofs using constrained devices while assuring privacy of user sensitive information in Bluetooth Low-Energy exchanges and can be used in many other real-world use cases.

Keywords: Internet of Things, Location Certification Systems, User Experience, Security, Privacy

Contents

List of Tables

List of Figures

Chapter 1

Introduction

In recent years, we witnessed an increasing interest and usage of Internet-connected devices with the intent of providing more harmonious ways of life. These connected devices can gather sensed data, and this data can be used for more efficient processes and actuators that can be used to perform useful actions. These so-called *smart objects* can be combined and constitute the Internet of Things (IoT) [UHM11]. They can be used for multiple purposes, from simple alarm clocks connected to the Internet gathering and working with information about sleep schedules to sophisticated smart homes [AAM19]. This heterogeneity of usage is reflected in the heterogeneity of the underlying technology that manages heating, security, and other management chores. There is not a common technology in all IoT devices, and different purposes bring different software and hardware necessities.

Wearables are a type of IoT device often used in health monitoring applications and are heavily based on data collection through sensors. Kiosks, another type of IoT device, serve as the physical frontier between the consumer and the application, and their usability is very important. Users are already acquainted with the interaction with such devices, thanks to their use as smart interactive assistant [SCRH20b], voting booth [GUMS18] and library information points [KM18], just to name a few examples. Depending on the use case for such devices, the requirements may differ. Thanks to these types of devices, people are becoming even more connected to the world, extending the reach of smartphones in our daily lives, exploring new interacting ways with constrained devices. It opens a door for many types of applications that may benefit from short-range communications, feasible through Bluetooth Low-Energy, a pivotal network technology in the IoT world as it can be used to connect devices between themselves and including the smartphones of the users.

In this work, our focus is on *location-based applications* that need the trustable user location context to provide certain functionalities. Usually, these services need to know the location of

the user or the collection of past tracked locations and use evidence to allow for location proofs.

1.1 Location Proofs

Currently, many location-based mobile applications do not verify the given user location information because they do not have the means to do it or simply because they think it is unnecessary, making them vulnerable to location spoofing attacks. Or even if they do verify, it is not in an effortless and good experience, requiring multiple and complex actions from the user. Sometimes users have no actual reason to spoof, as they want to receive the service. However, malicious users can abuse this trust and attempt location spoofing or even denial-of-service attacks, to obtain exclusive or specific functionalities from valuable services that, otherwise would be inaccessible due to geographic restrictions.

One solution to prevent location spoofing is *location certification systems* [ZC11] [CCCDP13] which are essential enablers for secure and reliable location-based services (LBS) that protect and verify information about the location of users. The functionalities of these systems can be leveraged with the usage of IoT and constrained devices, to seamlessly produce and verify location proofs. Users may lose interest in systems that are too focused on security and that ignore user experience. Leveraging the usability of location certification systems has not been a concern for the past years, but a *usable security* solution is required.

Location certification systems can benefit beyond smartphones and smartwatches from location-fixed devices equipped with a user interface, like a kiosk, to verify the location of a user on the spot through physical and non-physical interactions. Non-physical interactions include short-range communications which can be realized through Bluetooth, just like in past location certification systems [WPZM16][ZC11], or Bluetooth Low-Energy (BLE) for constrained devices. BLE presents many security issues that compromise the confidentiality of the exchanged messages [WNK+20][ZWD+20][PBB+17][LZ19]. These design flaws are later described in Section 2.2.3 and were the motivation for the development of an application-layer protocol to provide message confidentiality and other security guarantees in communications between IoT devices, which is also a general-purpose outcome of this work.

This work was done in the context of the SureThing project [FP18][dSCRP20][**?**]. It allows the creation and validation of location certificates through Smartphones and IoT devices in all types of locations.

1.2 Contributions

Our contributions in this work are the following:

1. Development of SurePresence, a smartwatch and kiosk-based location certification system for smartcards and ubiquitous devices and expansion of the core data and utility libraries of the SureThing framework;

2. Implementation of SurePresence in a medical use case, where patients can verify their presence when attending a medical appointment that led to the development of three novel location proof techniques based on the interactions of three different devices with a kiosk device;

3. Assessment of the user experience provided by the novel location proof techniques through a user study;

4. Implementation of POSE, an end-to-end application-layer protocol to secure Bluetooth Low-Energy Protocol Buffer-based communications between constrained devices and its evaluation regarding its packet overhead, CPU usage, and processing time.

Overall, our work shows that location proofs are viable and useful in a real-world use case with the potential for many more.

1.3 Dissertation Outline

The remainder of the document is structured as follows. Chapter 2 presents related work in the fields of location certification systems and relevant use cases for location proofs, IoT technologies, including IoT devices, background theory on Bluetooth Low-Energy and respective vulnerabilities, application-layer protocols, and an overview on usable security. Chapter 3 describes the SurePresence system, its core framework, all aspects regarding its design, a specific implementation for a medical use case, including the used platforms, the novel kiosk-based location proof techniques implemented, and the details of the application-layer protocol, POSE, providing security to BLE communications. Chapter 4 presents the evaluation of the user experience provided by the novel location proof techniques through a user study and the evaluation of the performance of POSE regarding its packet overhead, CPU usage, and processing time. Finally, Chapter 5 presents the conclusion and future work.

Chapter 2

Background & Related Work

We start by showing work related to existing location certification systems and use cases for location proofs, in Section 2.1. Then, in Section 2.2, we summarize works that have been presented in the area of IoT, specifically related to two types of devices: Wearables and Kiosk devices. In Section 2.2.3 we detail some background on Bluetooth Low-Energy and its vulnerabilities, our chosen protocol to realize all necessary short-range communications. Motivated by these flaws, we give an overview of application-layer protocols and object security in the context of the IoT in Section 2.2.4. Finally, in Section 2.3 we show previous work related to Usable Security, one of the most important concepts regarding usability in secure systems. In the context of location proof techniques, usability must coexist in harmony with security, allowing the creation of secure systems against spoofing attacks with a human-centric focus, which can be leveraged with the usage of IoT technologies.

2.1 Location Certification Systems

Location certification systems provide reliable and verified information about the location context of a user. Location-sensitive applications need this information because they cannot rely only on GPS [HLR11] for example, to obtain a correct and legitimate location of a user.

In 2011, Zhu and Cao [ZC11] introduced APPLAUS, a witness system based on neighboring mobile devices, capable of providing location proofs for nearby provers, through Bluetooth communication. PROPS [GKRT14] an architecturally similar system developed three years later is also based on a witness system but provides stronger privacy mechanisms, discussed in the next section. Ferreira and Pardal [FP18] present another system that also relies on witnesses to verify the presence of the devices of other users along with three location proof techniques, including geographic and Wi-Fi/Beacon fingerprinting. Similarly, Alamleh and AlQahtani [AA20] show

us a cheat-proof system to verify and validate GPS-produced location data, also based on Wi-Fi access points existent on that location. However, both systems do not address user privacy. Nosouhi et al. [AA20] show us PASPORT, a system capable of producing and verifying location proofs without the need of fixed wireless infrastructure in that location and based on a witness model focused on the anonymization of users.

2.1.1 Location Certification Architecture

We can conclude the state-of-art location certification systems have converged to the generic architecture that can be seen in Figure 2.1 and includes the following entities:

- **Prover** - The user of the system trying to prove his presence at a location.
- **Verifier** - An entity that validates the location proof submitted by the prover.
- **Witness** - A neighboring user of the prover that provides location proof for him. Can also be a prover.
- **Certificate Authority** - Third-party entity responsible for authenticating the users of the system.

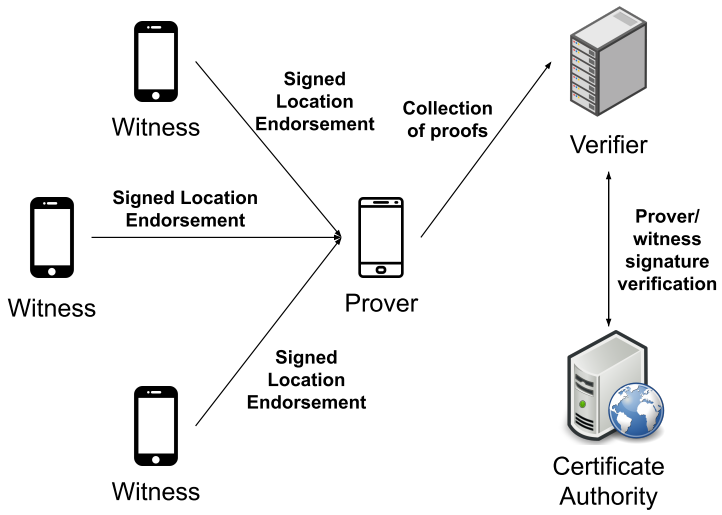


Figure 2.1: Location certification systems generic architecture.

2.1.2 User Privacy

Some location certification systems have also taken a step forward in providing privacy to the users, both witnesses, and provers. The APPLAUS system [ZC11] provides location privacy to its users, by using pseudonyms to identify each mobile device. A Certification Authority issues a certain set of private/public key pairs for each mobile device, where the public keys serve as pseudonyms. In PROPS [GKRT14] the location privacy of the user is guaranteed by hiding the digits of the GPS coordinates of the prover into hash chains. The prover chooses the granularity of its privacy (the revealed leftmost digits of his position) when providing this information to the witness. Witnesses never disclose their exact position because they never share their location and a possible eavesdropper only knows that they were in the vicinity of a prover. In PASPORT [NSY+20] it is proposed a private and secure distance bound mechanism, similar to [ABG+17], where LP request messages are anonymously broadcasted to ad-hoc witnesses.

2.1.3 Use of Location Proofs

Location proofs can be used in all types of sectors. In the workplace, can be used to confirm attendance in reunions or schools, for students to prove their attendance in classes or exams. A tourist can use location proofs to verify his presence in specific landmarks of a tourism trip. This concept can also be applied to festivalgoers when they visit different festival stands to receive small gifts. In the commercial sector, location proofs can measure the loyalty of a shop client, by verifying how many times a customer visits a shop and then rewarding him [SW09].

In Figure 2.2 we see different use cases for location proofs, from all types of sectors. A wearable device is proper to trace an itinerary while a kiosk device may be more suitable to verify a presence in a trusted location, like a medical office. The chosen device differs depending on the considered use cases and we will be focusing on two: Track Singular Locations and Trace an Itinerary. These represent the fundamental cases of rastreability: *Track* and *Trace*.

Track Singular Locations is an elementary use case related to a singular place when it is only needed to verify the presence of a person in one unique location. This represents the most simple case of location proofs: does not require verification and analysis of previous or future proofs to guarantee the integrity of this one. A good example of this use case is when a person shows up for a medical appointment. Although this example may depend on the duration of the presence, patients usually do not show up at the check-in and leave after, since they need to leave the insurance card with the secretary. The development of the SurePresence system, described in Chapter 3 will be focused on this use case.

Trace an Itinerary is a use case related to an itinerary as a whole proof. An itinerary

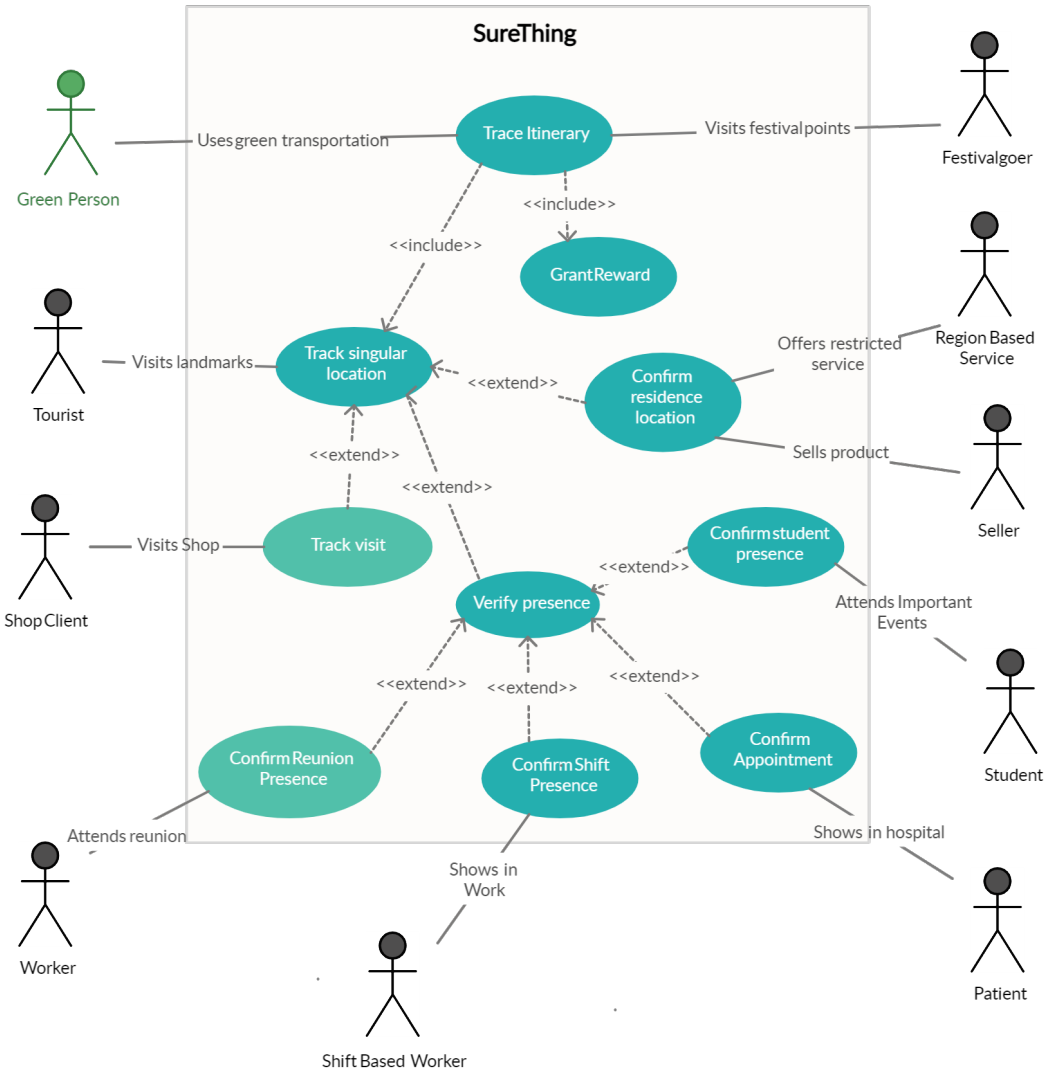


Figure 2.2: Use cases for location proofs.

represents a location chain, which is a composition of location points with a specific visiting order. Thus, the verification process of points of interest may be different depending on the location and the environment, but it still depends on the verification of previous location points. Smart tourism is a specific real-life use case where that verification process should be implemented. This is an important use case of the SureThing project, thanks to the CROSS system [**?**].

2.2 IoT Technologies

IoT devices benefit from emerging technologies but still present some security and privacy threats. Al-Fuqaha et al. [AGM+15] give us a complete summary of all related research work done in IoT. It is done as a survey covering all aspects of IoT including architecture, core elements, standards for IoT protocols, main challenges, and the following key ideas:

1. A five-layer architecture - Business, Application, Service Management, Object Abstraction, Objects - is the most applicable model for IoT applications, providing simplicity and abstraction. Can be seen in Figure 2.3.

2. There are six core elements, representing the building blocks, for the development of any IoT functionality: Identification, Sensing, Communication, Computation, Services, and Semantics.

3. Regarding the computation element, Android is the most complete operating system to be used in IoT environments.

4. There is no correct standardized protocol to be used in the different architectural layers because they depend on the scenario and functionality we want to provide.

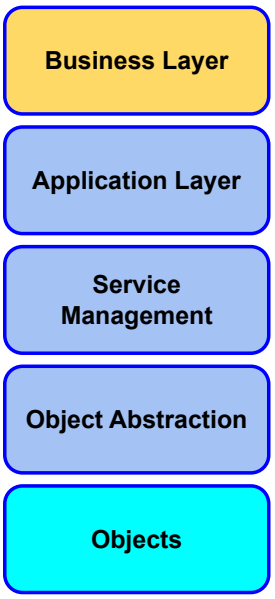


Figure 2.3: Five-layer IoT architecture. Adapted from [AGM+15].

Frustaci et al. [FPAF18] give us a summary of security issues relative to different protocols, respective vulnerabilities, and possible attacks in all three layers of the basic IoT model: perception, transportation, and application. Denial of Service (DoS) attacks constitute the biggest threat to IoT security since they can be done in all three layers. It is partially explained by the heterogeneity and complexity of IoT networks and the limited processing power of IoT devices.

2.2.1 Wearable Devices

Wearable devices constitute the interface between IoT systems and human users. They can collect and transmit data over the internet or even interconnect with other IoT devices in the

same area, like in a smart space. However, communication impacts their battery life.

Al-Sarawi et al. [AAAA17] review and study different communication protocols through different criteria including power consumption, range, network, topology, and cryptography. 6LoWPAN, ZigBee, BLE, ZWave, and NFC are protocols designed for portable devices since they offer low power consumption. 6LoWPAN and BLE have matured and have become the state-of-art wireless communication protocols for IoT devices, showing great potential in power demand, bit rate, and latency [TMTG13]. Although, practical results show that 6LoWPAN signals fade more quickly and the connection gets lost when obscured by an obstacle, making it much more vulnerable to obstacles than BLE. Still related to power consumption, Trappe et al. [THM15] show the types of threats low-end Internet-connected devices face and study the tradeoff between the energy needed to execute the core application and the energy needed to compute conventional cryptography.

Other hardware and software vulnerabilities in two wearable devices are shown in [AWHJ15]. The security of the software stack depends on the integrity of the hardware stack, which can be modified through a vulnerability in the design. To mitigate these problems, authentication and integrity checks of the running software when booting should be done. Fitbit, a tracking wearable, is also studied to identify its vulnerabilities related to security and privacy, and proper fixes are provided [ZP14]. These vulnerabilities include communication in plaintext, exposing sensitive tracking information, and lack of authentication with the webserver. Fixing these problems includes cryptography, which increases the power consumption of such devices.

2.2.2 Kiosk Devices

Interactive kiosks have been developed since 1977, with the purpose to provide access to information and other services, like Internet access or ticket selling platform. Nowadays, these systems show a similar overall architecture: the kiosk device itself and a database server that collects and stores information provided by the kiosk. The kiosk is based on three components: The kiosk hardware, software, and application [KM18].

Single Board Computers equipped with sensors, communication actuators, and security functionalities, like Raspberry PI and Arduino Yun, can be used to realize such IoT products [AGM+15] in a cost-effective way. In [GUMS18] it is shown how a Raspberry Pi can be used to make a low power consumption electronic voting kiosk prototype integrating biometric authentication techniques. Sad et al. [SCRH20a] show the deployment of an information kiosk in a university with the inclusion of technologies like machine learning for image processing and privacy mechanisms to prevent sensitive information disclosure. This kiosk can properly verify

the user and can change locations since it is a plug-and-play device.

2.2.3 Bluetooth Low-Energy

Bluetooth Low Energy (BLE) is a variation of the classic Bluetooth protocol with reduced power consumption and a similar communication range. It transmits data over 40 channels in the 2.4GHz unlicensed ISM frequency band. Like classic Bluetooth, two devices need to create a communication channel to transmit data to each other. The following concepts and terms represent the specification of how to send and read data among BLE devices.

Important Concepts

Generic Attribute (GATT) Profile The GATT profile establishes in detail the specification on how to exchange all profile and user data over a BLE channel. It is built on top of the Attribute Protocol (ATT).

Attribute Protocol (ATT) The Attribute Protocol (ATT) standardizes and optimizes the data access in BLE devices, through attributes. Each attribute contains data that can be read or written. It is represented by a 128-bit UUID and formatted into services or characteristics.

Profile A BLE profile is a specification for how a device works in a particular application by providing certain services and characteristics. An overview of the hierarchy of a BLE profile can be seen in Figure 2.4.

Service A GATT service groups conceptually related attributes (characteristics) in one common section of information. A service can be an application that may provide or ask for information through readable or writable characteristics, respectively.

Characteristic A GATT characteristic is a container for user data. It contains a single value section for user-defined data and multiple descriptors that describe and provide metadata about the actual value. It is also represented by a 128-bit UUID.

Descriptor The characteristic descriptor is the smallest structure of the Service hierarchy. It is an attribute that specifies all types of metadata information about the characteristic value, from a human-readable description to the type and range of the value.

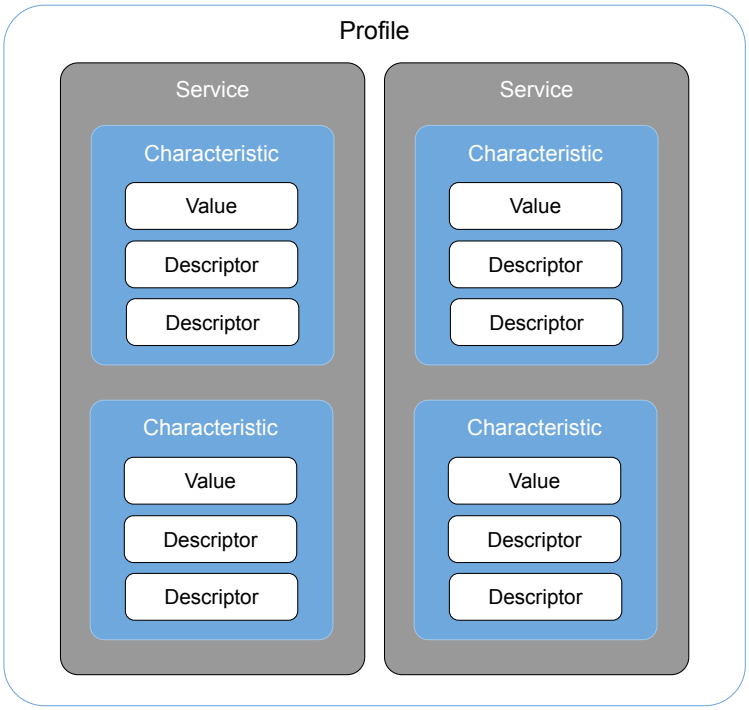


Figure 2.4: BLE GATT profile hierarchy.

Handle A handle is a unique 16-bit identifier for each addressable attribute by a BLE device. When a device makes services and characteristics available, it creates handles for each one, so that the connected device can access those attributes. The handle values are not predictable and the connected device must ask for the handles to discover them.

Roles

For two BLE devices to start interacting and exchanging data, they must assume specific roles, so that a connection can be successfully created. A device with the **Peripheral** role advertises that it is ready to receive a BLE connection. A device with the **Central** role scans for other devices that may be advertising. These roles are related to the discovery of devices. How the connection is established and how the two devices talk to each is dependent on the GATT role. The **GATT server** is the device that needs to send data (or make writable data available), while the **GATT client** is usually the device that needs to read data or receive updates. It is usually the device acting as the GATT server that advertises, with the peripheral role, while the device, with the central role, acts as the GATT client, but a client can still be peripheral.

Pairing

After establishing a connection, the BLE devices can start exchanging information. Without the pairing process, this information is transmitted in plaintext and can suffer multiple attacks.

The pairing process leverages that established connection into an encrypted connection, by authenticating both devices and distributing all the necessary keys. After this process, all the necessary information is stored in the devices, so that this process does not need to be repeated in future reconnections. This is the **bonding** process.

Bluetooth Low Energy 4.2 introduced LE Secure Connections, authenticated connections between BLE devices based on a Long Term Key (LTK), improving the previous LE Legacy pairing. The LTK is generated and exchanged between devices after the initial pairing procedure with the Temporary Key (TK), the Short Term Key (STK), and the ECDH public-key cryptography. The authentication of the connection depends on the pairing method. The pairing method depends on the agent of the BLE profile and the I/O capabilities of both devices. Figure 2.5 shows the resulting pairing method and resulting link security, depending on the I/O capabilities of both devices. The Initiator is the device that sends the pairing request while the Responder is the device accepting it or not.

	Initiator				
Responder	DisplayOnly	DisplayYesNo	KeyboardOnly	NoInputNoOutput	KeyboardDisplay
DisplayOnly	*Just Works* **Unauthenticated**	*Just Works* **Unauthenticated**	*Passkey Entry*: Initiator inputs, Responder displays **Authenticated**	*Just Works* **Unauthenticated**	*Passkey Entry*: Initiator inputs, Responder displays **Authenticated**
DisplayYesNo	*Just Works* **Unauthenticated**	*Just Works* (For LE Legacy Pairing) **Unauthenticated**	*Passkey Entry*: Initiator inputs, Responder displays **Authenticated**	*Just Works* **Unauthenticated**	*Passkey Entry* (For LE Legacy Pairing): Initiator inputs, Responder displays **Authenticated**
		Numeric Comparison (For LE Secure Connections) **Authenticated**			*Numeric Comparison* (For LE Secure Connections) **Authenticated**
KeyboardOnly	*Passkey Entry*: Initiator displays, Responder inputs **Authenticated**	*Passkey Entry*: Initiator displays, Responder inputs **Authenticated**	*Passkey Entry*: Initiator and, Responder inputs **Authenticated**	*Just Works* **Unauthenticated**	*Just Works* **Unauthenticated**
NoInput NoOutput	*Just Works* **Unauthenticated**	*Just Works* **Unauthenticated**	*Just Works* **Unauthenticated**	*Just Works* **Unauthenticated**	*Passkey Entry* (For LE Legacy Pairing): Initiator displays, Responder inputs **Authenticated**
Keyboard Display	*Passkey Entry*: Initiator displays, Responder inputs **Authenticated**	*Passkey Entry* (For LE Legacy Pairing): Initiator displays, Responder Inputs **Authenticated**	*Passkey Entry*: Initiator inputs, Responder displays **Authenticated**	*Just Works* **Unauthenticated**	*Numeric Comparison* (For LE Secure Connections) **Authenticated**
		Numeric Comparison (For LE Secure Connections) **Authenticated**			*Numeric Comparison* (For LE Secure Connections) **Authenticated**

Figure 2.5: BLE pairing methods based on the I/O capabilities of connected devices. Adapted from [Ren16].

There are four different pairing methods:

1. **Just Works** The devices initially exchange their public keys and then random nonces. This method gives more resilience to passive eavesdropping but does not protect the connection from Man-In-The-Middle (MITM) attacks.

2. **Out of Band (OOB)** Similar to the Just Works method but the values are exchanged via a different wireless technology.

3. **Passkey** The same 6 digit number is asked to be input into each connecting device. The two devices use this passkey, the public keys earlier exchanged and a nonce to authenticate the connection. This method is much more resilient to MITM attacks in the new LE Secure Connections than in the old LE Legacy Pairing.

4. **Numeric Comparison** Similar to the Just Works method but at the end, both devices will independently generate a 6 digit confirmation value using both nonces. These values are displayed in each respective device and the user manually checks if both values match, creating a connection resilient to MITM attacks.

As can be seen in Figure 2.5, even if both devices do not have keyboard input capabilities, it is still possible to obtain authenticated connections, through the Numeric Comparison method. Without any I/O capability from one of both devices, the pairing method will be Just Works and the connection will be unauthenticated. Passkey and Numeric Comparison are the most secure pairing methods, only requiring different actions from the user.

Vulnerabilities

Bluetooth Low-Energy presents many vulnerabilities that compromise the security and privacy of the exchanged messages, even though it has matured over the past years. Zhang et al. [ZWD+20] have shown four BLE design flaws in Android, which is the most complete operating system in IoT, as previously discussed in Section 2.2. These design flaws present serious security issues making the BLE exchanged messages vulnerable to Man-In-The-Middle and Downgrade attacks as well as information disclosure. The four Android BLE design flaws are:

- An Android application cannot enforce a specific pairing protocol (previously described in Section 2.2.3) even if it knows the capable pairing protocols of the peer BLE device.

- Android applications cannot cancel insecure pairing processes until conclusion or remove suspicious bonds.

- Android mishandles pairing errors, without notifying the application or the user, being vulnerable to fake devices.

- There are no mechanisms to obtain the negotiated pairing protocol on time or even start a new secure pairing process with the same peer BLE device.

The same authors present countermeasures for those issues but they present usability issues for the mobile users. Wu et al. [WNK+20] explore the vulnerabilities of the BLE link-layer to spoofing attacks by designing an attack (BLESA) where a rogue device pretends to be a previously-paired server device. This attack showcases the security flaws in the reconnection process of two previously paired devices. The rogue device rejects the authentication requests and can feed spoofed data to the client device. The authors also present mitigation techniques including fixing implementation bugs in the BLE stack and adjusting the security level of the connection based on the attributes access requirements, before sending the reading request. One alternative generic solution is to ignore the pairing protocol and use an application-layer protocol to obtain all necessary security guarantees.

2.2.4 Application-Layer Protocols

Most application-layer protocols used in the IoT world, i.e MQTT and CoAP, present a lack of security in the protocol design, where confidentiality of the exchanged messages is not assured, requiring extra standard services, like TLS [RFC 8446] and DTLS [RFC 6347], respectively [NC20]. To guarantee such property object security might be useful since it provides security beyond the simple communication transport. Such an approach has already been shown to be successful in the IoT world [VTR+14]. Tjäder [Tjä17] showed how to guarantee message confidentiality in Bluetooth Low-Energy exchanges by enhancing such communications with COSE [RFC 8152]. CBOR Object Signing and Encryption (COSE) is an IETF standard on how to process signatures, encryption, and Message Authentication Codes (MAC) computations for CBOR (Concise Binary Object Representation) message-encoding format. It was already implemented in RESTful Environments to protect CoAP messages exchanged between constrained devices, through the OSCORE [RFC 8613] standard, as an alternative to DTLS. However, COSE can still be improved with a more lightweight and efficient message-encoding format, regarding its usage on constrained devices. Jenkov [Jen19] has shown in practice that Protocol Buffer (Protobuf) messages with five different fields have a much higher read and write throughput than equivalent CBOR messages. Moreover, protobuf messages were already introduced in the world of heterogeneous constrained devices as a lightweight and interoperable alternative to standardized message encoding schemes like CBOR, JSON, or BSON [PPMT16].

2.3 Usable Security

Zurko et al. [ZS96] conceived one of the early concepts of user-centered security in the 1990s. They showed several approaches on how to achieve it depending on the stage of development.

User-centered security design from the early stages is the most highlighted approach in the paper, even presenting a case study. Applying usability to established and properly tested security systems is also one possible solution. To enhance the usability of software, lab testing with users performing specific tasks with the software, with their reactions and problems monitored, can be done. Although, one of the five lessons learned by Balfanz et al. [BDSG04] is that both concepts must coexist in the very first stages of system design and that applying one on top of the other after the design process is a mess. Another lesson, applied in the deployment of our Kiosk, is to think locally, act locally. Systems that follow this principle do not require coordination with infrastructure and allow bigger automation, aiming on being more user-friendly.

Authentication ceremonies are one of the most important and well-understood security usability challenges, specifically in secure messaging [FGK21][SHWR16]. Users must complete a sequence of secure manual operations to verify their identity but those may introduce awkwardness to the users because they may represent policies or mechanisms that go against their values, like reputation, trust, or even productivity. A solution, shown by Lorri et al. [LFC05] is to make users believe that their assets are under attack and that the security mechanisms provided by the authentication ceremony are effective against such attacks. Dodier-Lazaro et al. [DLASBS17] show that users rely on incorrect heuristics to understand the security of a system, their conception of security differs from the real security value and that designers must use visual and interactional cues to decrease the distance between those security conceptions. Assal et al. [AHIC15] showed that those misconceptions also extend to privacy, where the knowledge of the users is poor and are not aware of possible leaks. Furthermore, the authors also show that privacy-preserving mobile applications offer low usability, showing a new usability tradeoff.

Fassl et al. [FGK21] describe an entire user-centered design approach based on a four-stage process, where the last stage is a mixed-methods evaluation containing a user survey that tries to understand the perceived security, relative to the possible threats of the system, and a User Experience Questionnaire, along with a Systems Usability Scale section. We followed a similar approach to evaluate the usability of our solution. Dressel et al. [DLE19] evaluate a WebBluetooth two-factor authentication method using a framework based on a user study covering the performance, perceived usability, and possible attacks of this authentication system. A similar framework can be done for user-centered secure location-proof systems.

2.4 Summary

Location certification systems have come a long way, but have converged into a general architecture, including witnessing systems and digital signatures as proofs. Location proofs have multiple

use cases, from medical appointments to the commercial sector. IoT technologies are diverse and heterogeneous and there is not a single standard protocol to be used every time. The five-layer IoT architecture provides a model for developing any IoT functionality. Low power communication protocols like 6LoWPAN and BLE have shown great potential in wearable devices. Both hardware and software IoT stack show vulnerabilities and mitigating them cost processing power and energy consumption. Modern developments of kiosks can be built on top of Raspberry Pi as the computation building block, related to the hardware stack, and on top of Android OS as the software stack. Bluetooth Low-Energy presents many design flaws that motivate the search for an application-layer protocol to be applied on top of pairing-less communications. COSE provides object security for CBOR encoded messages and was already implemented in an IoT environment. Authentication ceremonies are one of the most well-understood challenges in usable security, being the borderline between the emotions and values of users and secure systems. There are no location certification systems especially targeted at wearable devices focused on leveraging the user experience.

Chapter 3

SurePresence

We developed SurePresence, a two-components application that allows a user to prove his location when interacting with a presence interactive kiosk through the use of location proofs. These two components are: *Client* and *Kiosk*; the first component represents the prover that makes location claims while the second represents a witness of the system that endorses the claim. SurePresence integrates the SureThing framework for location proofs, described in Section 3.1. In Section 3.2 we detail all aspects of the design of the SurePresence system and in Section 3.3 we show the implementation of our solution in a medical office use case where a patient can justify his absence of work or school when attending an appointment, using location proofs.

3.1 Framework

The SureThing framework defines a model for location certification and provides libraries and services to develop systems that issue, verify and store location certificates for mobile and ubiquitous devices, i.e IoT devices. In the heart of this framework, there is the SureThing Core, composed of three libraries, data, util, and API, detailed in the next Sections. We provide the definitions of what a location proof is in the context of the SureThing Framework.

3.1.1 Core Data

The SureThing Core Data library is responsible for defining the central datatypes provided by the SureThing framework, i.e, data types that can be reused in multiple different applications. The underlying representation format of data is Protocol Buffers (abbreviated as Protobuf) [KF10]. This library contains the protobuf message definitions supporting the representation of such entities in different major programming languages, through data marshallers which generate compact and efficient code, suitable for IoT platforms.

Next, we explain the concept of location proof, since it is represented by multiple SureThing concepts, including Location Claims, Location Endorsements, and Location Certificates.

Location Proof

In the SureThing context, a location proof is a procedure and associated data structure to users trying to prove their location with any application implementing the SureThing Core. A location proof is mainly represented by three structures: Location Claim, which represents the claim made by the prover about its location at a specific time, the Location Endorsement, which represents an endorsement made by a witness of a location claim and a Location Certificate, which is produced and signed by the Verifier upon validation of the digital signatures of both claims and endorsements. This last artifact is the strongest location certification guarantee from the SureThing framework. In our solution, the kiosk represents a witness and is capable of endorsing any claim produced by a prover at its location. Witnesses can endorse as many claims as they can and a claim can be endorsed by multiple witnesses.

Location Claim

A location claim is created by the prover and contains, among other types of information, the location of the prover. It is a SureThing data type from the SureThing framework, thus, its underlying representation format is Protocol Buffer. Its definition can be seen in Listing 1. The *claimId* is a 128-bits Universe Unique Identifier (UUID) generated by the prover. This field will be used to identify the claim when endorsing and when stored in a persistent Ledger. The proverId is the unique id of the user trying to prove his location. Can be an email address, a citizen card ID, or even a pseudonym, to guarantee user privacy. The Location field is a set of alternative location schemes. Can either be represented by latitude/longitude coordinates, a point of interest (POI), its proximity, or even an open location code. The time field can be either a timestamp, relative to any Epoch or a time interval. Finally, there is the evidence, which can be anything used by the prover to strengthen the claim, like a photo, a hand-writing signature, or even other types of biometric authentication.

To guarantee the non-repudiation of the claim, the prover digitally signs its claim creating a Signed Location Claim, which is another Protocol buffer message containing the claim and its respective signature, which is a byte string. It is represented in Listing 2. For this purpose, each user has a key pair, which is generated at the registered moment or retrieved when logging in. The Verifier retrieves the public key of the respective user from the Certificate Authority.

```
message LocationClaim{
    string claimId = 1; // unique identifier for the claim
    string proverId = 2; // unique id of the prover
    Location location = 3; // location of the prover
    Time time = 4; // Time of creation of the location claim
    string evidenceType = 5; // Type of the evidence supporting the claim
    Any evidence = 6; // Any evidence used to support the claim
}
```

Listing 1: SureThing location claim protocol buffer definition.

```
message SignedLocationClaim{
    LocationClaim claim = 1; // location claim
    Signature proverSignature = 2; // signature of the prover
}
```

Listing 2: SureThing signed location claim protocol buffer definition.

Location Endorsement

A location endorsement is created by a witness that endorses a location claim created by a prover. This endorsement can be created by any credited witness, for example, the kiosk. Its definition can be seen in Listing 3. It contains the id of the witness endorsing it and the id of the claim. The remaining fields are the same as in the location claim.

To guarantee the non-repudiation of the endorsement, the witness digitally signs its endorsement creating a Signed Location Endorsement, which can be seen in Listing 4.

Location Certificate

Location certificates are generated upon successful verification of location claims produced by provers. These certificates are issued by the Verifier and their definition can be seen in Figure 5. Besides referencing a location verification artifact, which is also produced by the Verifier, it contains its signature, to provide non-repudiation to the whole validation process.

The location verification structure indicates if the location claim is valid and if it was successfully verified. Its definition is shown in Figure 6. It includes a unique identifier of the Verifier

```
message LocationEndorsement{
    string witnessId = 1; // witness unique id
    string claimId = 2; // the original location claim of the prover
    Time time = 3; // time of endorsement
    string evidenceType = 4; // type of the extra evidence
    Any evidence = 5; // the extra endorsement evidence
}
```

Listing 3: SureThing location endorsement protocol buffer definition.

```
message SignedLocationEndorsement{
    LocationEndorsement endorsement = 1; // the original location endorsement
    Signature witnessSignature = 2; // the signature of the witness
}
```

Listing 4: SureThing signed location endorsement protocol buffer definition.

```
message LocationCertificate{
    LocationVerification verification = 1; // the successful verification artifact
    Signature verifierSignature = 2; // the signature of the verifier
}
```

Listing 5: SureThing location certificate protocol buffer definition.

(allowing multiples instances of verifiers), the identifier of the location claim being verified, the list of endorsements produced by multiple witnesses, the time the verification took place, following the same time schemes as the previous SureThing data types, and extra evidence that can be used to strengthen this structure.

3.1.2 Core Utils

The Core Utils library implements the specification for digital signatures to use in claims, endorsements, and certificates. It details how to use cryptographic algorithms to create and verify digital signatures of those data structures specified in the SureThing Core Data formats.

The procedures may be implemented in different programming languages, as a library, that can then be used in applications making use of location certificates.

The current version of the library contains Java and Python code, which were the programming languages used for the development of SurePresence.

3.1.3 Service API

The Service API specifies the interfaces defining the communication models for all types of systems and components integrating the SureThing framework. It defines how the communication

```
message LocationVerification{
    string verifierId = 1; // the verifier unique id
    string claimId = 2; // the original location claim of the prover
    repeated string endorsementIds = 3; // list of endorsements by witnesses
    Time time = 4; // time of verification
    string evidenceType = 5; // type of the evidence
    Any evidence = 6; // verification evidence
}
```

Listing 6: SureThing location verification protocol buffer definition.

is processed in a rigorous language, through well-documented skeleton code, for both server and client sides, in the same major programming languages as the Utils library. Such code and language may differ depending on the role and resources of the system using this framework.

3.2 Design

In this Section, we explain all aspects regarding the design and implementation of the SurePresence system and its location proof techniques. We discuss the assumptions, present the requirements identified, and an overview of the architecture of our system.

3.2.1 Assumptions

The entities utilizing SurePresence, which we call system operators, are the businesses that need location certificates to provide some service or functionality. They decide the locations where to use this application, which we call points of interest. These are just used, not owned, or operated by these entities, and influence the chosen location proof technique among the ones described in Section 3.3.2.

Related to the presence kiosk component, we also take on some assumptions. We assume the kiosk is a plug-and-play device [SCRH20a] and is only deployed in trusted locations with a reliable power source. Trusted locations are hand-picked points of interest that do not show any harm to the kiosk, possibly protected by a bystander, and that have trustworthy resources, like a secure Wi-Fi infrastructure.

We assume the devices where SurePresence is deployed are capable of providing a location or position of a user through GPS or GNSS. Our solution does not generate the location of a user, only proofs to ascertain its legitimacy. For now, we also assume that those services are responsible for the distribution of keypairs for their users and symmetric keys to be used in the communications between the users and the kiosk. We also assume the previous distribution of the digital certificates of the kiosk and the server, for secure communications. We leave for future work possible requests to a Certificate Authority to retrieve all necessary keys and certificates, as it is integrated into the state-of-art architecture of location certification systems, as described in Section 2.1.1. We also assumed an already logged-in account in the development and evaluation of the SurePresence application, as we leave the login paradigm for future work.

Finally, we assume that either the kiosk or the client will have a Wi-Fi connection at some point, to submit the location certificate to the server. The best-suited technique to be used will be determined by which entity has a Wi-Fi connection. Otherwise, our solution will not work. We do not take any more assumptions related to the density of Wi-Fi networks, Bluetooth

beacons, or infrastructures in the points of interest.

3.2.2 Requirements

SurePresence should only produce location certificates for authenticated users. Even if users do not have an account, they should be able to authenticate themselves when interacting with the kiosk. The need of having an account drastically reduces the user experience of the authentication ceremony since the user is forced to register an account before interacting with the kiosk. Finally, our solution does not depend on the location, environment, or context of usage to produce location certificates. So we can summarize the functional requirements:

- **R1** - Location proofs can only be produced for authenticated users.
- **R2** - The user does not need an account when approaching a kiosk.
- **R3** - The kiosk component of SurePresence only requires a consistent power source from the deployment location to successfully endorse a claim and submit it to the verifier.

The most important non-functional requirement is *usability* to leverage the user experience provided by SurePresence and the SureThing framework. Other requirements are:

- **Portability** - SurePresence should be ready to be used in some mobile and wearable devices. Should also be able to create location certificates in the absence of such devices.
- **Adaptability** - Correlated with usability. The authentication ceremony needed to create location certificates must be familiar to users.
- **Interoperability** - The different operators, entities, and technologies should be able to communicate among themselves to create the location certificate.
- **Verifiability** - Any location certificate produced should be verified and cannot be consumed if it is not.
- **Extensibility** - Our system must be flexible for future modifications and extensions.
- **Security** - All components of our solution should resist to all types of attacks.
- **Privacy** - Sensitive and private information about the user should not be disclosed in the absence of a bystander in the kiosk location.

3.2.3 Architecture

The architecture of the developed SurePresence application is illustrated in Figure 3.1 and it is based on the state-of-art location certification systems architecture shown in Figure 2.1. In

SurePresence, though, witnesses cannot act as provers, since our only witness is a trusted interactive presence kiosk. Provers of our system are portrayed as SureThing users. The server acts as a verifier by receiving the location claims and endorsements through the Service API library previously explained in Section 3.1.3 and verifying their legitimacy using the Core Utils library, previously explained in Section 3.1.2, generating location certificates previously explained in Section 3.1.1. In our architecture, we do not have a Certificate Authority (CA), since both witness and verifier are trusted entities and provers authenticates themselves in every request using an API authentication token, as you will see in Section 3.2.4.

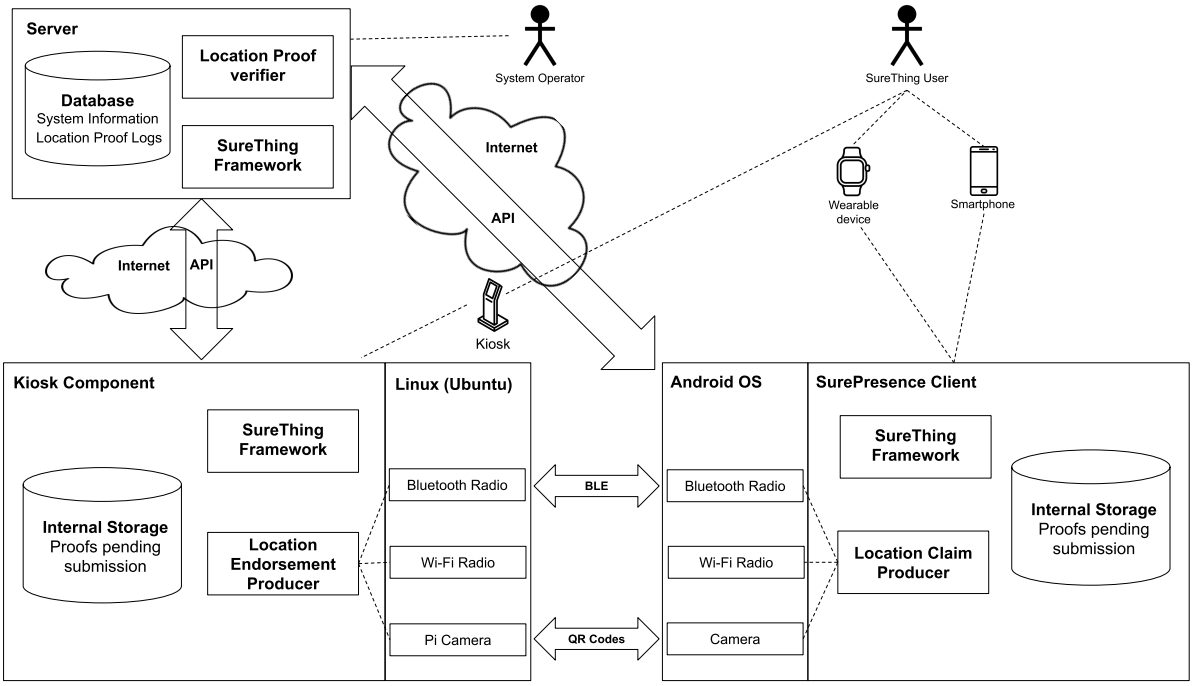


Figure 3.1: SurePresence architecture.

The client component on the bottom right side represents the prover. It is a component that runs on smartwatches and smartphones and has the responsibility of authenticating the user when interacting with the kiosk and generating signed location claims, previously described in Section 3.1.1. Our witness model consists of just one entity, the kiosk component on the bottom left side. Just like in APPLAUS [ZC11], PROPS [GKRT14], PASPORT [NSY+20] and in [AA20], the witness is capable of generating both claims and endorsements, depending on the location proof technique. The server, which can be seen on the top left corner of the figure, acts as the verifier and is responsible for storing verified location certificates. The API request handler will deal with all requests made by the kiosk and client API, over the Internet. The remaining elements will be better explained throughout this work.

This client-server model with a single witness also represents a simpler architecture when

compared, for example, with a peer-to-peer communication model. A centralized approach is easier to manage, protect, and trust, as the server is managed by the service provider (and the system operator) with whom the user may have some trustworthy relation.

As detailed in Figure 3.1, proofs pending submission can be stored locally and submitted once a Wi-Fi connection is established in both kiosk and client devices. Thus, a constant connection to the server is not necessary.

3.2.4 Security and Privacy

In our model, we consider the server and the kiosk as trusted entities, and the client as untrusted. Although it is unusual to consider a witness as a trusted entity, the kiosk will not have human control and will not help provers in possible collusion attacks. The user even after authentication may still be a malicious prover. Although the server is trusted, we consider the system operators as possible eavesdroppers of the information stored in the server.

The communication between the client and the server is similar to the communication between the kiosk and the server, which is over a REST API using the HTTPS protocol. REST is an architectural style for web services, defining a set of constraints to be used in the communication with those services. One of the constraints that define a RESTful system is that it must be based on a client-server architecture, just like the SurePresence, but does not restrict the technologies communicating. REST supports many data formats for both request and response messages. We use Protocol Buffer payloads as our message format as described in Section 3.1.1.

HTTPS is a secure extension of HTTP with TLS in the transport layer, to provide applications with secure TCP connections. These are commonly used and widely supported by most programming languages. We studied the possibility of using COAP, instead of HTTP, but it does not include any security features, besides showing bad packet delivery [YSAz16] since it uses UDP datagrams, which are unreliable. To provide security, DTLS would need to run on top of UDP, but it is not suitable to be used in IoT devices [KCVGAZ15].

These communications follow the Service API specification of the SureThing framework, which includes support for HTTPS communications across multiple programming languages. Since all communications with the server are done using TLS in the transport layer, we guarantee confidentiality, integrity, and freshness of the transmitted data and server and kiosk authentication using public-key cryptography. Digital certificates provide authentication to both the kiosk and server. These certificates are bundled with the client application, on both the kiosk and user side, as part of our assumptions, previously described in Section 3.2.1.

For client authentication, certificates could also be used but represent a big drawback related

to the server storage since it would need to have certificates for all users containing their public keys. Instead, we leave for future work, the use of an API token that is provided to the user once he authenticates with his credentials. That token is sent along with future requests and is verified by the API request handler on the server-side. We considered the same logged-in account, as described in Section 3.2.1.

HTTPS can be used to protect the privacy of user information. Although, encrypting user information with the public key of the server or hashing before storing it protects the server stored information from the system operators. We leave such strategies for future work.

The communication between the client and the kiosk is over Bluetooth Low-Energy, previously explained in Section 2.2.3. Motivated by the security issues described in the same section, we implemented POSE, an end-to-end application layer security protocol for pairing-less BLE communications. This protocol is later detailed in Section 3.3.3.

3.3 Implementation

We deployed the SurePresence application in a medical office use case, where a patient, acting as prover, authenticates himself with the interactive kiosk, present in the medical office, and generates a location certificate that can be used to obtain the legitimacy of the medical appointment, in case the patient needs to obtain any legal document for the absence of work or school. This use case represents the singular *location case of traceability*, presented in Section 2.1.3, since attending a medical appointment does not require assessing any past locations or proofs. A more detailed architecture of the SurePresence system adapted to this use case can be seen in Figure 3.2, where we show the components required for the novel location proof techniques of both applications more thoroughly. In Section 3.3.1 we detail the platforms and technologies used to deploy SurePresence. In Section 3.3.2 we describe the novel location proof techniques a prover can perform when interacting with a kiosk to generate location certificates. In Section 3.3.3 we explain the application-layer protocol we developed to ensure secure Bluetooth Low-Energy communications, regarding the second novel location proof technique. In Section 3.3.4 we do a summary of the major aspects of our implementation of SurePresence.

3.3.1 Platform

The SurePresence client wearable application is targeted for Android Wear-based smartwatches. We developed this application for the Android Wear OS since Android is the most complete operating system to be used in IoT environments, as we previously concluded in Section 2.2. Our client application is written in Java 8 and uses Java libraries from the SureThing framework

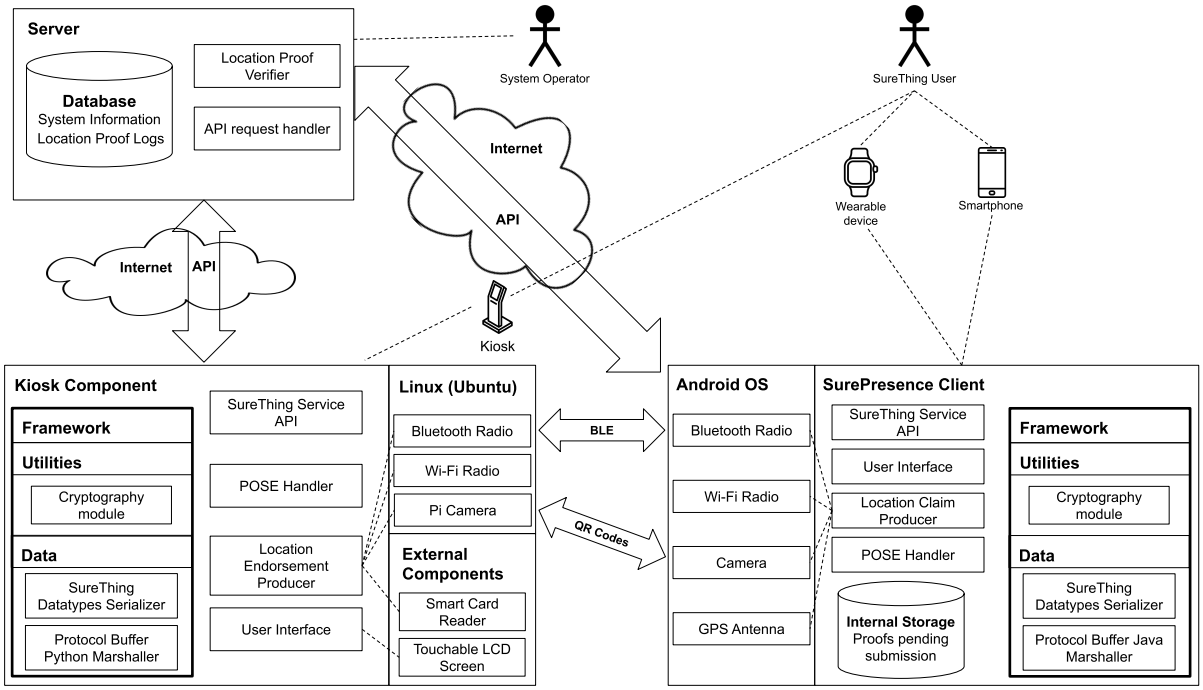


Figure 3.2: SurePresence detailed architecture for the medical use case.

deployed in the local Maven repository. Android Wear OS constraints the minimum Android Software Development Kit (SDK) to be used. Since we developed our application for Android Wear 1.0, the minimum SDK is 23. Our chosen SDK version was 26. This version of Android removes programmatic access to the local hardware identifier of the running device, for applications using Wi-Fi and Bluetooth APIs. This means our application cannot access the wearable Bluetooth adapter MAC address, which was one of the main restrictions when developing SurePresence. Such restriction motivated us to use the name of the device for identification.

The SurePresence client smartphone application is targeted for Android (8.0) based smartphones, which is justified by the same reason as in the wearable application. This application was also written in Java 8, uses the SDK 26, and uses the same libraries as the wearable application.

The SurePresence kiosk application is targeted at any Ubuntu-based device. The application was mainly written in Python 3, including the Graphical User Interface, which was developed using the PyQt[1] library. The application includes a Java 8 component required to execute the official Portuguese government middleware for reading the citizen card (smart card)[2]. Regarding the Bluetooth stack, this application uses the BlueZ[3] protocol libraries and utilities to discover nearby Bluetooth Low-Energy connectable devices and uses the gatttool[4] tool to connect and to

[1]https://wiki.python.org/moin/PyQt
[2]https://www.autenticacao.gov.pt/web/guest/cc-aplicacao
[3]http://www.bluez.org/
[4]http://manpages.ubuntu.com/manpages/cosmic/man1/gatttool.1.html

Table 3.1: Platform summary of the SurePresence components.

Component	Operating System	Platform	Version
Wearable application	Android Wear OS 1.0	Java	8
Smartphone application	Android 8.0 (Oreo)	Java	8
Kiosk application	Linux (Ubuntu Desktop 20.04)	Python	3.7

read attributes from other devices acting as GATT servers, as previously explained in Section 2.2.3. To read QR codes, the application uses the Zbar[5] code reader library. Just like in the previous applications, also makes use of the Python libraries from the SureThing framework. A summary of the platforms and operating systems used can be seen in Table 3.1.

3.3.2 Novel Location Proof Techniques

We present the novel location proof techniques through which a user can generate a location certificate when using the SurePresence system and interacting with the kiosk. They benefit from the presence kiosk that authenticates the user and generates location endorsements to strengthen the claims produced by the users. These techniques interact differently with it, through different devices presenting flexibility and portability on the production of location certificates. Those interactions represent authentication ceremonies for the first two presented techniques. In the third technique, QR codes are used to exchange location proof information between the prover and the kiosk. Since it is easier for the users to understand the concept of authentication rather than location proof, they are always informed that they are authenticating themselves in any technique. In Figure 3.3 you can see the main screen of the kiosk, where users select the device they want to use to authenticate themselves. Each device represents a location proof technique. We thoroughly explain each one and respective authentication ceremony.

Kiosk-Only Technique

This technique allows a user to prove his location only by using his citizen card. We make use of the official Portuguese official citizen card middleware, as previously mentioned in Section 3.3.1, to read all the information we need to authenticate the prover, without damaging or modifying the smart card. This is taken care of by the smart card reader embedded in the kiosk.

The user approaches the kiosk, selects the citizen card authentication, and reaches the interface shown in Figure 3.4. An auxiliary thread running in the background periodically checks the introduction of the citizen card in the reader, without requiring any extra confirmation or

[5]https://github.com/Polyconseil/zbarlight

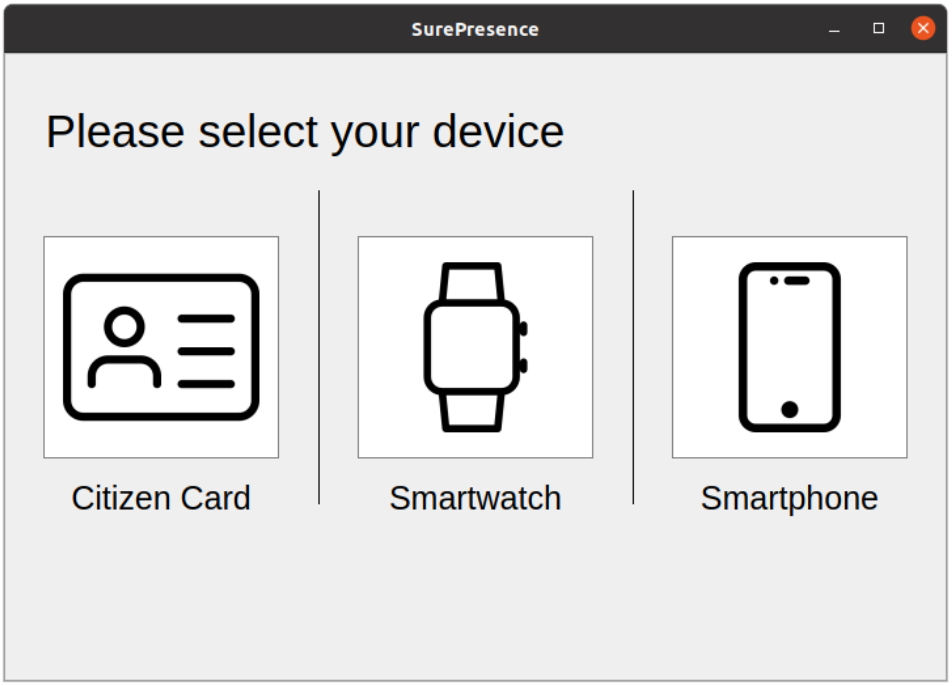


Figure 3.3: Main screen of the Kiosk UI.

any other action from the user. The thread is killed when pressing the "back" button or by successfully introducing and reading the citizen card.

Only public information is read from the citizen card since private information is not necessary and would require an authentication PIN from the user. That public information includes the name and the citizen card ID from the user. The photo of the card can also be used as extra evidence when creating the location claim since it is also public.

Unlike the other techniques, the prover does not create his location claim, which is taken care of by the kiosk. Although it may seem odd that it is a witness claiming the presence of the prover, it is justified by the physical interaction with the kiosk when introducing the citizen card to the reader.

The location claim requires a unique ID from the user. In this technique, we use the Citizen Card ID while in the other techniques we use e-mail. Other information, which can be seen in Section 3.1.1, like the time and location of the authentication ceremony moment, are also provided by the kiosk, which knows its exact deployment location.

The kiosk creates a signed location endorsement, as described in Section 3.1.1, from that same location claim and sends both artifacts to the Verifier, which requires a Wi-Fi connection. After concluding this process, the kiosk informs the users they can remove their citizen card, as you can see in Figure 3.5.

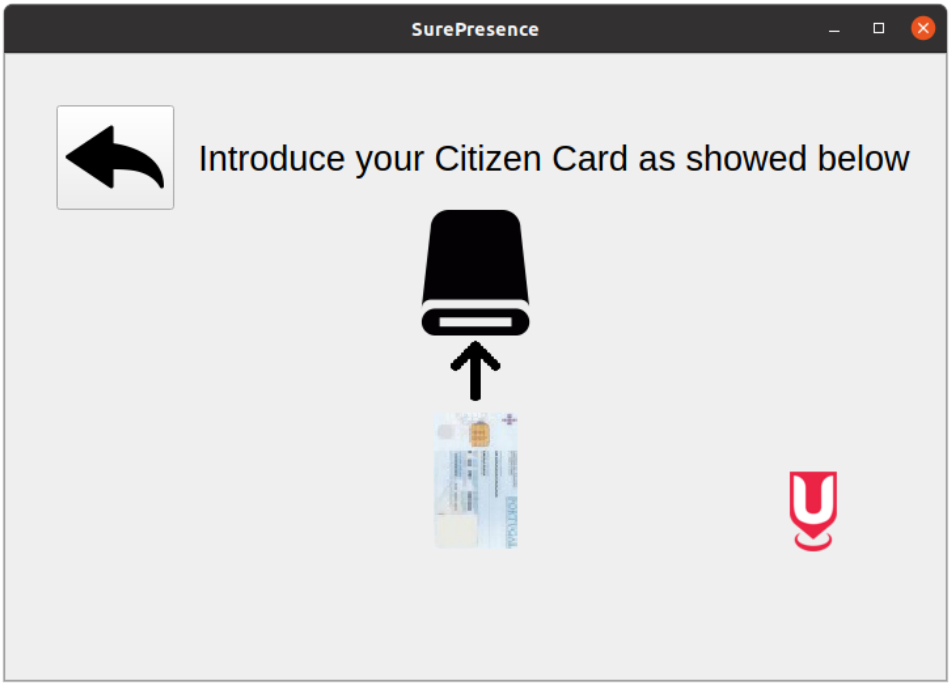


Figure 3.4: Citizen card screen of the kiosk UI.

Kiosk-Wearable Technique

This technique is the most IoT-ready of the three. It generates a location certificate through the interaction of a wearable device with the kiosk. The security of the communications between these two devices is assured by the POSE protocol, described in Section 3.3.3.

To start the authentication ceremony, the user needs to select the smartwatch icon of the kiosk main screen. After pressing it, the user reaches the screen shown in Figure 3.6.

The instructions on this screen are clear. Users have to open their wearable application and select the "Kiosk" button. After pressing it, an auxiliary thread on the wearable will initiate all the necessary Bluetooth structures to connect to the kiosk. This could be done when the user logs in or initiates the application, but that would consume unnecessary energy and wearable resources, which are important regarding IoT devices. We only consume those when we are certain that the user is in front of the kiosk and ready to interact. When the wearable has everything set up, it displays its device name, so that it can be identifiable by the user. This wearable screen is the second screen displayed in Figure 3.7.

Regarding the BLE concepts explained in section 2.2.3 we now explain the roles that both the wearable and the kiosk take in this BLE communication. The wearable has to send user information to the kiosk so that users are authenticated and a location certificate can be created for them. Therefore, the wearable is the **GATT Server** and since the kiosk has to read information from the wearable, it takes the **GATT Client** role. The kiosk does not know the name

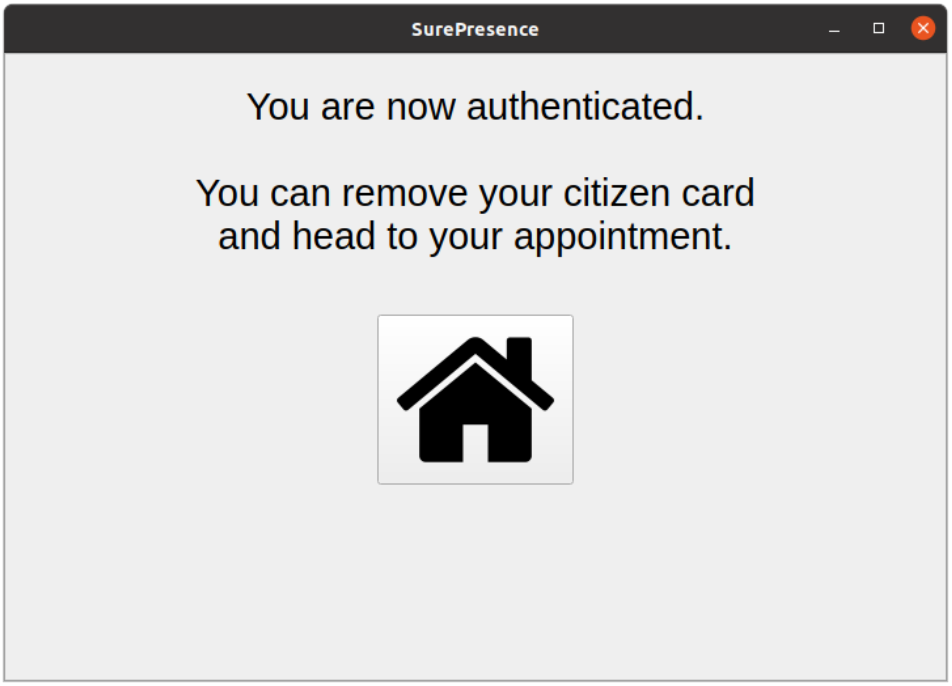


Figure 3.5: Citizen card final screen of the kiosk UI.

or the MAC address of the wearable that is supposed to connect while the wearable application knows exactly the MAC address of the kiosk. Therefore, the kiosk has to scan for the wearable device. Otherwise, it would need to accept any BLE connection, which is extremely unsafe. So, the kiosk takes the **Central** role and the wearable is the **Peripheral**.

After pressing the "next" button in the kiosk UI, the kiosk immediately starts scanning for nearby devices. For each device found, it will update its interface with the name of the device found (or the MAC address, in case the name is not defined). The user selects his device from the list of scanned devices, which can be seen in Figure 3.8. Android does not allow a device to access its own MAC address, previously explained in Section 3.3.1, which is the main reason why we are using the name of the device for identification. It shows to be a problem, as it can be solved with randomly generated MAC addresses translated into user-friendly identifiable information. We leave this and other possible solutions for future work.

The selected device becomes highlighted and the "Confirm device" button only shows up after selecting a device from the list. Double tapping on the selected device also confirms it. After confirmation, the kiosk sends a connection request to the chosen wearable, which is waiting for connections. If the connecting device does not have the known MAC address of the kiosk, the connection request is rejected. Motivated by the vulnerabilities in the pairing protocols of BLE, previously detailed in Section 2.2.3, we decided these devices should not pair.

Without the pairing process, the communication between the wearable and the kiosk would

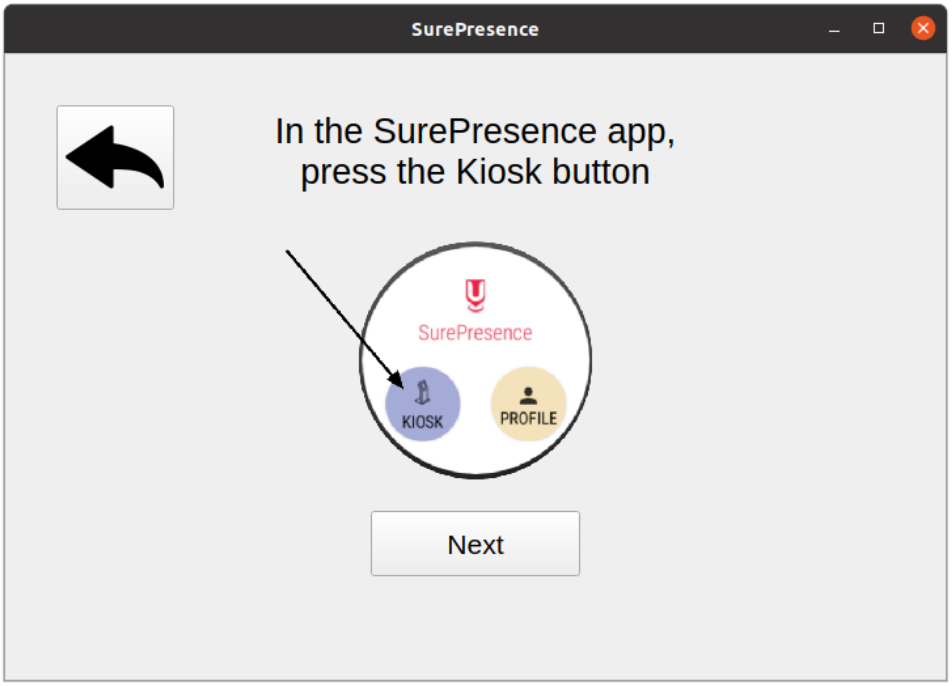


Figure 3.6: First kiosk screen of the Kiosk-Wearable technique.

be done in plaintext, which would become vulnerable to eavesdropping and MITM attacks. To overcome such vulnerabilities, we implemented the POSE protocol, an end-to-end application layer protocol to secure the exchange of such messages between IoT devices, on top of pairing-less BLE communications, which can be seen in section 3.3.3.

The wearable creates a location claim with the e-mail of the logged-in user and other relevant information, just like in the previous technique. Obtaining the exact location through GPS in such a constrained device as a smartwatch may consume too many resources, possibly putting in jeopardy the user experience provided by the wearable application. Since both the MAC address and the deployment location of the kiosk are known, we created a file where we do this MAC address - Latitude/Longitude relation. This file is bundled with the wearable application. The wearable signs the claim, creating a signed location claim which becomes readable through a GATT characteristic, provided by a GATT service, as explained in Section 2.2.3. The kiosk reads such characteristic, creates a location endorsement with the information provided by that claim, and signs it creating a signed location endorsement, which is sent to the Verifier.

Once the characteristic is read and the process is finished, the user is notified in the wearable, as you can see in the third screen in Figure 3.7 and the kiosk, just like in the previous technique.

Figure 3.7: Different screens of the SurePresence wearable application.

Kiosk-Smartphone Technique

This technique was originally proposed by Maia et al. [**?**], in the SureThing context, but never deployed. It may be the most familiar technique for the users since it makes use of a device that everyone uses every day and maybe the most suitable for current location-based applications. Just like in the previous techniques, we keep using the same authentication concept, which is easier for the user, but the interactions he has with the kiosk are related to location claims. The user selects the smartphone icon and reaches the screen shown in Figure 3.9.

The user needs to access the "QR Code" section on his smartphone application which automatically creates a signed location claim in the same way as the previous technique. It uses the e-mail of the logged-in user as the unique ID and obtains its exact location through GPS. The smartphone does not communicate through BLE with the kiosk and does not know its MAC address. The QR code shown in Figure 3.10 displays a signed location claim, encoded in a Base64 string.

The kiosk, using its camera module, scans the QR code and obtains the signed location claim. It generates the signed location endorsement encoded in a Base64 string and displays it in a new QR code, which can be seen in Figure 3.11. Since a QR code does not know when it is scanned, we had to include in the kiosk UI a "done" button on the top right corner of the screen to conclude the technique on the kiosk side.

Unlike the previous techniques, it is the prover who submits both claim and endorsement to the Verifier side. To obtain the signed location endorsement, the user scans the QR code displayed in Figure 3.11, through the "Scan" section of his smartphone application. If the scan is successful, there is audio feedback and a message, which you can see in Figure 3.12.

The smartphone locally persists all generated signed location claims and scanned signed

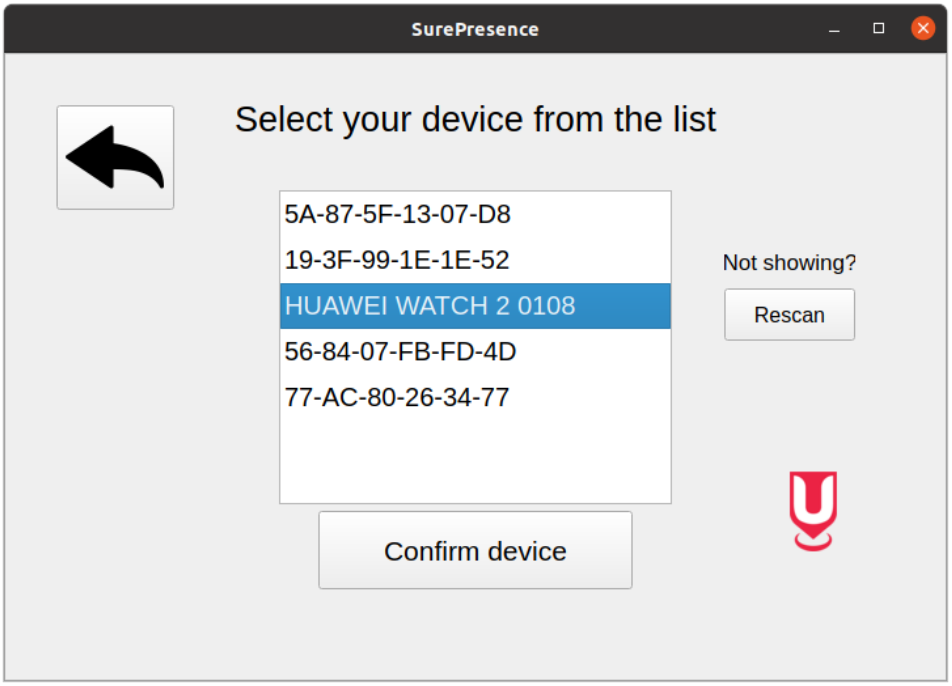


Figure 3.8: Kiosk screen with the list of scanned devices.

location endorsements, through an SQLite database. This database provides two tables, one for each artifact, where the key in both tables is the UUID identifying the proof (claims and endorsements), as explained in section 3.1.1. Both claims and endorsements are stored as byte blobs. If the UUID of a stored claim does not exist in the table of endorsements, it means the claim has not been endorsed by any witness, or in this case, by the kiosk. A location claim can be endorsed by many witnesses. The endorsement table also includes a "Status" column. This boolean value indicates if the claim and respective endorsements have already been sent to the Verifier side which is automatically done once a Wi-Fi connection is established and is shown in the "Presences" section of the smartphone application, as you can see in Figure 3.13.

This technique appears to be suited to remote locations or isolated environments where it can be impossible to obtain a signal or to establish a Wi-Fi connection, for example, when doing Smart tourism. The usage of a smartwatch instead of a smartphone could be done, but displaying a QR Code on a small screen and making the camera of the kiosk scan it, would bring usability problems besides the fact that wearable does not usually have an embedded camera.

With the help of a friend, a malicious prover can prove his location at the kiosk while being in any other place. This is something that can only be countered through a bystander checking for strange behavior or even biometric authentication, which we leave for future work.

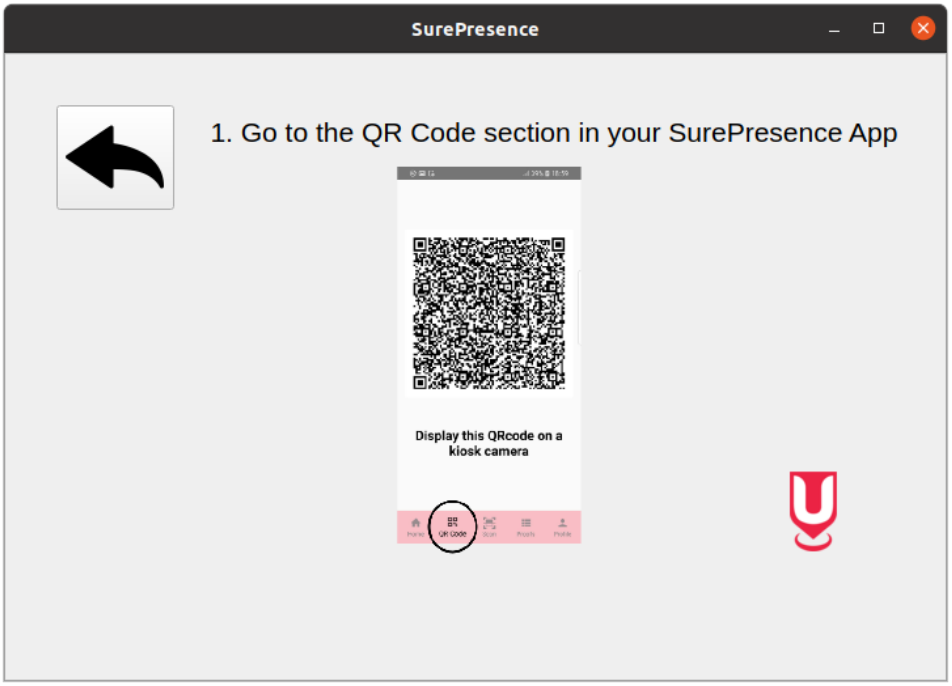


Figure 3.9: First kiosk screen of the Kiosk-Smartphone technique.

3.3.3 Message Security

The messages exchanged between the smartwatch and the kiosk, regarding the Kiosk-Wearable technique, over BLE need protection. To address this technical challenge, we developed the POSE (**P**rotocol buffer **O**bject **S**igning and **E**ncryption) protocol which is designed to provide an end-to-end security layer over BLE communications on constrained devices. The protocol is based on the COSE (CBOR Object Signing and Encryption) specification [RFC 8152] which is created to provide basic security services for the CBOR (Concise Binary Object Representation) data format [RFC 8949]. The COSE specification describes in detail how to create and process message signature, authentication codes and encryption using CBOR as the data encoding format for small size messages on constrained and limited devices. Fundamentally, POSE follows the COSE specification. However, it takes advantage of the SureThing framework internal data encoding format, which is Protobuf, to abstract the BLE exchanged messages.

Protocol Buffers have surpassed CBOR in popularity over the years, showing a much higher number of usages in multiple code repositories, including Maven Central[6] (3589 and 446, respectively) and PyPI[7] (13301 and 37, respectively). The "universality" of the data format is especially important in limited devices because they may not have the means to convert from and to other formats, at least, not without wasting precious energy.

[6]https://search.maven.org/
[7]https://pypi.org/

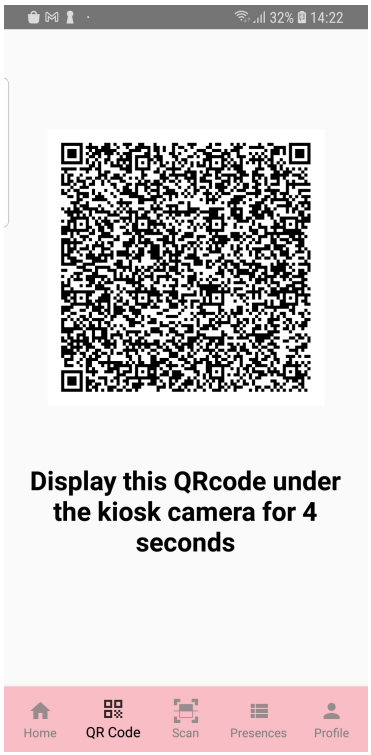


Figure 3.10: Smartphone QR code containing the signed location claim.

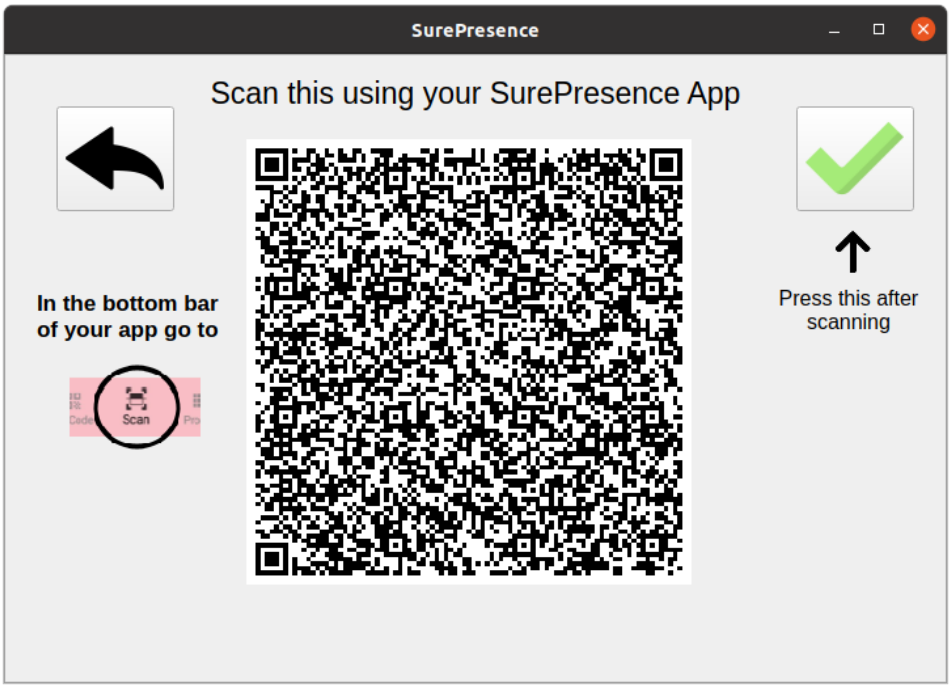


Figure 3.11: Kiosk screen with the QR code containing the signed location endorsement.

We start by describing the message types in Section 3.3.3 and then the messages format, in Section 3.3.3. In Section 3.3.3, we explain how POSE provides confidentiality and integrity of the exchanged messages and in Section 3.3.3 how it provides freshness. In Section 3.3.3 we detail how to create the specific POSE message type that provides all the security guarantees

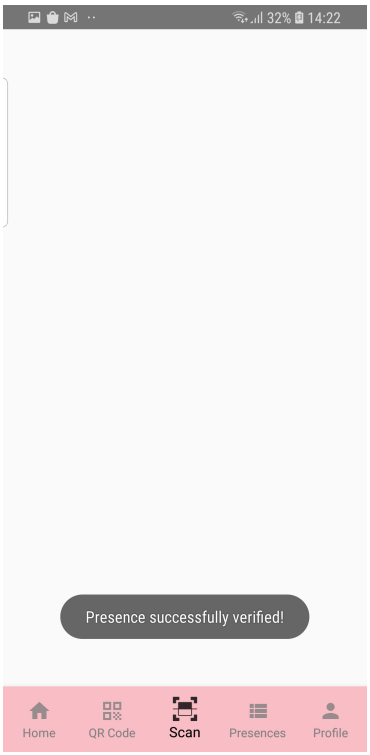


Figure 3.12: Successful QR code scanning screen of the SurePresence smartphone application.

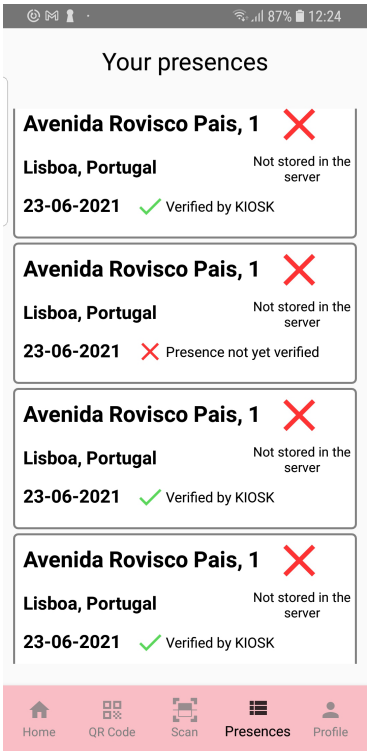


Figure 3.13: List of presences of a user in the SurePresence smartphone application.

needed by BLE, according to the format and fields previously explained.

Table 3.2: POSE message types.

POSE Message Type	Semantics
POSE_Sign1	Signed data object
POSE_Mac0	Mac data object
POSE_Encrypt0	Encrypted data object

```
message POSE_Sign1{
    bytes protected = 1;
    HeaderMap unprotected = 2;
    bytes payload = 3;
    bytes signature = 4;
}
```

Listing 7: POSE_Sign1 protocol buffer definition.

Message Types

In the context of a single recipient one-way BLE communications, POSE specifies three message types, each offering different security guarantees. These messages depend on the knowledge of who is the recipient and on an implicit symmetric key/asymmetric key pair, previously established. The different POSE message types can be seen in Table 3.2, as they follow the same grammar and have the same fields as the remaining COSE objects, as detailed in Section 3.3.3.

Signed Data Object The `POSE_Sign1` message type object ensures the non-repudiation of the transmitted data, by one signer only. It makes use of the same signature algorithms as used in the COSE standard, which is the Edwards-curve Digital Signature Algorithm (EdDSA) and the Elliptic Curve Digital Signature Algorithm (ECDSA). The protected bucket and payload are both signed, where the latter can even include another POSE message type, allowing the creation of sealed objects. The `POSE_Sign1` protocol buffer definition can be seen in Listing 7.

Mac Data Object The `POSE_Mac0` message type object ensures integrity and authenticity of both the payload and the protected bucket by generating a tag with a Message Authentication Code (MAC). It makes use of the same algorithms as COSE, which can either be a block cipher algorithm, like AES-MAC, or a hash algorithm, like HMAC. The Protobuf definition of the `POSE_Mac0` message type can be seen in Listing 8.

Encrypted Data Object The `POSE_Encrypt0` message type object is the one that offers the most security guarantees and is the type in which we focused on in our implementation (Section 3.3). This type ensures message confidentiality and integrity and message freshness. The Protobuf definition of the `POSE_Encrypt0` message type can be seen in Listing 9.

```
message POSE_Mac0{
    bytes protected = 1;
    HeaderMap unprotected = 2;
    bytes payload = 3;
    bytes tag = 4;
}
```

Listing 8: POSE_Mac0 protocol buffer definition.

```
message POSE_Encrypt0{
    bytes protected = 1;
    HeaderMap unprotected = 2;
    bytes ciphertext = 3;
}
```

Listing 9: POSE_Encrypt0 protocol buffer definition.

Message Format

POSE protocol supports the same primitive types as defined in the COSE specification, like Booleans, Integers, Byte strings, etc. Each field in a POSE object message can be a primitive type or Protobuf-defined message, depending on the type of the message, however, it always starts with the three fields: protected header parameters, unprotected header parameters, and the content of the message. Both protected and unprotected headers are 'label'-'value' maps. Each label is a well-defined Integer, while a value is a primitive type or another POSE message. The protected bucket contains parameters about the current layer and information about the used cryptographic algorithms, allowing flexibility on the chosen algorithms. All this information is protected since it is used in the cryptographic computation, where is either signed, hashed, or used as associated data in the encryption. The unprotected field is similar but not protected, meaning that it may not be authentic when it reaches the recipient. Both fields are serialized into a byte string. The content of the message field can be either plain-text, cipher-text, or another POSE message.

Message Confidentiality and Integrity

The different security guarantees are provided by the different message types, detailed in Section 3.3.3. To ensure both the confidentiality and integrity of the transmitted data, the `POSE_Encrypt0` message must be used. It uses the Authenticated Encryption with Associated Data (AEAD) as the form of encryption and the Advanced Encryption Standard-Galois/Counter Mode (AES-GCM) as the mode of operation for the encryption computation of the payload.

AES-GCM takes full advantage of pipelining and processing techniques unlike the remaining

modes of operations of block ciphers, making it ideal for constrained devices with inexpensive resources and lower rate applications of the IoT world [SKS18]. Other cryptographic algorithms can be used thanks to the specific format of POSE messages, previously explained, allowing exchange information about the algorithms used by the sender and that should be used by the recipient. This promotes flexibility on the encryption algorithms and opens a door for the usage of other more lightweight algorithms.

AEAD ensures the authenticity of the associated data used in encryption along with the integrity and confidentiality of the cipher-text. The encryption makes use of a pre-shared 128-bit symmetric key. We leave a possible handshake for a key agreement for future work. This mode of operation requires an *Initialization Vector* (IV) to guarantee the uniqueness of each encryption, which we randomize and use to guarantee the message freshness, as shown in Section 3.3.3.

Message Freshness

Although the AEAD with AES-GCM offers a lot of security guarantees, it does not guarantee the freshness of the transmitted data. This problem is not addressed in the COSE standard. POSE, however, solves this problem by using the partial IV of the AEAD encryption as a nonce field. This partial IV is in the protected field and is used to create the associated data.

Past used nonces are persistently stored in the kiosk, costing memory. The time these nonces persist is dependent on the system operator and the application making use of this protocol. In the context of our solution, we kept these nonces for 24 hours (schedule of a medical office).

After decrypting and verifying the signature of the signed location claim, we also verify if the date of the claim is from today. Since the kiosk memorizes nonces for 24 hours, replayed claims older than that would be considered fresh. To counter it, the kiosk only accepts, verifies, and endorses claims from the same day.

Security Processing

The first step to produce a `POSE_Encrypt0` message object is to create a consistent byte stream for the associated data later used in the cryptographic computation. For that purpose, a new object `Enc_Structure` should be created according to the Protobuf definition shown in Listing 10, following the POSE protocol. We describe the steps for the encryption/decryption processes.

Encryption

1. Create the `Enc_Structure` message object and fill it with the appropriate fields such as the context string and the protected attributes. The context string is a well-defined string

```
message Enc_Structure{
    string context = 1;
    bytes protected = 2;
    HeaderMap unprotected = 3;
    POSE_Encrypt0 body = 4;
}
```

Listing 10: Enc_Structure protocol buffer definition.

that identifies the type of the payload.

2. Serialize the created `Enc_Structure` object into byte stream using protocol buffers encoding to create the associated data (AD).

3. Call the encryption algorithm with the previously loaded encryption key K, the plaintext of the `POSE_Encrypt0` message object, and the AD. Then, include the result in the cipher-text field of the `POSE_Encrypt0`.

Decryption

1. Create an object of `Enc_Structure` message similar to the one received but without the cipher-text.

2. Serialize the created `Enc_Structure` object into byte stream using protocol buffers encoding to create the associated data (AD).

3. Call the decryption algorithm with the previously loaded decryption key K, the cipher-text of the `POSE_Encrypt0` object from the received `Enc_Structure` message and the AD.

3.3.4 Summary

In this Chapter, we presented our solution including the framework in the heart of SurePresence, its design aspects, including the requirements and assumptions, as well as its architecture and a security assessment of the whole system. We then detailed the implementation of SurePresence for a medical appointment use case, including the platforms where it was developed and three novel location proof techniques based on the interaction of a presence kiosk. Finally, we explained how messages are secured with the POSE end-to-end security layer protocol to guarantee the confidentiality, integrity, and authenticity of BLE exchanged messages.

Chapter 4

Evaluation

This chapter presents the evaluation of the SurePresence system. We first do a requirements check on the implemented code in Section 4.1.

In Section 4.2 we detail the technical evaluation comparing the developed POSE protocol with COSE, for the same implementation, which was the Kiosk-Wearable technique. We compared both protocols regarding their performance, CPU usage, and packet overhead on constrained devices and discuss the obtained results.

In Section 4.3 we describe the user study performed to the SurePresence system and its location proof techniques in the context of medical appointments. We do a quantitative and qualitative evaluation of the results from the user study, analyzed and discussed in Section 4.4.

In this Chapter, the novel techniques might be renamed for their respective devices, instead of their previous names (Kiosk-Only technique is the Citizen Card technique, for example).

4.1 Requirements Assessment

We start by assessing the functional and then the non-functional requirements of the SurePresence system.

4.1.1 Functional Requirements

All three requirements, enumerated in Section 3.2.2, were ensured by the flexibility of the different ways of generating location certificates through the three techniques. The **R1** requirement - Location proofs can only be produced for authenticated users - is ensured by the fact that, without user authentication, a location certificate is never produced. If a user does not have an account logged in any of the two ubiquitous devices, he can still use his citizen card to authenticate himself using the Kiosk-Only technique. Otherwise, a location certificate is not

produced.

When approaching a kiosk the user does not require an account, both logged in or simply created. Similar to the previous requirement, the citizen card ensures the **R2** - The user does not need an account when approaching a kiosk - requirement, as it is sufficient to authenticate the user and generate location certificates, leveraging the usability of our system. Enforcing a patient to create an account at the medical office, just to authenticate himself, would provide a poor user experience, and our system would probably be ignored at the office. We can see that the interception of both R1 and R2 requirements leads to the use of the citizen card.

The **R3** requirement - The kiosk component only requires a consistent power source to successfully endorse a claim and submit it to the verifier - is completely ensured by the Kiosk-Smartphone technique as it locally stores the location claims and endorsements. The kiosk component only requires a reliable power source to successfully witness a prover and endorse a claim. A reliable Wi-Fi infrastructure is not required to endorse location claims and submit these artifacts to the verifier since through the Kiosk-Smartphone technique, it is the prover that submits both claims and endorsements to the verifier side. For the remaining techniques, a Wi-Fi connection is not also required to successfully endorse a claim, but the system will not work since the kiosk will not have a connection to the verifier.

4.1.2 Non-functional Requirements

Regarding the non-functional requirements, the SureThing framework ensures most of them. Thanks to both Core Utils and Core Data libraries, previously described in Sections 3.1.2 and 3.1.1, we ensure the **Interoperability**, **Verifiability** and **Extensibility** requirements. The **Portability** requirement is ensured by the Kiosk-Only technique since it is not needed any ubiquitous device to produce both location claims and endorsements. The **Adaptablity** requirement can only be assessed through the user evaluation, detailed in Section 4.3. We discuss if this requirement was ensured in Section 4.4. Both **Security** and **Privacy** requirements were ensured for the Kiosk-Wearable technique, thanks to the POSE protocol to protect messages, presented in Section 3.3.3. We provide an in-depth evaluation of POSE, in Section 4.2. Regarding the remaining techniques, since we consider the kiosk as a trusted witness, we considered it will not disclose any type of information. The Kiosk-Smartphone technique is vulnerable to location spoofing attacks since the QR codes can be exchanged between multiple users and can generate location certificates for other users. The solution is a bystander on the spot or biometric authentication. The same technique presents a privacy vulnerability since the QR code can be read by any other malicious user and can obtain sensitive information. We leave for future

work implementation of POSE on top of QR codes.

4.2 Message Protection Evaluation

This Section presents a comparison between the developed POSE protocol and the COSE standard on the same example application. POSE was developed mainly for constrained devices and IoT applications. Thus, we evaluated the POSE processing time required to create and serialize/deserialize the exchanged messages, in Section 4.2.2, its packet overhead, in Section 4.2.2 and the CPU usage for both processing moments in Section 4.2.3. The comparison baseline was COSE and this quantitative evaluation will allow us to answer the following questions:

MP1. How large is the packet overhead of POSE in the exchanged messages compared to COSE and BLE alone?

MP2. How significant is the CPU usage of POSE when compared to COSE?

MP3. Is POSE faster than COSE in serializing/deserializing the exchanged BLE messages?

4.2.1 Processing Time

The processing time measures the required time for the serialization and deserialization of the 150 messages that have been exchanged between the smartwatch and the kiosk devices during the experiments using POSE and the standard COSE protocol. Table 4.1 presents the respective results. The creation and serialization of such messages include the following operations:

- Creation of the `Enc_Structure` object with the required protected and unprotected fields
- Generation of associated data based on the protobuf/CBOR serialization of the `Enc_Structure` to bytes
- Encryption of the payload using the AEAD algorithm
- Creation of the `POSE_Encrypt0` object inside the `Enc_Structure`.

The deserialization of the exchanged messages includes similar operations, with a decryption computation instead of an encryption one.

The obtained results show high processing time and mostly similar values in both protocols with slightly lower processing time in COSE serialization. These results indicate that the limited resources on the devices might have much more influence on the processing time rather than the application-layer protocol. Considering the lightweightness of Protocol buffer messages, the

Table 4.1: Time required to process the POSE and COSE messages.

	Serialization (ms)		Deserialization (ms)		Total Time (ms)
	avg.	*std.*	*avg.*	*std.*	*sum*
COSE	144.66	15.188	257.79	2.102	402.45
POSE	164.85	24.301	257.09	2.544	421.94

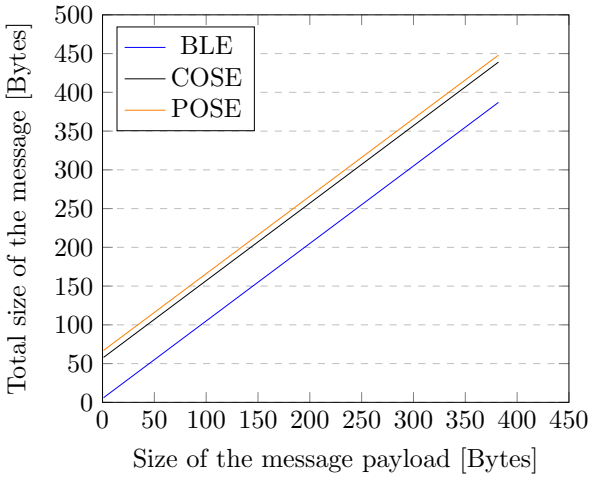


Figure 4.1: Packet overhead of exchanged messages in multiple protocols.

conversion from and to protobuf messages was expected to be much more efficient, which leads us to think that the processing power of the CPU is much more important rather the data format. POSE presents to have a slightly bigger processing time in the serialization moment than COSE, which answers the last research question. The total time required to process POSE messages is too high and can jeopardize the scalability of our protocol.

4.2.2 Packet Overhead

Packet overhead measures the additional data that POSE requires to securely exchange payloads between the prover device (smartwatch) and the witness (kiosk). In this experiment, we compared the POSE packet overhead against the standard COSE overhead and also against the default BLE with no security features (without pairing) as well as BLE pairing with authenticated LE Secure Connections [Ren19].

Figure 4.1 illustrates the obtained results. For simplicity, we merged the overhead results of the two BLE levels into just BLE, as both showed similar results in the experiments. The results were captured on the Kiosk device using Wireshark[1].

As shown in Figure 4.1, packet overhead generated by POSE protocol stays linear as the size of the application payload increases, showing around 66 bytes overhead. It is justified due to the mandatory added protected and unprotected fields of the POSE message types as well

[1]https://www.wireshark.org/

Table 4.2: CPU usage measurements on the serialization of messages

	CPU cores used	CPU execution time (%)	Wall Clock Duration (s)	CPU duration (s)
COSE	2	76.01	16.25	14.16
POSE	2	80.53	17.13	14.97

as the bytes allocated to the tag of each field of the Protobuf messages. The generated packet overhead by the COSE protocol presents the same behavior, showing around 57 bytes overhead, which is slightly less than POSE. On the other hand, the packet overhead of both BLE modes is residual, showing around 5 bytes overhead. However, the increasing packet overhead of POSE is justified by the increasing security guarantees and the added message freshness feature. Overall, the packet overhead difference between the two protocols can be seen as a trade-off between efficiency and protection against replay attacks. POSE presents a 9 bytes overhead difference compared to COSE and 61 bytes overhead when compared to both BLE modes, thus answering the first research question.

4.2.3 CPU Usage

The CPU usage metric is used to quantify how the smartwatch of the prover and the kiosk processor have been utilized when creating and serializing/deserializing data packets. High CPU usage may indicate that the protocol has high demands for processing power. The experiment here is similar to the processing time experiment, described in section 4.2.1. We collected measurements of 150 messages. These messages were generated on the wearable device (smartwatch) and read through BLE on the kiosk device. The reading cycle period for each message was 4 seconds, to allow for the smartwatch to restart Bluetooth since each interaction needs to start from the radio activation. During this experiment, we obtained CPU usage measurements on both constrained devices: CPU usage of the main thread responsible for the data payload on the smartwatch and the CPU usage on the receiving kiosk device. We divided the evaluation into the following sections based on the main processes of the protocol.

Serialization

Table 4.2 presents the results obtained for the CPU usage using the Android Profiler [2] tool.

Our SurePresence implementation for this experiment ran using two cores. The information about the frequency of the CPU cores was not available for this kind of equipment. The CPU execution time gives us information about the percentage of time the CPU was used to execute the scheduled job when compared to the main thread. This means that 76.01% of the total

[2]https://developer.android.com/studio/profile/android-profiler

Table 4.3: CPU usage on the deserialization of the POSE and COSE messages.

	Load average on Deserialization (%)					
	Last 1 Minute		Last 5 Minutes		Last 15 Minutes	
	avg.	*std.*	*avg.*	*std.*	*avg.*	*std.*
COSE	0.310	0.1254	0.416	0.2534	0.306	0.1753
POSE	0.463	0.4685	0.384	0.2938	0.296	0.1418

CPU execution time of the main thread of the application was serializing the COSE messages. Regarding this metric, the results for both protocols were similar. The Wall Clock Duration is the total elapsed real-time of the serialization, while the CPU duration is the total time the serialization process consumed CPU resources. The Wall Clock Duration includes the time the main thread responsible for the experiment was idling.

Deserialization

The total waiting time of the deserialization part was 10 min (4s x 150 samples); while the total deserialization time was 21 seconds (21699 ms) for all messages. We obtained the CPU usage during the whole experiment using the uptime[3] Linux command, with a polling rate of 1 second. Therefore, the more meaningful and more representative measurement is the Last 1 Minute one, which you can see in Table 4.3.

The reference value for CPU full utilization is 4.0 since the Raspberry Pi 4 is quad-core. The average CPU usage values measured for the last 1 minute for both protocols are similar, justified by the overlapping confidence intervals. The CPU utilization in both COSE and POSE is 0.310 and 0.463, respectively which is low and shows the CPU bottleneck is not a problem for POSE, regarding constrained devices.

Regarding the obtained results for the serialization moment, the CPU usage of both protocols is very similar, consuming similar resources, with close execution times. We leave for future work a more thorough analysis of the CPU cores used behavior.

Regarding the obtained results for the deserialization moment, the CPU usage in the conversions of both protocols is low and shows the CPU bottleneck is not a problem for POSE, regarding constrained devices. These results were expected considering the complexity of the exchanged messages. For both conversion moments, the CPU bottleneck of POSE was similar to COSE, thus answering the second research question.

[3]https://man7.org/linux/man-pages/man1/uptime.1.html

4.3 User Evaluation

We start by outlining the research questions we aim to answer with this evaluation. In Section 4.3.2 we do a characterization of the users recruited for the user study. In Section 4.3.3 we describe the relevant materials used in the user study, as well as the environment it took place. In Section 4.3.4 we detail the steps of the procedure that conducted the study. The information that was collected throughout the user study is described in Section 4.3.5. In Section 4.3.6 we specify the design of the study and how we analyze its resulting information. Finally, in Section 4.3.7 and Section 4.3.8 we present the results obtained from both evaluation methods.

4.3.1 Research Questions

We conducted a user study to help us answer the following research questions regarding our implementation:

RQ1. Which location proof technique is the user more comfortable with?

RQ2. Which technique has the smallest time per action ratio?

RQ3. What is the effectiveness of each novel location proof technique?

RQ4. From the perspective of the user, which technique is less vulnerable to the presented threats?

4.3.2 Participants

We recruited 32 participants in the Instituto Superior Técnico Alameda Campus, where all of them were students from different courses of this institution and were randomly approached on the campus. Their average age was 21.28 (*std.* = 2.247), where the majority were men (56.3%). Only 1 participant had been weekly using a touchable device (3.1%) while the remaining had been using it daily (96.9%). The majority of the participants (59.4%) had never used a smartwatch while 40.6% had already used one with basic applications, including fitness and weather ones.

4.3.3 Apparatus

The user study took place in a room in the Alameda Campus of the Instituto Superior Técnico Alameda Campus with a consistent power supply and Wi-Fi connection. The interactive kiosk was built using a Raspberry Pi 4 equipped with a 720x480p resolution touchable screen, a Pi camera, and an external USB smart card reader. We used a dummy citizen card to simulate

the kiosk-only location proof technique. A Huawei Watch 2 was used to simulate the smartwatch necessary for the Wearable technique. A Samsung Galaxy S9 was used to simulate the smartphone necessary for the Smartphone technique.

4.3.4 Procedure

At the beginning of the study, each participant was told to read a description of the overall purpose of the study, which can be seen in Appendix A.1. After reading it, all remaining doubts and questions the participants could have about the study were cleared.

Following this, the participants were asked to complete the authentication ceremony for each location proof technique, randomly ordered. Once the authentication ceremony was over, they would move on to the next technique. During each ceremony, the number of actions performed by each participant was being noted, as well as the time needed to complete it.

After completing the three techniques, the participants were given a questionnaire to complete regarding the experience. The questionnaire started with the user characterization section. The following section was the User Experience Questionnaire (UEQ) where the participant had to classify the authentication ceremony of each technique using a 5-point Likert scale [Lik32] (1 - Totally disagree; 5 - Totally agree) for each User Experience metric, represented by a phrase:

1. **Easiness** - "It was easy to use"
2. **Fastness** - "It was fast to use"
3. **Comprehensiveness** - "It was easy to understand"
4. **Stress** - "It was a stressing experience"
5. **Trust** - "Felt confidence in the whole process"
6. **Security** - "Felt the authentication method was safe"
7. **Privacy** - "Felt my privacy assured"

The following section of the questionnaire aimed to understand the perception of the participants on multiple threats for the different authentication ceremonies. The participants were asked to classify the level of vulnerability of each technique using a 5-point Likert scale (1 - Not vulnerable; 5 - Very vulnerable) regarding each threat represented by a storyboard. The first threat was a Man-In-The-Middle attack, which respective storyboard can be seen in Figure 4.2. This attack was described to the users as the following: "In the three authentication methods you previously tested, you had to directly communicate with the Kiosk, either physically or

using one of three devices. Imagine that an attacker is trying to intercept all communication you make with the Kiosk, trying to steal all exchanged messages, and trying to obtain access to all types of information, for example, about the application, the location, or even you."

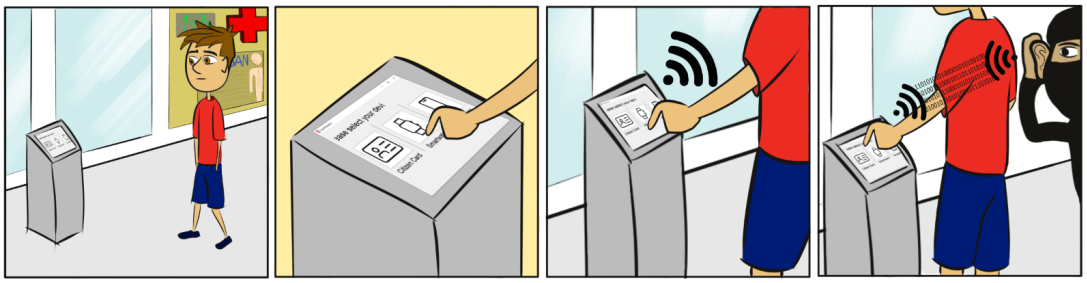


Figure 4.2: Man-In-The-Middle threat storyboard.

The second threat was user information disclosure, which can be seen in Figure 4.3 and the last threat was location spoofing, which is displayed in Figure 4.4. This attack was described as the following: "Imagine that an attacker is trying to obtain, using a fake Kiosk or through message interception, sensitive information about you, for example, your name, email, password, location, citizen card ID, or another type of information."

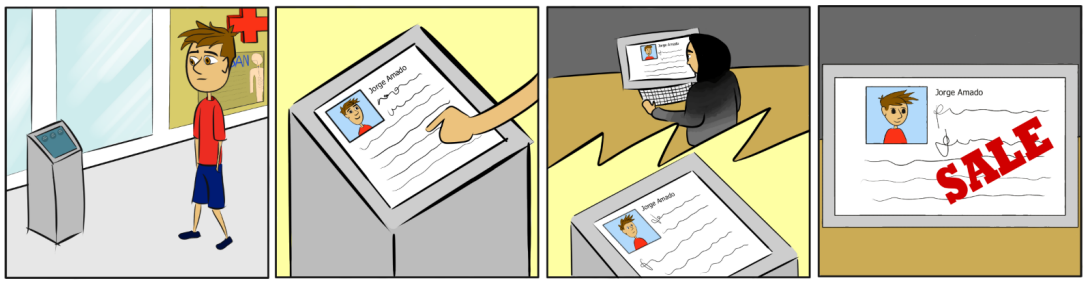


Figure 4.3: User information disclosure threat storyboard.

The third threat was a location spoofing attack, which respective storyboard can be seen in Figure 4.4. The narrative for this attack was the following: "Imagine that you are trying to fake your presence in the Kiosk location, being simultaneously in another remote location, with the help of a friend that would authenticate as if he were you."

The final section of the questionnaire was destined to understand if SurePresence was well contextualized with the medical appointment use case and in which other real use cases it could be helpful. The full questionnaire can be seen in Appendix A.2.

After completing the questionnaire, the participants were asked three questions regarding their favorite authentication ceremony, the safest authentication method and in which other contexts could SurePresence be used (not mandatory since was already in the questionnaire).

Figure 4.4: Location spoofing threat storyboard.

This small interview can be seen in Appendix A.3.

4.3.5 Dependent Measures

We collected both quantitative and qualitative data.

Quantitative

During the execution of the authentication ceremony task of each location proof technique we collected several quantitative values:

- Time needed to conclude the ceremony;
- Total number of actions performed;
- Successfullness of the ceremony.

Regarding the User Experience section of the questionnaire, we collected the values of the Likert scale for each metric previously mentioned in Section 4.3.4.

Regarding the threat perception section of the questionnaire, we collected the vulnerability values of the Likert scale of each technique regarding each threat.

Qualitative

We collected data regarding the answers from the last section of the questionnaire and the interview. This data includes:

- Usefulness of the SurePresence system (5-point Likert scale);
- If SurePresence makes sense in a medical context;
- If SurePresence is a good alternative to the current medical office solutions;
- In which other contexts can SurePresence be used;

Table 4.4: Average time and number of actions results for all three location proof techniques authentication ceremonies.

	Citizen Card		Wearable		Smartphone		Friedman Test Statistic		
	avg.	*std.*	*avg.*	*std.*	*avg.*	*std.*	*chi-square*	*df*	*p-value*
Nr. actions	3.06	0.354	7.47	0.842	7.19	0.738	55.143	2	0
Time (s)	12.74	3.838	25.53	10.589	48.28	19.429	54.813	2	0

- Favourite location proof technique of the participants;
- Which authentication ceremony is safer from the point of view of the participant.

4.3.6 Design and Analysis

Regarding the quantitative measures mentioned in Section 4.3.5, we started by analyzing them using descriptive statistics, including average values, respective standard deviation, and confidence interval. Then, we compared the same measurements against a within-subject factor using three conditions, which represent the three different location proof techniques, where the participants only had one trial for each condition. This was done using the non-parametric Friedman test [Fri37] since we do not guarantee the normality of the samples and the measured dependent variable is at the ordinal level (5-point Likert-Scale). For the measurements that showed no statistical differences between the different conditions, we applied post-hoc tests using Mann-Whitney tests [MY16] with Bonferroni correction [Hay13].

4.3.7 Quantitative Evaluation

We did a quantitative evaluation of the results obtained from the user study. We first analyze the results of the measurements previously described in Section 4.3.5 where we refer to the first two points as authentication metrics, as shown in Section 4.3.7 and evaluate the results obtained from the User Experience section of the questionnaire in Section 4.3.7. Finally, we evaluate the results obtained on the perceived vulnerability of each technique on the presented threats.

Authentication Metrics

The results obtained for the average number of actions and the average time required to successfully perform each location proof technique can be seen in Table 4.4. The average number of actions required to complete the Wearable and the Smartphone techniques are too similar, as you can see by the overlapping confidence intervals, showcased in Figure 4.5 while regarding the time metric, the results were too different. As previously described, we performed a within-subject factor comparison using three conditions (techniques) for both metrics.

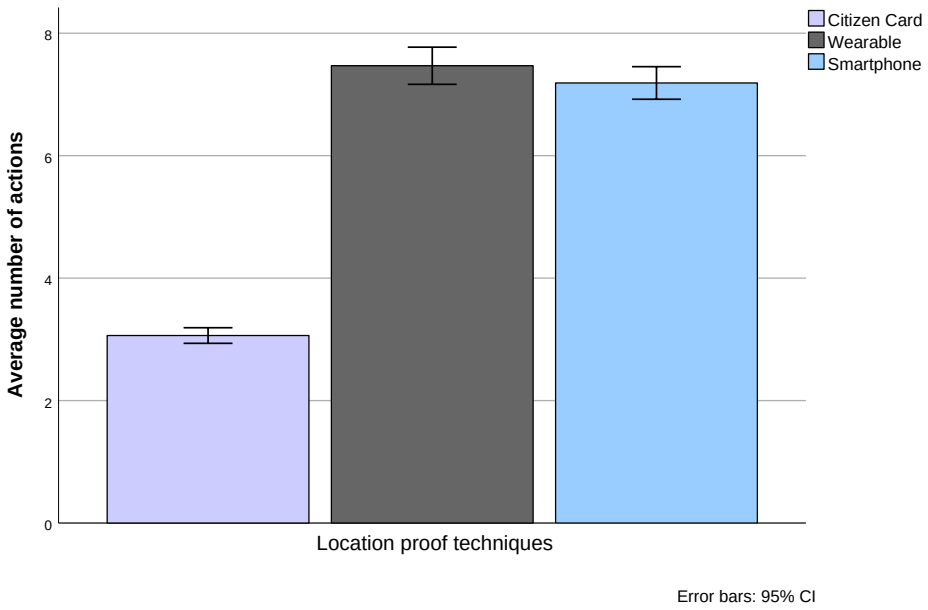


Figure 4.5: Average number of actions of each location proof technique authentication ceremony.

A Friedman test revealed a significant effect of each technique on the number of actions and time needed for authentication ($\chi^2 = 55.143$, p-value < 0.01 and $\chi^2 = 54.813$, p-value < 0.01), respectively. Such results can also be seen in Figure 4.4. A post-hoc test using Mann-Whitney tests with Bonferroni correction, resulting in a significance level set at p-value < 0.017 ($\alpha = 0.05$, $df = 2$), showed the significant differences in the number of actions ranks between the Wearable - Citizen Card techniques ($Z = -5.071$, p-value < 0.001) and Smartphone - Citizen Card ($Z = -5.286$, p-value < 0.001). The Smartphone and Wearable techniques showed no significant statistical difference between them ($Z = -1.468$, p-value $= 0.142$). The citizen card technique requires 3.06 actions, on average, to successfully authenticate a user, which is significantly less when compared to the remaining techniques (Wearable: 7.47; Smartphone: 7.19).

A similar post-hoc test using Mann-Whitney tests with the same Bonferroni correction, showed the significant differences of time ranks between all techniques: Wearable - Citizen Card ($Z = -4.806$, p-value < 0.001), Smartphone - Citizen Card ($Z = -4.937$, p-value < 0.001) and Smartphone - Wearable ($Z = -4.619$, p-value < 0.001). The citizen card technique requires 12.74 seconds, on average, to successfully authenticate a user, which is significantly less time when compared to the remaining techniques (Wearable: 25.53s; Smartphone: 48.28s).

User Experience Metrics

The obtained average results of the User Experience metrics can be seen in Table 4.5. These results represent the perception of the users regarding the User Experience metrics, for each

Table 4.5: Average values for each User Experience metric in all three location proof techniques.

	Citizen Card		Wearable		Smartphone		Friedman Test Statistic		
	avg.	*std.*	*avg.*	*std.*	*avg.*	*std.*	*chi-square*	*df*	*p-value*
Easiness	4.75	0.803	4.44	0.801	3.66	1.066	29.718	2	0.000
Fastness	4.75	0.803	4.47	0.879	3.78	1.070	27.831	2	0.000
Compreh.	4.75	0.803	4.38	0.833	3.56	1.162	29.459	2	0.000
Stress	1.44	1.045	1.81	1.330	2.19	1.424	12.057	2	0.002
Trust	4.34	1.260	4.13	0.833	3.75	1.164	11.268	2	0.004
Security	4.34	0.983	4.16	0.987	4.31	0.965	5.148	2	0.076
Privacy	3.81	1.230	4.09	1.088	4.28	0.958	7.815	2	0.020

technique. The Citizen Card technique presented the best results for all metrics, except for the privacy one (Citizen Card: 3.81; Wearable: 4.09; Smartphone: 4.28). The Wearable technique presents more similar results to the Citizen Card technique rather than the Smartphone one. Regarding the easiness, fastness, and comprehensiveness metrics, the Smartphone technique distances itself from the Citizen Card, showcased by the not overlapping confidence intervals presented in Figure 4.6. We followed the same procedure as in the authentication metrics.

A Friedman test revealed a significant effect of each technique on all user experience metrics, except for the security one ($\chi^2 = 5.148$, p-value $= 0.076$), where it showed that for the users, the three techniques have similar security levels. These results can be also seen in Table 4.5.

A post-hoc test using Mann-Whitney tests with Bonferroni correction, resulting in a significance level set at p-value < 0.017 ($\alpha = 0.05$, $df = 2$), was done for every metric, except for security. Regarding the easiness, fastness, and comprehensiveness metrics, the test showed significant differences in their ranks between the three location proof techniques, where the p-value < 0.017 in all pairwise comparisons between techniques. Regarding the stress metric, the test only showed significant differences between the Smartphone and the Citizen Card ($Z = -2.827$, p-value $= 0.005$). These differences can also be visualized in Figure 4.6, where the Citizen Card technique was the one inducing less stress to the users. About the trust metric, the test showed no significant differences between all techniques, since all obtained p-value > 0.017. Finally, regarding the privacy metric, the test showed significant differences between the Smartphone and the Citizen Card techniques ($Z = -2.721$, p-value $= 0.007$). In Figure 4.6 we can see that the Smartphone technique was the one ensuring the users more privacy.

Threat Models

The obtained average values perceived by the users of the vulnerability of the techniques for each threat can be seen in Table 4.6. For the users, the Citizen Card technique is the least vulnerable technique regarding Man-In-The-Middle and Location Spoofing attacks, while the Wearable is

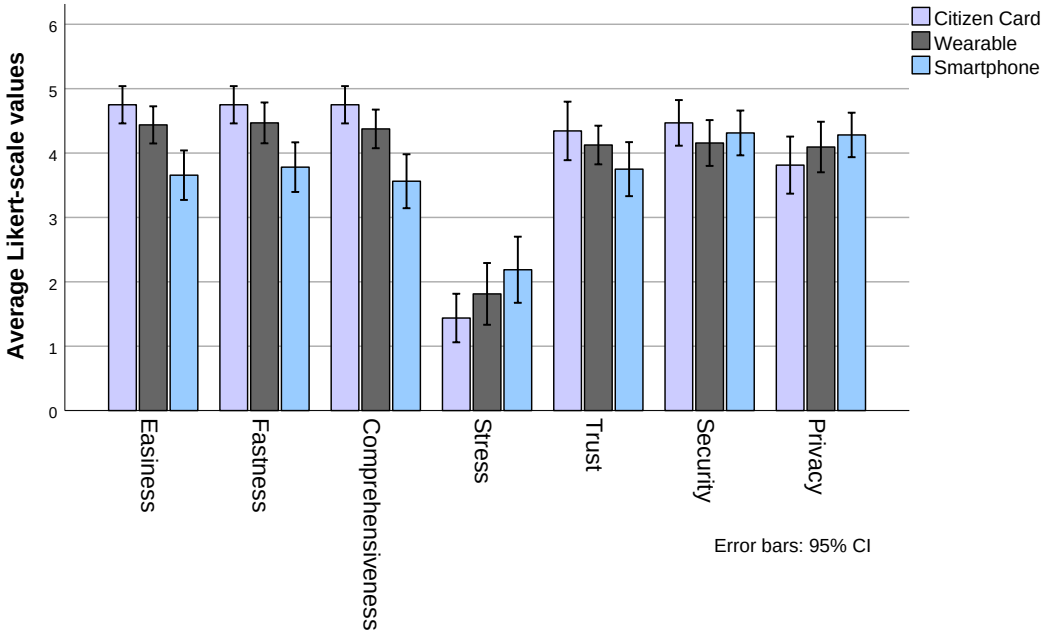


Figure 4.6: User Experience metrics plot with confidence intervals for each location proof technique.

Table 4.6: Average responses of the users perception of three threat models on all three location proof techniques.

	Citizen Card		Wearable		Smartphone		Friedman Test Statistic		
	avg.	*std.*	*avg.*	*std.*	*avg.*	*std.*	*chi-square*	*df*	*p-value*
Man-In-The-Middle	2.03	1.307	2.75	1.218	2.47	1.164	4.515	2	0.105
Information Disclosure	3.06	1.435	2.47	1.016	2.87	1.264	5.952	2	0.051
Location Spoofing	2.91	1.510	3.75	0.916	3.44	1.076	13.816	2	0.001

perceived as the most vulnerable. These roles switch for the Information Disclosure threat, which is corroborated by the previously obtained values for the privacy metric, as previously explained in Section 4.3.7. Overall, the results were similar between all techniques, showcased by the confidence intervals in Figure 4.7.

A Friedman test revealed a significant effect of the technique on the vulnerability perception of the users on the Location Spoofing attack ($\chi^2 = 13.816$, p-value < 0.01), while it did not show statistical differences between techniques for the remaining attacks, which makes the Citizen Card the least vulnerable for MITM and Information Disclosure attacks, as previously mentioned, but not by a being margin, as showcased by the overlapping confidence intervals in Figure 4.7. These results can be also seen in Table 4.6. A post-hoc test using Mann-Whitney tests with the same previous conditions (same correction and significance level) showed significant

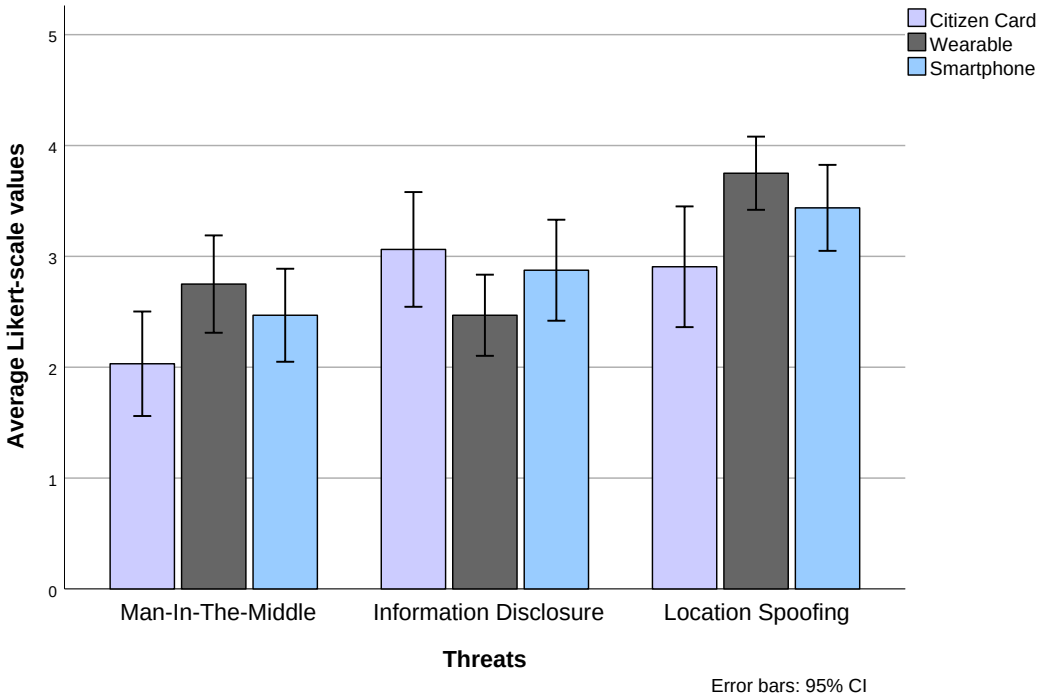


Figure 4.7: Average vulnerability values of each location proof technique for each presented threat.

vulnerability differences between the Wearable and the Citizen Card techniques ($Z = -2.922$, p-value $= 0.003$).

User Questionnaire

We asked the users how useful would this system composed of these three techniques be to obtain an excuse of absence for both work and school matters, where a value of five is very useful and a value of one is not useful at all. We obtained an average value of 4.53. We also asked if this system does make sense in the context of medical appointments, where we obtained 100% of "yes" responses. 30 out of the 32 participants (93.8%) prefer this system when compared to the usual authentication routine of showing up for a medical appointment. We can see that, for the users, this system is fit and makes perfect sense to be used in a medical context.

4.3.8 Qualitative Evaluation

In this section, we do a qualitative evaluation of the results obtained from the user study, specifically from the last section of the user questionnaire and from the interviews.

Interviews

We asked the users which of the three techniques they prefer and why. The Citizen Card technique was the favorite of the participants (N = 24): "It is the easier, faster, simpler, more convenient, safer, more efficient and more universal technique". We did a similar question but now to understand which one of the three techniques the users felt more secure. The Citizen Card technique was also the chosen one (N = 19), where 12 of those 19 responses were following this line of thought: "The citizen card presents less danger when being used since I feel I have more control on the stored sensitive data thanks to the security inherent to the smart-chip." The smartphone technique was the safer for 9 participants where the majority of the responses were: "the communications cannot be intercepted since they use code scanning, instead of wireless communications." We leave the analysis of these responses to the discussion.

Finally, we asked the participants for other contexts where this system could be used. Registering students in classes, exams, conferences, and workshops and registering workers in their workplace daily were the majority of the answers.

4.4 Discussion

We now discuss the results obtained from the SurePresence evaluation done through the user study and analyze RQ1 through RQ4 (stated in Section 4.3.1).

Which location proof technique is the user more comfortable with? The Citizen Card technique is the most straightforward technique when looking at the authentication metrics, as would be expected since the number of actions in an optimal authentication with the citizen card is inferior to the number of actions needed for the remaining techniques. Such difference to the remaining techniques was shown by the users and can be seen in Figure 4.5, where the confidence intervals of the remaining techniques are not even near the Citizen Card one. The most complex action the users needed to complete was to insert the citizen card in a specific way, which was clear from the instructions displayed in the kiosk interface. Looking at the user experience metrics, the citizen card was the clear winner, where the Friedman test revealed a significant distance for the remaining techniques regarding all metrics. Complementing that, the average values displayed in Table 4.5 showed superiority. The perception of the users is correct when talking about the first three user experience metrics. The Citizen Card technique is the easiest, fastest, and most comprehensive technique of the three. It only requires one action different from clicking and only has one specific screen, that contains clear and simple instructions. It is considered to be the most trustworthy technique, thanks to the trust of the

Table 4.7: Time per action ratio for each location proof technique.

	Citizen Card	Wearable	Smartphone
Time per action (s)	4.16	3.42	6.71

users in the security inherent to the smart-chip, developed by the Portuguese government. The citizen card is already used to authenticate patients in hospitals and clinics, supporting such a claim. Users trust this device and show no problems using it, which is shown by the trust metric difference between techniques in Figure 4.6. Although, regarding this metric, we can see the similarity in the confidence intervals between the Citizen Card and the Wearable technique, which makes us optimistic about the usage of wearables in a hospital context. The only negative result regarding the Citizen Card technique is the Mann-Whitney test between the citizen card and the smartphone technique, regarding the privacy metric, as mentioned in Section 4.3.7. Even if we did an overall average value, giving the same weight to all metrics, the Citizen Card technique would still be the technique providing a better user experience (Citizen Card: 4.33; Wearable: 4.12; Smartphone: 3.74). Finally, combining all of this with the answers from the first question of the interviews, where the Citizen Card technique was the preferred one (24 out of the 32 participants), we conclude the kiosk-only (or citizen card) technique is the one the user feels more comfortable, thus answering the first research question and ensuring the **adaptability** non-functional requirement from Section 3.2.2.

Which technique has the smallest time per action ratio? The time needed for a user to authenticate is directly related to the number of actions he needs to perform. The total time encompasses the time to understand the user interface, to know what to do next, to complete specific actions, and the time the system takes to process each action. From this perspective, it is understandable that the Mann-Whitney tests between the Wearable and the Smartphone techniques for both authentication metrics showed different results. The actions of having a QR Code being scanned and scanning another one are much more complex than just selecting a device from a list. Taking both metrics into account in an independent way and getting conclusions on the efficiency of the techniques is unfair. We need to look at the time per clicks ratio to reach more meaningful conclusions. This ratio represents the time needed to act (click, insert or scan).

In Table 4.7 we can see the time per action ratio, in seconds, for each location proof technique. The Wearable and Smartphone techniques have a similar number of actions, as shown in Table 4.4 so this ratio tells us there is something wrong with the smartphone technique. The actions relative to the QR codes may be too difficult or the sequence of actions required may be unclear.

The kiosk interface may be badly designed or the usage of QR codes may not be adequate in the context of location certification systems. The scanning of the first QR code may take too long because of the hardware that was used. These are all possible justifications for this result. The wearable ratio is smaller than the citizen card ratio, despite the bigger number of actions which is explained by the lack of processing feedback in the kiosk interface, while it communicates with the smartwatch, which made users keep constantly clicking the "Confirm device" button and even commentating "is this working?" or "should I do anything else?". We leave this issue for future work, which can reduce the number of actions needed to conclude this technique. The processing time needed to read the public citizen card information may justify the bigger ratio for the Citizen Card technique. Regarding the second question, it is the Wearable technique that presents the smallest time per action of all three techniques. This is a promising result for the usage of IoT devices in location certification systems and is even more impressive when looking at the fact that 59.4% of the participants had never used a smartwatch before the study.

What is the effectiveness of each novel location proof technique? Regarding the effectiveness of the Smartphone technique, 37.5% of the participants failed to scan the last QR Code, not being able to complete the authentication and needing to try again. This means the smartphone technique had 62.5% effectiveness, while the remaining techniques had 100%. This answers the third research question. It shows that people are not as acquainted with QR Codes as they think since most of these participants did not know the difference between scanning a QR code and having a QR code being scanned. After having their QR code scanned, the users thought the authentication ceremony was done and they could press the "Done" button, completely ignoring the instructions displayed. Some of the users were even surprised when told their authentication was not successful. This QR code paradigm is corroborated with the results for the stress metric, with the smartphone technique being the most stressful, as you can see in Figure 4.6.

From the perspective of the user, which technique is less vulnerable to the presented threats? The obtained results regarding the security metric show *there are no statistically significant differences between techniques* but we can make a more fine-grained analysis looking at the vulnerability perception of the users on the different threats. Although there are no statistically significant differences between the techniques regarding the Man-In-The-Middle threat, the participants perceived the Wearable technique as the most vulnerable. Thanks to the implementation of the POSE protocol on top of the pairing-less BLE communications between the smartwatch and the kiosk, this fact is far from true. Lack of information on the user side may justify such perception. The Information Disclosure threat showed similar results, but with

the Citizen Card technique perceived as the most vulnerable, corroborated with the results of the privacy metric from Section 4.3.7, where this technique was considered the less private. This time, the perception of the users is correct. The Citizen Card is, in fact, the most vulnerable technique to the disclosure of private information, if we consider a bystander in the smartphone technique denying any malicious user scanning QR codes of other users. Without a bystander, both techniques are at the same vulnerability level. Regarding Location Spoofing attacks, the Wearable technique is once again perceived as the most vulnerable technique. However, thanks to the extra feedback after the interview, the participants considered that the Wearable and the Smartphone suffer from the same issue: sharing such devices with friends is extremely easy, thus allowing the creation of location certificates for others. Such opinion is more divided when it comes to the Citizen Card. Some participants consider the citizen card a unique and intransmissible device, while others consider that can be given to a friend to spoof their location. This discussion motivates the necessity of a bystander in the place of the authentication ceremony, and, if not possible, two-factor authentication using biometric data, which has the strongest guarantees against Location Spoofing attacks. We leave biometric authentication as future work. Overall, the citizen card shows to be the most secure technique for the participants regarding the three possible threats, which answers the last research question.

4.4.1 Summary

The Citizen Card technique is perceived as the most complete and most ready to be used in a medical context, only presenting one downside: user information privacy. Therefore, we have a tradeoff between an excellent user experience and privacy issues.

The wearable technique showed great and surprising results regarding user experience, but with a bigger tradeoff. Users do not feel safe when authenticating with this technique, which is solved by providing more information about the used protocols. Hiding the details of the implemented security mechanisms leads to a lack of trust [RAH+16]. However, by providing more information, the users may have sufficient knowledge on the correct actions they are required to do, decreasing the awkwardness and increasing the usability of the technique [LFC05][JZS07].

The results regarding the usage of a smartphone were surprising. The smartphone technique provides a bad user experience, not being ready to be used in location certification systems leveraging usability. This technique must be used in non-critical contexts where users may spend more time authenticating and feel safe to fail. Regarding the location spoofing attacks, a bystander in the kiosk deployment place may not be enough to counter such attacks. Biometric authentication is needed and should be used in critical situations.

Comparing the results between the Wearable and Smartphone techniques, we can say that smartwatches potentiate the functionalities and ways of generating location proofs, providing a good user experience and positively impacting the users of location certification systems, even those who had never used a smartwatch before. We could not achieve any results regarding the benefits of using a kiosk device as a witness of the system, regarding usability. We noted the users were well acquainted with kiosk devices and their functionalities, having no doubts on how to use them when executing the tasks of the user study. Besides that, we do not have any results to show. We leave for future work the implementation of the Wearable technique in smartphones to directly compare which devices provide a better user experience. To understand if the role of the kiosk as a single witness is critical in such systems, an evaluation between SurePresence and a multiple witness peer-to-peer location certification system should be done also as future work.

Chapter 5

Conclusion

Location certification systems are crucial enablers for secure and reliable location-based services. They verify information about the location of users and prevent location spoofing attacks on applications. An attempt to make such systems usable and secure is to leverage IoT and constrained devices, allowing the generation and validation of location certificates in more seamless ways, potentiating a better user experience at the authentication and in other ceremonies.

We presented SurePresence, a location certification system, based on the SureThing framework, that through the interaction of multiple ubiquitous devices with an interactive kiosk, allows a prover to generate location claims and a singular witness, to endorse such claims.

We implemented a prototype of SurePresence for a medical use case where a patient can issue location proofs when attending a medical appointment, verifying his presence through three novel kiosk-based location proof techniques using three devices: a citizen card, a smartwatch, and a smartphone. We implemented POSE, an end-to-end security layer to protect BLE exchanged messages on constrained devices. This protocol follows the COSE specification while taking full advantage of the SureThing framework internal data encoding format, Protocol Buffers, a widely used, lightweight, and interoperable data format that provides better data conversions from other data formats, with a small increase of packet overhead when compared to COSE, but with protection against replay attacks. Our results also show the low CPU usage and similar processing time as COSE on both serialization and deserialization moments.

We assessed the user experience and perceived security provided by the SurePresence system and its location proof techniques through a user study covering user experience metrics and the perceived vulnerability of each technique for the illustrated threats. The technique that uses the citizen card as the authentication device was perceived as the most usable, most secure, and most ready to be used in a medical context, with a user information privacy tradeoff. The Wearable technique presented surprising results regarding the provided user experience and the

time per action required to authenticate the user, especially when 59.4% of the participants had never used a smartwatch before the study. It was perceived as a non-secure technique, which can be solved by providing more information to the users about the implementation of our security layer. The Smartphone technique was the clear loser, presenting security issues, perceived from our security requirements assessment and bad user experience. This result was unexpected when compared to the previous technique and just shows the feasibility of wearable devices in location certification systems, leveraging the user experience of such systems when compared with smartphones.

5.1 Achievements

We developed a location certification system capable of verifying the presence of a prover in a singular location through the interaction of multiple devices with a kiosk. We contributed to the development of the different libraries of the SureThing framework, integrated into the SurePresence system, including the data types, the cryptographic algorithms, and the service API for the different system entities. We implemented SurePresence in a medical use case, where a patient, attending a medical appointment, can verify his presence, by interacting with the presence kiosk in the check-in ceremony. We implemented the POSE end-to-end security protocol to secure the BLE communications between constrained devices, more specifically between smartwatches and the kiosk. We evaluated the user experience provided by the novel location proof techniques as well as the perceived security of the SurePresence system and its location proof techniques. Finally, we evaluated our end-to-end solution to protect BLE communications, regarding its packet overhead, CPU usage, and processing time, when compared to the standard specification it follows, for the same implementation.

5.2 Future Work

Some of our assumptions are matters for future work. The login paradigm as well as the creation of an account for the wearable application must be left for future work since it would require the registering operation to happen in the smartphone and exchange information with the wearable. After the login, the prover uses an authentication token that should be inside both the location claims and endorsements, instead of the email. We also leave for future work a Certificate Authority deployment to issue certificates for the kiosk and the verifier, instead of previously pre-distributed ones. Regarding the location proof techniques, we leave for future work the use of extra evidence to strengthen the claims, including photos. Regarding the Kiosk-Wearable

technique, random MAC addresses should be used as the way to identify the scanned devices, instead of their names, to provide privacy and to counter possible ambiguous names of equal devices. Regarding the Kiosk-Smartphone technique, to counter location spoofing attacks, we leave a biometric authentication factor to be included in the authentication ceremony.

A comparative evaluation between a single witness and peer-to-peer witnesses should be done, not only to compare which client-server model better fits this location certification system but also to understand if the role of the kiosk is critical or it can be replaced, regarding the user experience it provides.

Finally, we leave the possibility of implementing SurePresence on other real-world use cases, including the verification of presences in reunions, conferences, and classes, as possibilities for future work.

Bibliography

[AA20] H. Alamleh and A. A. S. AlQahtani. A cheat-proof system to validate gps location data. In *IEEE International Conference on Electro Information Technology (EIT)*, pages 190–193, 2020.

[AAAA17] S. Al-Sarawi, M. Anbar, K. Alieyan, and M. Alzubaidi. Internet of things (iot) communication protocols: Review. In *8th International Conference on Information Technology (ICIT)*, pages 685–690, 2017.

[AAM19] K. Agarwal, A. Agarwal, and G. Misra. Review and performance analysis on wireless smart home and home automation using iot. In *3rd International conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC)*, pages 629–633, 2019.

[ABG+17] Gildas Avoine, Xavier Bultel, Sébastien Gambs, David Gérault, Pascal Lafourcade, Cristina Onete, and Jean-Marc Robert. A terrorist-fraud resistant and extractor-free anonymous distance-bounding protocol. In *Proceedings of the ACM on Asia Conference on Computer and Communications Security*. ACM, apr 2017.

[AGM+15] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash. Internet of things: A survey on enabling technologies, protocols, and applications. *IEEE Communications Surveys Tutorials*, 17(4):2347–2376, 2015.

[AHIC15] Hala Assal, Stephanie Hurtado, Ahsan Imran, and Sonia Chiasson. What's the deal with privacy apps? a comprehensive exploration of user perception and usability. In *Proceedings of the 14th International Conference on Mobile and Ubiquitous Multimedia*, MUM '15, page 25–36, New York, NY, USA, 2015. Association for Computing Machinery.

[AWHJ15] O. Arias, J. Wurm, K. Hoang, and Y. Jin. Privacy and security in internet of things and wearable devices. *IEEE Transactions on Multi-Scale Computing Systems*, 1(2):99–109, 2015.

[BDSG04] D. Balfanz, G. Durfee, D.K. Smetters, and R.E. Grinter. In search of usable security: five lessons from the field. *IEEE Security & Privacy Magazine*, 2(5):19–24, sep 2004.

[CCCDP13] Eyüp S. Canlar, Mauro Conti, Bruno Crispo, and Roberto Di Pietro. Crepuscolo: A collusion resistant privacy preserving location verification system. In *International Conference on Risks and Security of Internet and Systems (CRiSIS)*, pages 1–9, 2013.

[DLASBS17] S. Dodier-Lazaro, Ruba Abu-Salma, Ingolf Becker, and M. Sasse. From paternalistic to user-centred security: Putting users first with value-sensitive design. 2017.

[DLE19] Thomas Dressel, Eik List, and Florian Echtler. SecuriCast. In *Proceedings of the ACM SIGCHI Symposium on Engineering Interactive Computing Systems*. ACM, 2019.

[dSCRP20] Henrique Figueiredo dos Santos, Rui Claro, Leonardo S. Rocha, and Miguel Pardal. STOP: a location spoofing resistant vehicle inspection system. In *19th International Conference on Ad Hoc Networks and Wireless (AdHoc-Now)*, October 2020.

[FGK21] Matthias Fassl, Lea Theresa Gröber, and Katharina Krombholz. *Exploring User-Centered Security Design for Usable Authentication Ceremonies*. Association for Computing Machinery, New York, NY, USA, 2021.

[FP18] J. Ferreira and M. L. Pardal. Witness-based location proofs for mobile devices. In *IEEE 17th International Symposium on Network Computing and Applications (NCA)*, pages 1–4, 2018.

[FPAF18] M. Frustaci, P. Pace, G. Aloi, and G. Fortino. Evaluating critical security issues of the iot world: Present and future challenges. *IEEE Internet of Things Journal*, 5(4):2483–2495, 2018.

[Fri37] Milton Friedman. The use of ranks to avoid the assumption of normality implicit in the analysis of variance. *Journal of the American Statistical Association*, 32(200):675–701, 1937.

[GKRT14] S. Gambs, M. Killijian, M. Roy, and M. Traoré. Props: A privacy-preserving loca-

tion proof system. In *IEEE 33rd International Symposium on Reliable Distributed Systems*, pages 1–10, 2014.

[GUMS18] M. G. Gurubasavanna, S. Ulla Shariff, R. Mamatha, and N. Sathisha. Multimode authentication based electronic voting kiosk using raspberry pi. In *2nd International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC)I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC), 2018 2nd International Conference on*, pages 528–535, 2018.

[Hay13] Winston Haynes. Bonferroni correction. In *Encyclopedia of systems biology*, pages 154–155. Springer New York, 2013.

[HLR11] W. He, X. Liu, and M. Ren. Location cheating: A security challenge to location-based social network services. In *31st International Conference on Distributed Computing Systems*, pages 740–749, 2011.

[Jen19] Jakob Jenkov. Rion vs. json vs. protobuf vs. messagepack vs. cbor. `http://tutorials.jenkov.com/rion/rion-performance-benchmarks.html`, September 2019. Online.

[JZS07] Audun Jøsang, Muhammed Al Zomai, and Suriadi Suriadi. Usability and privacy in identity management architectures. In *Proceedings of the Fifth Australasian Symposium on ACSW Frontiers - Volume 68*, ACSW '07, page 143–152, AUS, 2007. Australian Computer Society, Inc.

[KCVGAZ15] Vasileios Karagiannis, Periklis Chatzimisios, Francisco Vázquez-Gallego, and Jesús Alonso-Zárate. A survey on application layer protocols for the internet of things. *Transaction on IoT and Cloud Computing*, pages 9–18, 2015.

[KF10] Gurpreet Kaur and Mohammad Muztaba Fuad. An evaluation of protocol buffer. In *Proceedings of the IEEE SoutheastCon 2010 (SoutheastCon)*, pages 459–462, March 2010.

[KM18] H. Kaur and S. Malhotra. Use of "kiosks" as a self service tools in libraries. In *5th International Symposium on Emerging Trends and Technologies in Libraries and Information Services (ETTLIS)*, pages 269–271, 2018.

[LFC05] Simson Garfinkel Lorrie Faith Cranor. *Security and Usability: Designing Secure Systems That People Can Use*. O'Reilly Media, 2005.

[Lik32] R Likert. A technique for the measurement of attitudes. *Archives of Psychology*, 22(140):5–55, 1932.

[LZ19] Karim Lounis and Mohammad Zulkernine. Bluetooth low energy makes "just works" not work. In *3rd Cyber Security in Networking Conference (CSNet)*. IEEE, oct 2019.

[MY16] Thomas W. MacFarland and Jan M. Yates. Mann–whitney u test. pages 103–132. Springer International Publishing, 2016.

[NC20] Giuseppe Nebbione and Maria Carla Calzarossa. Security of IoT application layer protocols: Challenges and findings. *Future Internet*, 12(3):55, mar 2020.

[NSY+20] M. R. Nosouhi, K. Sood, S. Yu, M. Grobler, and J. Zhang. Pasport: A secure and private location proof generation and verification framework. *IEEE Transactions on Computational Social Systems*, 7(2):293–307, 2020.

[PBB+17] John Padgette, John Bahr, Mayank Batra, Marcel Holtmann, Rhonda Smithbey, Lidong Chen, and Karen Scarfone. *Guide to Bluetooth Security*. Special Publication (NIST SP), National Institute of Standards and Technology, 2017.

[PPMT16] Srdan Popic, Drazen Pezer, Bojan Mrazovac, and Nikola Teslic. Performance evaluation of using protocol buffers in the internet of things communication. In *International Conference on Smart Systems and Technologies (SST)*. IEEE, oct 2016.

[RAH+16] Scott Ruoti, Jeff Andersen, Scott Heidbrink, Mark O'Neill, Elham Vaziripour, Justin Wu, Daniel Zappala, and Kent Seamons. *"We're on the Same Page": A Usability Study of Secure Email Using Pairs of Novice Users*, page 4298–4308. Association for Computing Machinery, New York, NY, USA, 2016.

[Ren16] Kai Ren. Bluetooth pairing part 2 key generation methods. `https://www.bluetooth.com/blog/bluetooth-pairing-part-2-key-generation-methods/`, 2016. Online.

[Ren19] Kai Ren. Bluetooth pairing part 4: Bluetooth low energy secure connections – numeric comparison. `https://www.bluetooth.com/blog/bluetooth-pairing-part-4/`, 2019. Online.

[SCRH20a] A. M. H. Sad, M. M. S. Choyon, A. H. M. Rhydwan, and C. A. Hossain. An interactive low-cost smart assistant system: Information kiosk as plug play device.

In *27th Conference of Open Innovations Association (FRUCT)*, pages 193–199, 2020.

[SCRH20b] Asm Mehedi Hasan Sad, Md Mashrur Sakib Choyon, Abu Hasnat Md Rhydwan, and Chowdhury Akram Hossain. An interactive low-cost smart assistant system: Information kiosk as plug amp; play device. In *27th Conference of Open Innovations Association (FRUCT)*, pages 193–199, 2020.

[SHWR16] Svenja Schröder, Markus Huber, David Wind, and Christoph Rottermanner. When signal hits the fan: On the usability and security of state-of-the-art secure mobile messaging. In *1st European Workshop on Usable Security*, Proceedings of 1st European Workshop on Usable Security, July 2016.

[SKS18] Byung-Yoon Sung, Ki-Bbeum Kim, and Kyung-Wook Shin. An AES-GCM authenticated encryption crypto-core for IoT security. IEEE, jan 2018.

[SW09] Stefan Saroiu and Alec Wolman. Enabling new mobile applications with location proofs. In *Proceedings of the 10th Workshop on Mobile Computing Systems and Applications*, HotMobile '09, New York, NY, USA, 2009. Association for Computing Machinery.

[THM15] W. Trappe, R. Howard, and R. S. Moore. Low-energy security: Limits and opportunities in the internet of things. *IEEE Security Privacy*, 13(1):14–21, 2015.

[Tjä17] Hampus Tjäder. End-to-end security enhancement of an iot platform using object security. Master's thesis, Linköping University, Information Coding, 2017.

[TMTG13] Rohan Tabish, Adel Ben Mnaouer, Farid Touati, and Abdulaziz M. Ghaleb. A comparative analysis of BLE and 6lowpan for u-HealthCare applications. In *7th IEEE GCC Conference and Exhibition (GCC)*. IEEE, nov 2013.

[UHM11] Dieter Uckelmann, Mark Harrison, and Florian Michahelles, editors. *Architecting the Internet of Things.* Springer, 2011.

[VTR+14] Mališa Vučinić, Bernard Tourancheau, Franck Rousseau, Andrzej Duda, Laurent Damon, and Roberto Guizzetti. Oscar: Object security architecture for the internet of things. In *Proceeding of IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks 2014*, pages 1–10, 2014.

[WNK+20] Jianliang Wu, Yuhong Nan, Vireshwar Kumar, Dave (Jing) Tian, Antonio Bianchi, Mathias Payer, and Dongyan Xu. BLESA: Spoofing attacks against

reconnections in bluetooth low energy. In *14th USENIX Workshop on Offensive Technologies (WOOT 20)*. USENIX Association, August 2020.

[WPZM16] Xinlei Wang, Amit Pande, Jindan Zhu, and Prasant Mohapatra. STAMP: Enabling privacy-preserving location proofs for mobile users. *IEEE/ACM Transactions on Networking*, 24(6):3276–3289, dec 2016.

[YSAz16] Muneer Bani Yassein, Mohammed Q. Shatnawi, and Dua' Al-zoubi. Application layer protocols for the internet of things: A survey. In *International Conference on Engineering & MIS (ICEMIS)*. IEEE, sep 2016.

[ZC11] Z. Zhu and G. Cao. Applaus: A privacy-preserving location proof updating system for location-based services. In *Proceedings IEEE INFOCOM*, pages 1889–1897, 2011.

[ZP14] W. Zhou and S. Piramuthu. Security/privacy of wearable fitness tracking iot devices. In *9th Iberian Conference on Information Systems and Technologies (CISTI)*, pages 1–5, 2014.

[ZS96] Mary Ellen Zurko and Richard T. Simon. User-centered security. In *Proceedings of the 1996 workshop on New security paradigms - NSPW '96*. ACM Press, 1996.

[ZWD+20] Yue Zhang, Jian Weng, Rajib Dey, Yier Jin, Zhiqiang Lin, and Xinwen Fu. Breaking secure pairing of bluetooth low energy using downgrade attacks. In *29th USENIX Security Symposium (USENIX Security 20)*, pages 37–54. USENIX Association, August 2020.

Appendix A

User Study Documents

A.1 Consent document

The user study you are about to participate in is in the scope of a Information Systems and Computer Engineering Master Thesis in Instituto Superior Técnico.

The main objective of this thesis is the creation of Location Proofs through the interaction of different IoT devices. A location proof is something that proves the presence of someone in a certain place, in a certain time and that might be verified by a witness. In a theoretical exam, for example, a student's presence is verified by a professor through his signature in the exam sheet.

Here, the context is medical appointments, where the patient has to verify his presence in a clinic or a hospital so that he can gather a collection of legal documents like excuse of absence or insurance papers. The patient when authenticating towards a Kiosk, creates a Location Proof which is verified by the Kiosk at that moment. Three different devices can be used to authenticate the patient.

The system you are about the test is called SurePresence.

This study is divided into three parts.

In the first one, you have to authenticate yourself towards the Kiosk using three different devices.

In the second part, you will be asked to answer a questionnaire related to the experience of the first part.

Finally, I will do a quick interview where you will be asked 3 questions.

If you have any doubt, let me know!

We will not collect any sensitive information about yourself.

A.2 Questionnaire document

SurePresence Questionnaire

This questionnaire has the goal of determining the user perception relatively to certain aspects of usability, user experience, and security/privacy issues after authenticating himself with different devices in the creation of location proofs using the SurePresence system.

Duration: 5 min

*Required

1. What is your gender? *

Mark only one oval.

- ◯ Female
- ◯ Male
- ◯ Other: _____

2. What is your age? *

_____

3. If you have a touchable device, how often do you use it? *

Mark only one oval.

- ◯ I don't have a touchable screen
- ◯ Monthly
- ◯ Weekly
- ◯ Daily

4. Have you ever used a Smartwatch? *

Mark only one oval.

- ◯ Yes
- ◯ No

5. If you answered "Yes" in the previous question, tell us which Smartwatch applications you used.

Tick all that apply.

- [ ] Watch (Cronometer/Alarm/Hours)
- [ ] Meteorology
- [ ] Fitness
- [ ] Maps
- [ ] Calls
- [ ] Calendar

Other: [ ] _____

User Experience

This section is destined to characterize the SurePresence system. You will classify each authentication method you previously tested.

6. Classify the Citizen Card authentication method relative to the following phrases. *

Mark only one oval per row.

	1 - Totally disagree	2	3	4	5 - Totally agree
It was easy to use	◯	◯	◯	◯	◯
It was fast to use	◯	◯	◯	◯	◯
It was easy to understand	◯	◯	◯	◯	◯
It was a stressing experience	◯	◯	◯	◯	◯
Felt confidence in the whole process	◯	◯	◯	◯	◯
Felt the authentication method was safe	◯	◯	◯	◯	◯
Felt my privacy assured	◯	◯	◯	◯	◯

7. Classify the Smartwatch authentication method relative to the following phrases. *

Mark only one oval per row.

	1 - Totally disagree	2	3	4	5 - Totally agree
It was easy to use					
It was fast to use					
It was easy to understand					
It was a stressing experience					
Felt confidence in the whole process					
Felt the authentication method was safe					
Felt my privacy assured					

8. Classify the Smartphone authentication method relative to the following phrases. *

Mark only one oval per row.

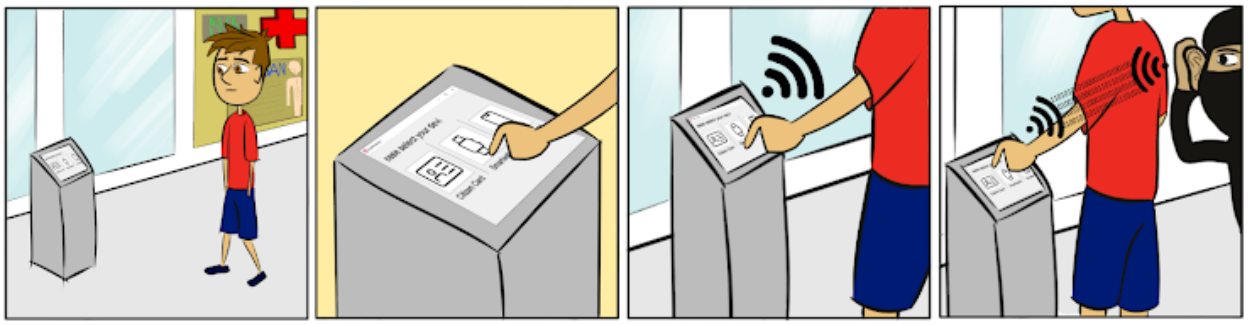
	1 - Totally disagree	2	3	4	5 - Totally agree
It was easy to use					
It was fast to use					
It was easy to understand					
It was a stressing experience					
Felt confidence in the whole process					
Felt the authentication method was safe					
Felt my privacy assured					

Threat Perception

The following sections are destined to elucidate the user about different threats and attacks that may affect the SurePresence system and its authentication methods.

Interception of the communication with the Kiosk

In the three authentication methods you previously tested, you had to directly communicate with the Kiosk, either physically or using one of three devices. Imagine that an attacker is trying to intercept all communication you make with the Kiosk, trying to steal all exchanged messages, and trying to obtain access to all types of information, for example, about the application, the location, or even you.



9. According to your opinion, classify how vulnerable is each authentication method relative to the showed threat. *

Mark only one oval per row.

	1 - Not vulnerable	2	3	4	5 - Very vulnerable
Citizen Card	◯	◯	◯	◯	◯
Smartwatch	◯	◯	◯	◯	◯
Smartphone	◯	◯	◯	◯	◯

Obtaining valuable user information

Imagine that an attacker is trying to obtain, using a fake Kiosk or through message interception, sensitive information about you, for example, your name, email, password, location, citizen card ID, or another type of information.

10. According to your opinion, classify how vulnerable is each authentication method relative to the showed threat. *

Mark only one oval per row.

	1 - Not vulnerable	2	3	4	5 - Very vulnerable
Citizen Card	◯	◯	◯	◯	◯
Smartwatch	◯	◯	◯	◯	◯
Smartphone	◯	◯	◯	◯	◯

Presence simulation with the help of a friend

Imagine that you are trying to fake your presence in the Kiosk location, being simultaneously in another remote location, with the help of a friend that would authenticate as if he were you.

11. According to your opinion, classify how easy it would be to simulate your location for each authentication method. *

Mark only one oval per row.

	1 - Not easy	2	3	4	5 - Very easy
Citizen Card	◯	◯	◯	◯	◯
Smartwatch	◯	◯	◯	◯	◯
Smartphone	◯	◯	◯	◯	◯

Contextualization

This section is destined to understand if the SurePresence system and its location proof techniques with different authentication methods are well fit to be used in the context of the medical appointment and, in the future, in other contexts.

12. How useful is this system to obtain excuses of absence to show in work or school? *

Mark only one oval.

	1	2	3	4	5	
Not useful	◯	◯	◯	◯	◯	Very useful

13. Do you think it makes sense to use this system in the context of medical appointments? *

Mark only one oval.

- Yes
- No

14. Would you rather use this verification system instead of walking to a counter and verify your presence with a person? *

Mark only one oval.

- Yes
- No

15. In which other contexts can this system be used?

A.3 Interview document

SurePresence Questionnaire Interview

1. Which device do you prefer to authenticate with and why?
2. Which authentication method do you feel safer with and why?
3. In which other contexts could this system and these techniques be used?