iscte

UNIVERSITY
INSTITUTE
OF LISBON

Enhancing E-ID cards authentication with NFC

Tariq Youssef Costa do Vale

Master in Computer Science

Supervisor:
PhD Carlos Coutinho, Assistant Professor,
ISCTE – IUL

October, 2023

[This page has been intentionally left blank]

TECHNOLOGY
AND ARCHITECTURE

Department of Information Science and Technology

Enhancing E-ID cards authentication with NFC

Tariq Youssef Costa do Vale

Master in Computer Science

Supervisor:
PhD Carlos Coutinho, Assistant Professor,
ISCTE – IUL

October, 2023

[This page has been intentionally left blank]

Enhancing E-ID cards authentication with NFC

[This page has been intentionally left blank]

To my beloved mother and father, grandparents, brother and uncle. Thank you for being my pillars of support and my source of motivation. Words fall short to express my gratitude, yet the heart knows.

[This page has been intentionally left blank]

Acknowledgements

First and foremost, I am immensely grateful to Allah (SWT) for the blessings and opportunities bestowed upon me throughout my life, which have played a crucial role in reaching this academic milestone.

I want to express my sincere gratitude to my family, whose constant support of and encouragement for my academic endeavors served as a foundation. Through every obstacle I've faced on my road, I've been driven forward by their love and sacrifices.

I owe a great deal of gratitude to my supervisor, Prof. Dr. Carlos Coutinho, whose excellent advice, support, and belief in my talents were crucial in helping me finish this thesis. His knowledge, compassion, and commitment to fostering my academic development have had a lasting impression on my educational experience and enabled the fulfillment of this thesis.

I would also like to extend my heartfelt thanks to Caixa Mágica for all the support and expertise provided to me during the development of this thesis. Their invaluable insights and assistance played a significant role in shaping the outcome of this research.

The process of writing this thesis has been a team effort, and I am appreciative of the encouragement and support that each person who has contributed to this chapter of my life has provided.

[This page has been intentionally left blank]

Abstract

In contemporary times, the profound impact of Information and Communication Technologies (ICT) in daily life introduces numerous challenges for the average citizen. These challenges encompass managing digital profiles across a variety of services and institutions, from retail shops to tax obligations. The escalating demand for authentication services for even basic tasks can be daunting. In response, many countries have devised sophisticated solutions to centralize authentication services around a single Electronic Identification (E-ID) card. This card facilitates diverse tasks such as paying taxes, obtaining criminal records, interacting with financial institutions, and digitally signing electronic documents. However, the inconvenience associated with using a smartcard reader for authentication and the intricacies of Mobile Digital Tokens can be overwhelming, especially for those less familiar with digital tools.

Near-field communication (NFC) is an increasingly popular technology, recognized for its simplicity and speed. This thesis proposes a secure, user-friendly, and efficient method to implement NFC-based authentication in an E-ID card. This thesis explores existing authentication methods and offers an enhanced solution. Safety remains a primary concern, and this is addressed by integrating cryptography techniques, including public key certificates and Multi-Factor Authentication (MFA), to ensure robust authentication.

Keywords:

Authentication;
E-ID Card;
NFC;
PKI;
Cryptography;
Multi-Factor Authentication;

[This page has been intentionally left blank]

Resumo

Nos dias de hoje, o profundo impacto das tecnologias de informação e comunicação no quotidiano introduz enumeros desafios para o cidadão comum. Estes desafios englobam a gestão de perfis digitais numa variedade de serviços e instituições, desde lojas de retalho até obrigações fiscais. A crescente demanda por serviços de autenticação para tarefas básicas pode ser assustadora. Em resposta, muitos países criaram soluções sofisticadas para centralizar os serviços de autenticação em torno de um único cartão de identificação eletrónica. Este cartão facilita diversas tarefas como o pagamento de impostos, a obtenção de registos criminais, a interação com instituições financeiras e a assinatura digital de documentos eletrónicos. No entanto, o incómodo associado ao uso de um leitor de cartões inteligentes para autenticação e as complexidades dos tokens digitais móveis podem ser avassaladoras, especialmente para aqueles menos familiarizados com ferramentas digitais.

O NFC é uma tecnologia com crescente popularidade, reconhecida pela sua simplicidade e rapidez. Esta tese propõe um método seguro, fácil e eficiente para implementar a autenticação baseada em NFC num cartão E-ID. Através da exploração de métodos de autenticação existentes, esta tese oferece uma solução aprimorada. A segurança permanece uma preocupação primária, e isso é abordado pela integração de técnicas de criptografia, incluindo certificados de chave pública e autenticação multi-fator para garantir uma autenticação robusta.

Palavras-chave:

Autenticação;
Cartão E-ID;
NFC;
PKI;
Criptografia;
Autenticação multi-fator;

[This page has been intentionally left blank]

Contents

List of Figures

[This page has been intentionally left blank]

List of Tables

[This page has been intentionally left blank]

Listings

[This page has been intentionally left blank]

Acronyms

CA Certification Authority.

CC Citizen Card.

CMD Chave Móvel Digital.

CRL Certificate Revocation List.

CSR Certificate Signing Request.

DB Database.

DN Distinguished Name.

DSRM Design Science Research Methodology.

E-ID Electronic Identification.

EU European Union.

ICT Information and Communication Technologies.

MFA Multi-Factor Authentication.

NFC Near-field communication.

OAP Online Authentication Process.

OCSP Online Certificate Status Protocol.

PCC Portuguese Citizen Card.

PKC Public Key Certificate.

PKI Public Key Infrastructure.

POC Proof of Concept.

PPAPs Portuguese Public Administration Portals.

RAII Resource Acquisition Is Initialization.

SDK Software Development Kit.

SSL Secure Sockets Layer.

TLS Transport Layer Security.

[This page has been intentionally left blank]

Chapter 1.

Introduction

Contents

This chapter establishes the foundation for this thesis by introducing the research problem centered around enhancing the online authentication process using E-ID cards, using NFC technology. It presents the motivation behind the research, coming from personal experiences and the broader challenges faced by citizens in interacting with online governmental services. The chapter outlines the research questions aimed at exploring the implementation, benefits, and impact of contactless authentication via E-ID cards. It also delineates the research objectives, methodology employing DSRM, and provides an outline of the thesis structure.

[This page has been intentionally left blank]

Chapter 1
Introduction

In today's digital landscape, citizens all around the world rely on their Electronic Identification (E-ID) card for a variety of tasks, from identity verification to accessing essential services. As more activities transition online, there's a growing need for a seamless and secure authentication process using an E-ID card. The importance of authentication cannot be dismissed [1]. The challenge is to ensure this process is not only fast and user-friendly but also secure to prevent potential identity theft and fraud.

However, the inconvenience associated with using a smartcard reader for authentication and the complexities of Mobile Digital Tokens can be overwhelming, especially for those less familiar with digital tools. These challenges underscore the need for a more streamlined and user-friendly solution.

This research proposes to integrate Near-field communication (NFC) technologies to the citizen E-ID card to mitigate the above described problems. NFC technology offers a promising way for enhancing the authentication process of E-ID cards. Its inherent benefits include speed, convenience and enhanced security. By integrating NFC into E-ID cards, we can promote a more trustworthy, efficient and simple interaction between citizens, governmental entities and companies.

An NFC-integrated E-ID card would streamline online authentication, enabling users to verify their identity with a single tap, followed by entering a decryption PIN and undergoing Multi-Factor Authentication (MFA). This approach not only simplifies the login process but also reinforces security by requiring the physical presence of the card, unlike other commonly available authentication methods, like the Chave Móvel Digital (CMD) in the Portuguese context. The decryption PIN serves as a crucial security measure to prevent unauthorized cryptographic operations using the E-ID card, ensuring that only authorized individuals can access and utilize the sensitive data stored on the card for authentication purposes [2]. The rational behind the inclusion of MFA in the authentication process is due to its importance in ensuring robust security, as highlighted in many studies [3][4][5].

Moreover, with the integration of NFC technology, there will no longer be a need for citizens to use the traditional smartcard reader for authentication. This eliminates an additional layer of complexity and hardware dependency, making the authentication process even more accessible and user-friendly for all.

Every individual's interaction with online platforms, especially governmental services, should be seamless, efficient, and secure. However, the reality often paints a different picture. The author has encountered numerous pains while accessing governmental online services, from the difficult authentication processes to the worry of potential security breaches, each interaction was a reminder of the evident gaps in the current system, mentioned in detail in section 2.3.3.

These personal experiences were not isolated incidents. Conversations with peers, colleagues, and fellow citizens in Portugal and other developed countries revealed a shared sentiment of frustration. Many expressed concerns about the time-consuming nature of authentication processes, while others voiced apprehensions about the security of their personal data. It became evident that the challenges the author faced were part of a larger, systemic issue affecting a significant portion of the population of multiple countries.

With a background in computer science and software engineering, the author recognized an opportunity to bridge these gaps. The world of technology is replete with innovations waiting to be used for the greater good. One such innovation, the NFC technology, stood out for its potential to improve online authentication. Its promise of quick, secure, and simple authentication resonated with the very challenges the author wants to address. This research is driven by a vision to enhance E-ID cards online authentication with NFC, making them more user-centric, secure, and efficient and thus creating a better way to authenticate ourselves online.

The relevance of this research have been acknowledged by the academic communities alike. The findings of this thesis have been selected for publication and presentation at the international Industry Sciences & Computer Sciences Innovation 2023 (ISCSI) conference. This recognition highlights the significance of exploring NFC technology for enhancing E-ID cards authentication.

1.1 Research Questions

The goal of this research is to create a seamless, secure, and user-friendly authentication process using E-ID cards. The goal is to investigate and clarify how NFC technology may improve authentication, leading to more effective and secure interactions between citizens, governmental organizations, and enterprises. The following research questions have been chosen to guide this investigative journey through the realms of contactless authentication, its implications for public administration and companies, and benefits for citizens.

[Q.1] How can contactless authentication through an E-ID card using NFC technology be well implemented (safe, easy and fast)?

[Q.2] What are the potential advantages for public administration and companies in adopting contactless authentication via E-ID cards?

[Q.3] How will citizens benefit from the contactless authentication feature of their E-ID cards?

For the first question, the focus will be on detailing the creation of a secure and efficient contactless authentication method using E-ID cards.

The second question will delve into the potential gains for public administration and companies, emphasizing enhancements in security, efficiency, and user experience.

Lastly, the third question will highlight the benefits for citizens, emphasizing the convenience and security of the new method.

1.2 Proposed solution and Objectives

The primary objective of this research is to enhance the online authentication process using E-ID cards, aiming to create a more efficient, user-friendly, and secure experience for all stakeholders involved.

1. **Citizens**: By enhancing E-ID cards Online Authentication Process (OAP), individuals will experience a more streamlined and intuitive authentication process. This will not only save time but also reduce the complexities often associated with online authentication.

2. **Government Services**: A more efficient authentication process will lead to fewer authentication-related queries, such as problems using a smart card reader, reducing the workload on government employees.

3. **Businesses**: Companies stand to benefit by adopting this secure authentication method, offering their users a trustworthy and efficient means to access services. This can enhance user trust and potentially increase user engagement.

In essence, this research aims to design an improved OAP for E-ID cards, addressing existing challenges and set the way for a digital authentication system that is fast, straightforward, and secure.

1.3 Methodology

In the course of this thesis, Design Science Research Methodology (DSRM) [6] was employed as a structured framework to guide the research process from problem identification to solution evaluation. The choice of DSRM was taken because of its proven efficacy in fostering a problem-solving approach, which was deemed of service for the nature of the challenges addressed in this research. Figure 1.1 shows a diagram representing the DSRM.

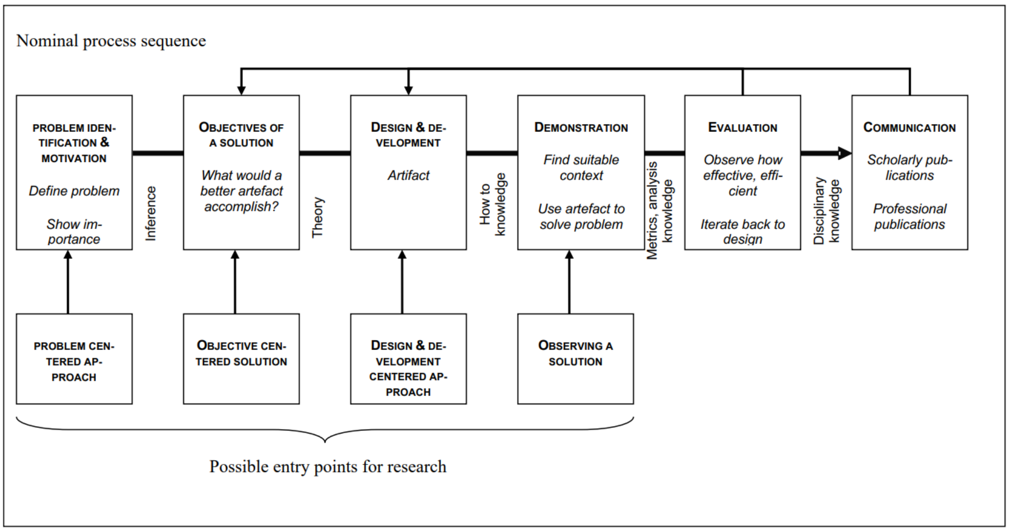

Nominal process sequence
PROBLEM IDENTIFICATION & MOTIVATION
Define problem
Show importance
Inference
OBJECTIVES OF A SOLUTION
What would a better artefact accomplish?
Theory
DESIGN & DEVELOPMENT
Artifact
How to knowledge
DEMONSTRATION
Find suitable context
Use artefact to solve problem
Metrics, analysis knowledge
EVALUATION
Observe how effective, efficient
Iterate back to design
Disciplinary knowledge
COMMUNICATION
Scholarly publications
Professional publications
PROBLEM CENTERED APPROACH
OBJECTIVE CENTERED SOLUTION
DESIGN & DEVELOPMENT CENTERED APPROACH
OBSERVING A SOLUTION
Possible entry points for research

Figure 1.1: DSRM diagram

The entry point for the implementation of DSRM in this research was the problem-centered approach. Other potential entry points include the objective-centered, development-centered, and observing a solution. The objective-centered approach focuses on achieving specific objectives, the development-centered approach emphasizes the creation of innovative artifacts, and the observing a solution emphasizes on evaluating existing design artifacts or methodologies.

The problem-centered approach was chosen as it aligns well with the nature of the challenges faced in this research. It facilitated a thorough understanding of the problem domain, which in turn guided the definition of solution objectives, the creation of the solution artifact, and the subsequent phases of the DSRM process.

The implementation of DSRM occurred through the following sequential phases, each contributing to the progressive completion of the research objectives:

1. **Problem Identification:** The initial phase entailed a thorough definition of the problem scope, highlighting the importance of enhancing online authentication mechanisms for E-ID cards.

2. **Solution Objectives Definition:** Based on the problem outline, clear objectives were set forth aiming at the development of a robust, user-centric, and efficient NFC authentication solution for E-ID cards.

3. **Artifact Creation:** A Proof of Concept (POC) was created, containing the proposed enhancements in an NFC authentication solution for E-ID cards.

4. **Solution Demonstration:** The operational efficacy of the solution artifact was demonstrated through a thorough analysis of the POC functionalities and attributes which help on addressing the identified problem in early stages.

5. **Solution Evaluation:** A comprehensive evaluation was conducted to assess the solution against the predefined objectives and the benefits to all stakeholders involved, which confirmed that the anticipated results had been achieved.

6. **Research Dissemination:** The research findings were publish and presented at an international conference called Industry Sciences & Computer Sciences Innovation 2023 (ISCSI) in order to contribute to the broader discourse of NFC authentication solutions for E-ID cards.

The initial phases of Problem Identification and Solution Objectives Definition are elaborated in Chapters 1 and 2, respectively, setting the groundwork for the research. The subsequent phases of Artifact Creation and Solution Demonstration are comprehensively tackled in Chapter 3, where the proposed solutions are delineated and their efficacy demonstrated. The Solution Evaluation phase is comprehensively tackled in Chapter 4, providing a thorough assessment of the proposed solutions against the set objectives.

The successful completion of these DSRM phases not only facilitated a systematic approach to problem-solving but also generated a substantive contribution to the field, as evidenced by the developed POC.

1.4 Thesis Outline

This thesis is organized as follows:

- **Chapter 1: Introduction** - This chapter introduces the research problem, objectives, and methodology.
- **Chapter 2: State of the art** - A review of the existing literature and state-of-the-art technologies relevant to the research problem.
- **Chapter 3: Proposed solution** - Detailed description of the proposed solution, including the design and implementation of the POC.
- **Chapter 4: Results Analysis** - Evaluation of the proposed solution against the set objectives and discussion of the results.
- **Chapter 5: Conclusion** - Summary of key findings, contributions, and suggestions for future work.

[This page has been intentionally left blank]

Chapter 2.

State of The Art

Contents

Following the DSRM methodology detailed in section 1.3, this chapter delves into the existing literature and technologies pertinent to the realm of online authentication, with a particular focus on the utilization of E-ID cards and NFC technology. It explores the current state of authentication methods in the Portuguese context and identifies their challenges. The chapter also examines the contemporary landscape of NFC authentication and its growing usage. It also explains thoroughly how can public key cryptography be use to create a safe authentication system. Finally, it delves into the Portuguese Public Key Infrastructure currently used.

[This page has been intentionally left blank]

Chapter 2
State of The Art

The analysis will be divided into the following topics in order to examine the state-of-the-art of this research topic:

- Introduction about Electronic Identification cards.
- Introduction about the Portuguese Citizen Card.
- The current available methods of authentication using the Portuguese Citizen Card.
- The contemporary landscape of Near-field communication.
- The role and importance of public key cryptography as an authentication mechanism.
- The current Portuguese Public Key Infrastructure.
- Importance of Transport Layer Security in data transmission.

This strategy guarantees a thorough examination of the important factors that influence this research.

2.1 E-ID cards

Electronic Identification (E-ID) cards are the next generation of traditional identification methods; they are embedded with an electronic chip that stores personal data, usually protected by a PIN, which ensures only authorized entities can access it. This digital storage allows for an array of functionalities beyond just identification.

E-ID cards facilitate secure online transactions, enable digital signatures, can be used for electronic voting in some countries [7] and a multitude of other tasks.

Moreover, E-ID cards are creating a path for a unified digital identity. With a single card, individuals can access an array of different kinds of services, be it governmental, banking or any other, without the need for multiple passwords or credentials. This not only simplifies the user experience but also reduces the risks associated with password breaches. Despite these advantages, this also poses security risks that need to be met with a high security standard.

Thus, a variety of countries around the world have recognized the potential of E-ID cards and are in various stages of their implementation. As technological progress continues, the capabilities of E-ID cards are also expected to improve, making them an indispensable tool in the digital age.

In the context of this research, the primary focus will be on the Portuguese Citizen Card (PCC) as a prime example of an E-ID card, highlighting its features, benefits, and potential for further enhancement.

2.2 The Portuguese Citizen Card

Currently the Portuguese Citizen Card (PCC), an E-ID card, is not just a mere identification tool. It represents a smart combination of technology and governance, designed for the people of Portugal and the broader EU. This smartcard has a chip embedded in it, which is pivotal in streamlining the identification process. Whether one is asserting their identity in a physical setting or online, the PCC stands out as a trusted example of identity verification [8]. A visual representation of this smartcard is depicted in Figure 2.1.

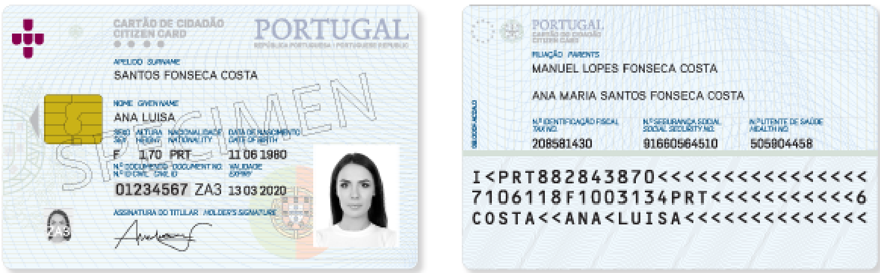


Figure 2.1: Specimen of the Portuguese Citizen Card

However, the Portuguese Citizen Card (PCC) does more than just identify. It acts as a digital key, giving easy access to many services and platforms. Through the portal of "autenticação.gov", it facilitates ingress to Portuguese Public Administration Portals (PPAPs) and a diverse array of other platforms. Whether it's doing civic duties like tax filings, getting personal documents like criminal records, doing bank transactions, or adding a digital signature to an online document, the PCC is essential.

An array of strong reasons support the widespread use and trust of the PCC, such as:

- **Official Recognition**: The PCC is an official document with a seal of approval from respected authorities.
- **Institutional Endorsement**: The importance of the card is increased because it's connected to the INCM (Imprensa Nacional-Casa da Moeda), a famous state institution known for its high standards and reputation.
- **Fortified Security**: In a time when data breaches are common, the PCC stands out with its strong security features. With tools like public key certificates and cryptography, it makes sure data is kept safe and unchanged.

The many functions of the PCC, especially its central role in authentication, are carefully examined in this thesis, highlighting its details and outlining the path for future improvements.

2.3 Current Methods of Authentication Using the Portuguese Citizen Card

In this section, the current methods of authentication available using the Portuguese will be explored, detailing their method of usage and areas of improvement.

Nowadays the authentication in Portuguese Public Administration Portals (PPAPs) and others can be done with a smart card reader or with a service called Chave Móvel Digital (CMD).

2.3.1 Online Authentication Using a Smart Card Reader

The first authentication method is the most used and is done using a smart card reader and a browser plugin[1]. We need to connect our PCC to our computer through the smart card reader and introduce our authentication pin when requested[9]. This authentication pin is generated upon the creation or renewal of the PCC and shipped in a letter to a given address[10]; if this letter is lost and the pin forgotten, a duplicate must be requested in person at a physical government service site(like ”Loja do Cidadão”)[11].

The plugin that is needed for the authentication does not work on smartphones, making the aforementioned authentication process impossible on a smartphone. A company called SmartCardOnMobile (SCOM) tried to solve that problem by creating a mobile application that can perform the authentication using a compatible mobile smart card reader, though this authentication can only be used to consult/share the card information and perform a digital signature[12]. They also created a Software Development Kit (SDK) that can be used by other developers to perform a similar PCC authentication to be used in their applications[13].

The information in the PCC that can be consulted from the SCOM application or using the SCOM SDK is both the public information and the protected data that the PCC contains[12].

2.3.2 Online Authentication with Chave Móvel Digital

Chave Móvel Digital (CMD) is the most recent way to authenticate. First, we need to activate the service; this can be done online, mainly using a smart card reader and our PCC, or in person at a physical government service site[14]. In both cases, we need to create a pin and select a contact form[2] that will be used in the authentication process.

Once the service is activated, to authenticate we just need to input the selected contact, the pin and the one-time-pin which will be delivered through the contact[15]. It’s possible to authenticate without a smart card reader using a computer or a smartphone.

2.3.3 Issues With the Current Authentication Methods

Using a smart-card reader to authenticate has some issues. First, it needs a smart card reader device, which needs to be bought, and most people don’t have one.

Another problem worth mentioning is that the SCOM solution for smartphone authentication solves only half of the smartphones problem; it is still impossible to login into PPAPs using a smart card reader in a smartphone even via CMD.

Chave Móvel Digital (CMD) also has some issues. It needs a subscription to a service, which, as said before, needs to be done in person or online with a smart card reader. Another problem is that the authentication can be done without the physical PCC, raising security concerns.

[1] This plugin is called Autenticação.Gov

[2] A contact can be a phone number, email, or others.

As stated above, the current OAPs using the PCC have many issues. Research and exploration have been done in other countries to provide solutions for similar problems that the PCC OAPs face nowadays. In fact, research on Spain's DNIe system investigates the potential of using NFC in E-ID cards for authentication[16]. If this approach was to be implemented, it would provide similar benefits to the implementation in the PCC, such as not needing additional hardware while keeping the need for the physical presence of the E-ID. Research like the one stated above and this thesis provide valuable solutions for implementing and improving E-ID card authentication with NFC.

2.4 Contemporary Landscape of Near-field Communication

In this section, the current contemporary landscape of Near-field communication (NFC) will be discussed providing resources that support its overall adoption.

2.4.1 What is NFC?

Near-field communication (NFC) is a group of standardised wireless protocols that enable one and two way communication between devices that are within a very short distance of each other (the max range is 10cm). This technology is widely implemented nowadays in many devices that are commonly used, like our smartphones, tablets, cards, and a lot more.

An NFC connection always has an initiator and a target. The initiator starts the interaction by sending a request, and the target answers it[17].

NFC devices can have two types, active and passive; Active devices are battery-powered, passive devices are powered through the electromagnetic field created while an active device communicates with them. Another difference worth stating is that a passive device can only be a connection target, never an initiator, whereas an active device can be both[17]. An example of an active device is a smartphone with NFC support, and an example of a passive device is a NFC tag or the new PCC.

2.4.2 Usage of NFC Technology

With all its characteristics, NFC has a wide variety of possible usages. These are some of the most relevant use cases [18].

Setting up a connection NFC can be used to setup wireless connections since it is an easy and fast way to setup a low-speed connection, one example is using a NFC to set up a WiFi connection. In fact, nowadays many peripherals like headphones and speakers use NFC to establish a Bluetooth connection.

Another example was a feature called Android Beam [3] which allowed the exchange of different kinds of data like contacts, photos, videos, files, and others by putting the devices back to back. Android Beam used NFC to establish a Bluetooth connection to carry out the transfer and then to disable that Bluetooth connection once the operation had finished.

[3] Android Beam was deprecated with the introduction of Android 10 in 2019 and replaced with Nearby Share

Pairing with tags Devices with NFC can pair with NFC tags and be programmed to execute certain actions. Therefore, using this technology, many actions in our smartphones can be automated like setting an alarm, make a call, send a message, enter a website, put in silent mode and so on and so forth. Not only this, but tags have also been used in museums, restaurants, shops, and healthcare.

Payments One of the biggest sectors where NFC technology is used is in contactless payments. It is used in contactless payments with smartphones(*"Samsung Pay, Google Pay, and Apple Pay all use your smartphone's NFC chip for contactless payments."*)[19] and with debit/credit cards. NFC chips can also be embedded within products so that they can be bought with just a tap, making the product sell itself. NFC makes payments simple, fast and safe.

Identity proofing and Access control NFC can be used for identity proofing and access control. NFC cards are used in public transportation access control in many cities like Hong Kong, San Francisco, Frankfurt, and London. The introduction of NFC technologies in public transportation brought benefits to both the customers and the business coordinators. The customers had an easier and faster way to pay for the services, leading to more satisfaction, while the business coordinators benefited from the satisfaction of their clients and lower costs(less paper being used).

Some companies use NFC cards to enhance overall security, like authenticating and tracking entries and exits of the office, logging in to computers and other activities. NFC cards had also been used as house and car keys.

Electronic Identification Card(E-ICs) and E-Passports have NFC technology that contains biometric information about the owner of the passport that is used to authenticate his identity.

A study conducted in 2022[20] asked about the situations where the respondents had used NFC contactless technology, the results were the following:

- 95% for paying for a product
- 48% for paying for public transport
- 38% for consumer product interaction
- 15% for vehicle access
- 14% for venue/stadium/event access
- 13% for home or office access
- 11% for appliance control
- 10% for headphone/speaker pairing
- 8% to get information about a menu or poster or place an order

According to this study, using NFC for payments is the use case more common for NFC.

2.4.2.1 NFC usage growth

The usage of NFC is increasing exponentially. Histogram 2.2 shows the number of devices with NFC support sold per year[21].

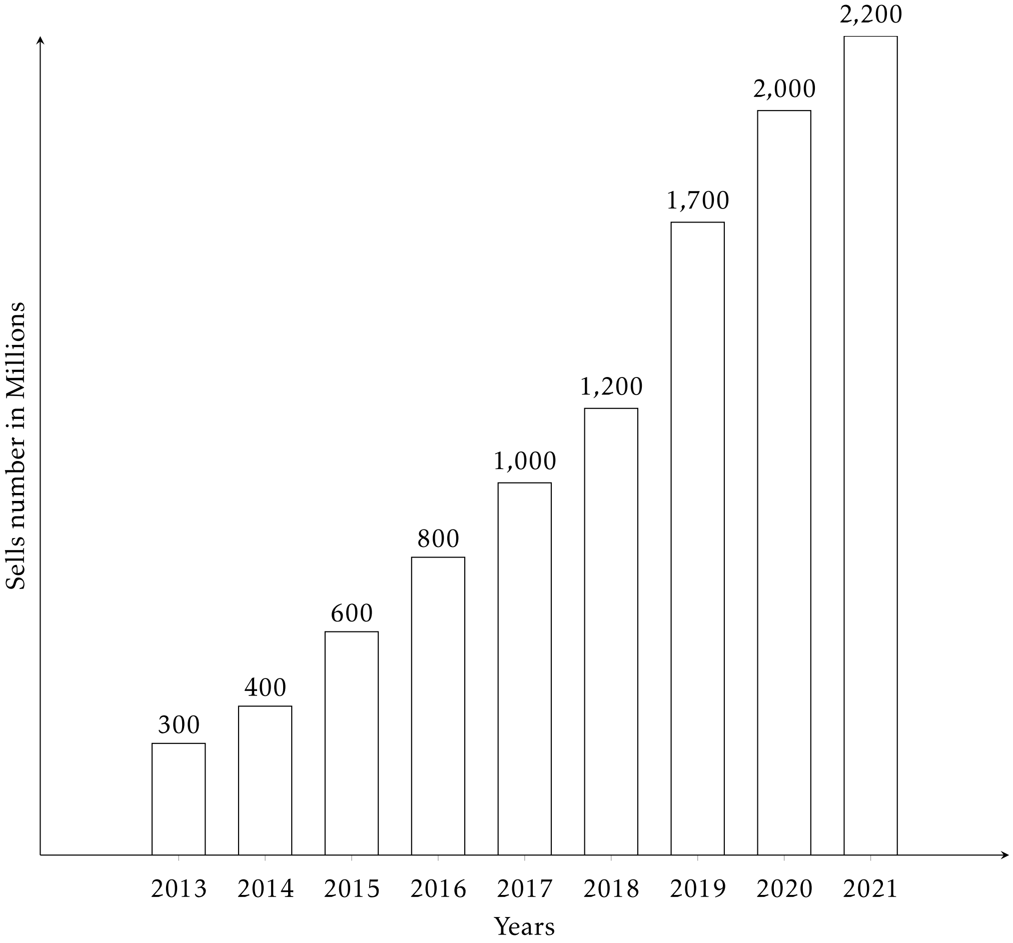


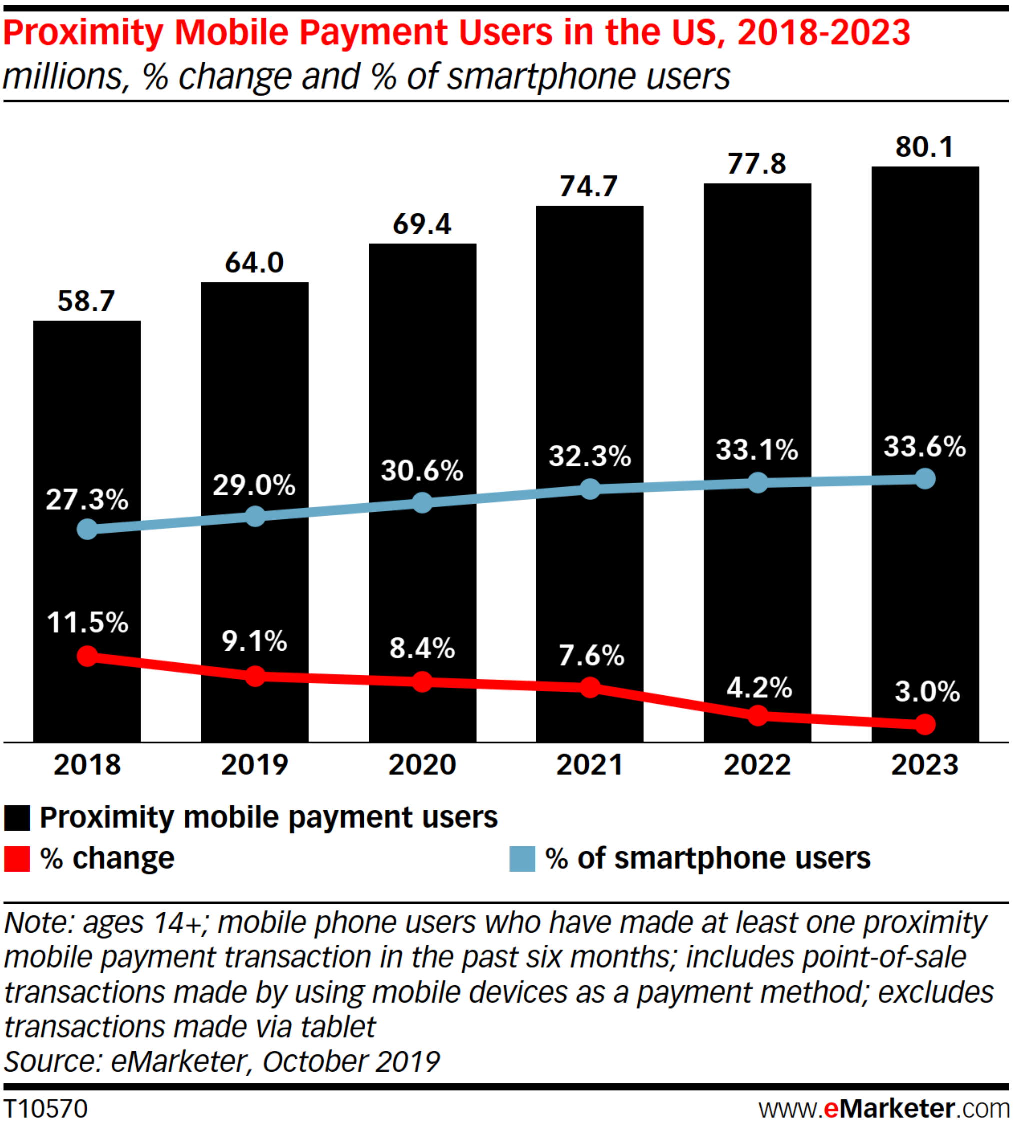
Figure 2.2: Number of devices with NFC support sold per year.

As it can seen the number of devices being sold with NFC support is growing every year. NFC support is now a standard in the mobile phone industry, most of the people are now familiar and use NFC related services daily.

> *"According to IHS Technology, 2.2 billion NFC-enabled smartphone units will be in use by 2020"*[21]

In addition to that, an eMarketer report[22] by Yoram Wurmser[4] about US Mobile Payment Users in 2019 showed that the number of proximity payment users(contactless payment) in the USA is growing and is projected to continue growing over the years. Figure 2.3 shows the growth in adoption of proximity mobile payment methods.

[4] https://www.insiderintelligence.com/analysts/yoram-wurmser


Proximity Mobile Payment Users in the US, 2018-2023

millions, % change and % of smartphone users

Note: ages 14+; mobile phone users who have made at least one proximity mobile payment transaction in the past six months; includes point-of-sale transactions made by using mobile devices as a payment method; excludes transactions made via tablet

Source: eMarketer, October 2019

Figure 2.3: Proximity mobile payment users in the US

Another research[20] that included people of various ages and from various countries also found out that the usage of NFC technologies for payments is growing. Most of the respondents were new users of this technology, having been using NFC for less than 2 years. This showed that, in fact, more people are starting to use NFC. Most of the respondents also believed that NFC is safe. The biggest problems pointed were that NFC can give a slow response, errors and that sometimes multiple attempts are needed.

In this same study, the following information was gathered regarding the familiarity of the respondents with NFC contactless technology:

- 67% of the respondents were familiar or very familiar.
- 32% were more or less familiar.
- 1% were not familiar.

As can be seen most of the respondents were familiar or very familiar making it clear that an NFC based authentication system wouldn't be something that would be hard to adapt to.

In conclusion, overall, the usage of NFC is growing and is here to stay. This trend is further supported by the increasing popularity of NFC in digital identification[23].

2.5 Public Key Cryptography as an Authentication Mechanism

In this section, the way public key cryptography can be used to provide a authentication will be discussed.

Authentication is the act of proving our identity to someone or something.

Challenge-response authentication is a set of protocols where a party that wants to be authenticated is asked a question (also called challenge) by another party that must be answered correctly for the authentication to be successful [24].

Consider, for example, party A wants to authenticate itself to party B; a diagram of their interaction using a challenge-response authentication protocol would look like this:

$$\begin{aligned}
A &\Rightarrow B : \text{Requests Authentication}\\
B &\Rightarrow A : \text{Sends Challenge}\\
A &\Rightarrow B : \text{Answers Challenge}\\
B &\Rightarrow A : \text{Verifies Answer}
\end{aligned}$$

A starts by requesting authentication from B, after which B will send a challenge to A, who will answer the challenge, and if the answer is right then A will be authenticated.

In a public key cryptography system, each party has a set of two keys, one public (every party can access it) and the other private (only the owner knows it, no other party can know it). Challenges are signed with the private key, and the public key is used to verify the signature (verify if the answer is correct).

If party A wishes to authenticate itself to party B then it must first send its ID to B in order for B to obtain A's public key, after which B will send a challenge, which A will sign with its private key and send back to B, and B will verify the signature with A's public key. A diagram using public key cryptography for authentication would look like this:

$$\begin{aligned}
A &\Rightarrow B : \text{Sends ID and B gets A's public key with it}\\
B &\Rightarrow A : \text{Sends challenge}\\
A &\Rightarrow B : \text{Signs challenge with its private key}\\
B &\Rightarrow A : \text{Verifies signature with A's public key}
\end{aligned}$$

All entities need access to the public keys to verify the challenge answers; this can be done using or not using certificates.

2.5.1 Certificateless Public Key Infrastructure

This method is simple and consists of having a single central Database (DB) with a pair of keys and IDs for all parties; therefore, when some entity requests a public key from the central DB, it must trust the integrity of the information in it. Table 2.1 is an example of a central DB.

Table 2.1: An example of how a central DB may look

ID	Public Key
1	"public key of the identity 1"
2	"public key of the identity 2"
3	"public key of the identity 3"
....	....

After a party sends its ID and public key to the central DB it can be authenticated by other parties. In the authentication process, the authenticating party just needs to send to the PKI Server the ID of the party requesting authentication to get its public key to verify its signature. Figure 2.4 shows the interaction between *B* and *A* for the authentication of *A*.

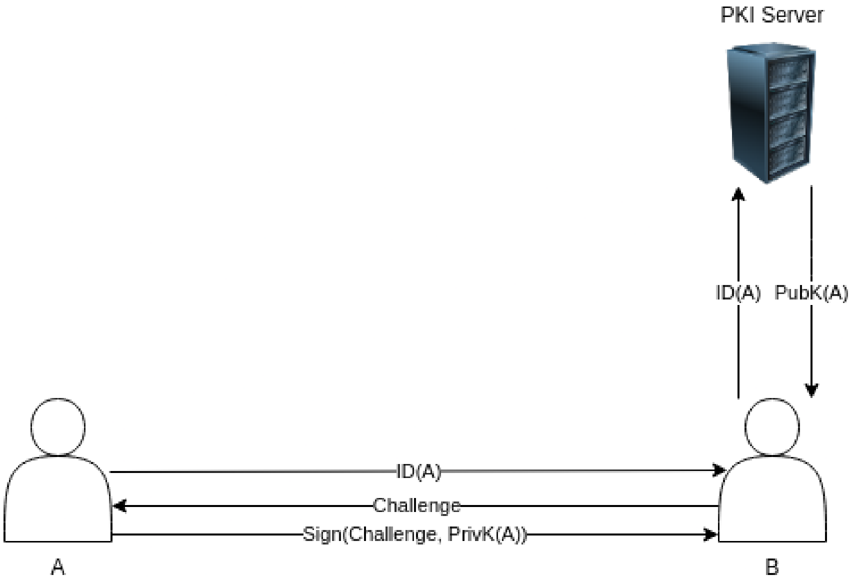


Figure 2.4: Interaction between *B* and *A* for the authentication of *A*

Whenever there is a need to stop an entity from being able to authenticate itself, its entry should be removed from the PKI Server central DB.

2.5.2 Public Key Infrastructure with Certificates

Another common method of authentication is by using certificates. Firstly, there is a need for a Certification Authority (CA) which must to be trusted by every verifying/authenticating party. The CA also has a private-public key pair.

In the registration act, a party sends its ID and public key to the CA, which will sign them with its private key creating a certificate. Then the generated certificate is sent back to the party that is being registered. After this process the party can be authenticated by others. Figure 2.5 shows the interaction between the CA and *A* for the registration of *A*.

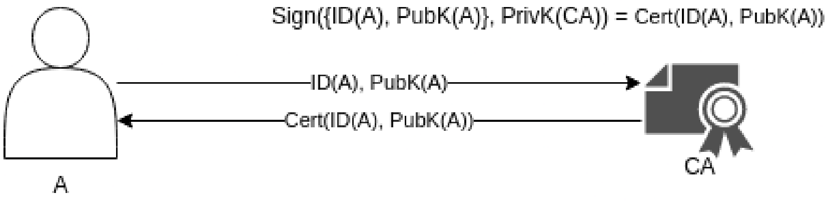


Figure 2.5: Interaction between the CA and *A* for the registration of *A*

In addition to the aforementioned registration process, it is important to note the role of a Certificate Signing Request (CSR) in enhancing security. A CSR is generated by the party on their server and contains their public key and some identifying information, which is then sent to the CA. Unlike merely sending the ID and public key, a CSR includes a digital signature created using the party's private key. This digital signature allows the CA to verify the integrity and authenticity of the CSR, ensuring it has not been changed during transit. The use of a CSR is a safer alternative as it provides a secure method for the party to send its public key and other identifying information to the CA in a way that can be verified and authenticated. This process significantly reduces the risk of tampering or fraud, making the registration act more secure. This added layer of security is crucial in ensuring the robustness of the authentication process as described in the subsequent sections. Figure 2.6 shows the interaction between the CA and *A* for the registration of *A* with a CSR.

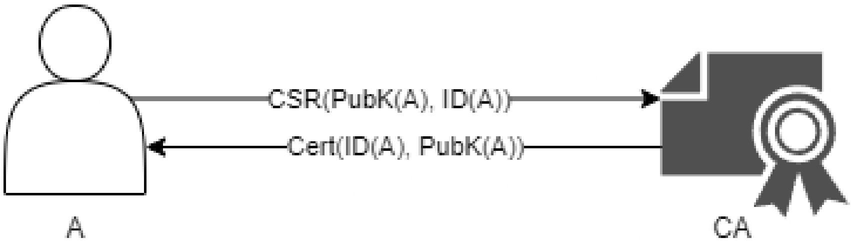


Figure 2.6: Interaction between the CA and *A* for the registration of *A* with a CSR

During the authentication process, the authenticating party will receive the certificate of the party that wants to be authenticated and verify its validity. The authenticating party uses the public key of the CA to verify the signature on the certificate to ensure that it is genuine and has not been changed. If the certificate is valid, the authenticating party will send a challenge to the party that wants to be authenticated. The party that wants to be authenticated will sign (encrypt) the challenge with its private key and send it back. The authenticating party then verifies the digital signature of the challenge using the public key retrieved from the valid certificate. If the verification is successful, the party is authenticated. Figure 2.8 shows the interaction between *B* and *A* for the authentication of *A*.

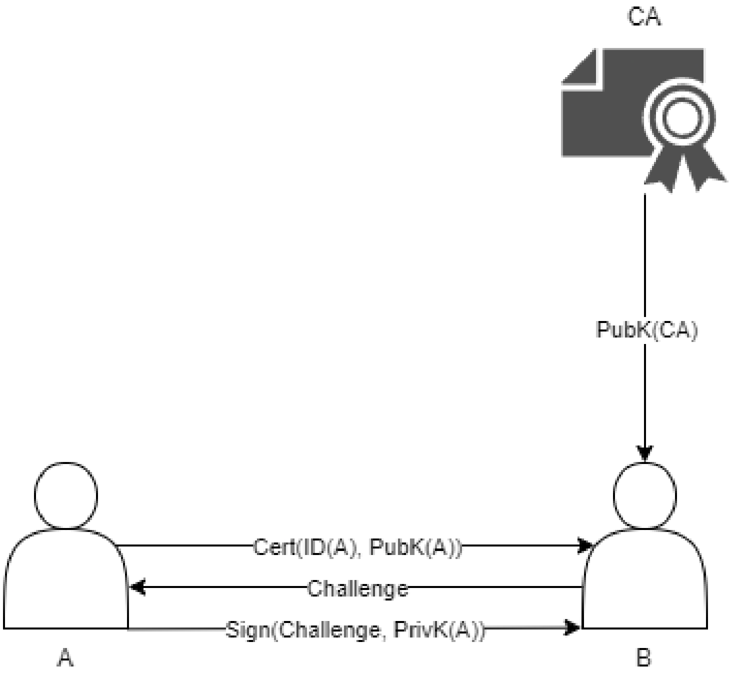


Figure 2.7: Interaction between *B* and *A* for the authentication of *A*

When using this method in order to verify the certificate's validity, it is common to have a DB where all revoked certificates are listed; Whenever a certificate is revoked it is added to that DB so that its revoking status can be verified in the authentication process. It is also possible to have something called an Online Certificate Status Protocol (OCSP) responder, which is a protocol used to check the validity of a certificate through a POST request [25].

In conclusion, a certificate-based PKI system holds a significant edge over a certificateless PKI system. The primary advantage lies in the robust authentication and verification mechanisms facilitated by certificates issued by a trusted CA. Certificates serve as a reliable means of associating a public key with the identity of the entity it represents, thereby establishing a chain of trust. This trust framework is essential for secure communications in most of the digital interactions today. Moreover, the use of certificates enables the implementation of additional security measures such as certificate revocation lists (CRLs) and OCSP for real-time verification of certificate validity. These features contribute to the overall integrity and trustworthiness of the system, making certificate-based PKI a superior choice for ensuring digital security and trust in an increasingly interconnected world.

2.5.3 Certificates Structure: X.509 Standard

A widely used standard for the structure of public-key certificates is X.509 [26]. The public key of the certificate holder and some identifying data are both included in X.509 certificates. These are the main elements from X.509 certificate's structure:

- **Serial Number**: A unique number assigned by the CA to each certificate in the issuance process.
- **Issuer**: The Distinguished Name (DN) of the issuer.

- **Validity**: The period during which the certificate is valid.
- **Subject**: The DN of the owner of the certificate.
- **Subject Public Key Info**: The public key of the subject along with algorithm information.

2.5.3.1 Distinguished Name Fields

An essential component of a X.509 certificate is the Distinguished Name (DN), which identifies the certificate's subject and issuer. A DN is made up of a number of fields, each of which represents a different element of the entity's information. Typical fields are:

- **CN (Common Name)**: The fully qualified domain name (FQDN) of the entity, e.g., "iscte-iul.pt"for a website.
- **O (Organization)**: The name of the organization to which the entity belongs.
- **OU (Organizational Unit)**: The department or division within the organization.
- **L (Locality)**: The city or locality.
- **ST (State or Province)**: The state or province.
- **C (Country)**: The two-letter country code, e.g., "PT"for Portugal.
- **E (Email Address)**: The email address of the entity.

Each field in the DN is represented as a pair of an attribute type and a value, and the complete DN is a comma-separated list of these pairs. For example, a DN might look like "CN=iscte-iul.pt, O=ISCTE - Instituto Universitário de Lisboa, C=PT".

2.6 Portuguese Public Key Infrastructure

In this section, the way the PCC authentication works regarding its PKI will be discussed. The Portuguese PKI uses certificates to help the authentication process.

Each PCC has 2 certificates [27]:

1. A certificate to perform digital signatures.
2. A certificate for authentication (identification of the cardholder).

Another certificate, that is used to authenticate via CMD, exists but is not present in the physical PCC, since this authentication method doesn't require the physical card.

Each certificate type has its own CA:

1. The Digital Signature CA issues the certificates used to perform digital signatures.
2. The CC authentication CA issues the certificates used to authenticate with the CC.

3. The CMD CA issues the certificates needed to authenticate via CMD.

Figure 2.8 shows the certification authority tree.

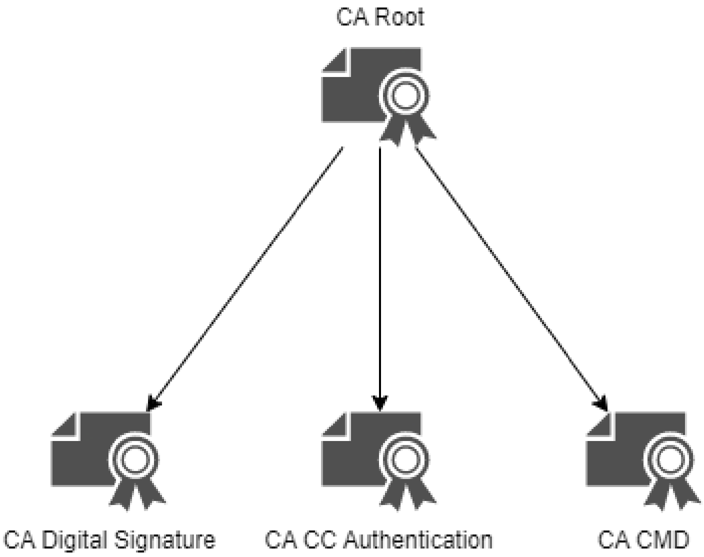


Figure 2.8: Certification authority tree

The Portuguese Public Key Infrastructure (PKI) consists of a hierarchical trust model, where the topmost CA is the CA Root. The CA Root issues certificates for the remaining subordinate CAs, which in turn issue certificates for the Portuguese Citizen Card (PCC) and the Chave Móvel Digital (CMD).

Each CA issues digital certificates with different expiration dates, with the highest expiration dates being at the top of the chain. For example, certificates issued by the CA Root have a higher expiration date than the ones issued by the CMD CA. This ensures that the digital certificates can be trusted and used for secure electronic transactions.

When a CA issues a new certificate, all subordinate CAs below it in the hierarchy must also issue new certificates. This is because each CA uses its own certificate to issue other certificates, and all certificates in the chain must be valid and trustworthy to ensure secure electronic transactions.

There are various reasons why a CA may issue a new certificate. One reason is to minimize risk, as certificates are often only valid for a limited time period and may need to be replaced if compromised. Another reason is to improve security features, such as using stronger encryption algorithms or changing key management practices. Additionally, some changes in policy or technology may require updating certificates, such as when the validity period of a Citizen Card in Portugal increased from 5 to 10 years. It is crucial for CAs to keep their certificates up-to-date to ensure the security of the PKI and the digital certificates issued to users. Valid and properly managed certificates are also important for secure electronic transactions and communications.

Checking the validity of a certificate To ensure the security of digital transactions, it is important to check the validity of certificates. There are two primary methods for checking certificate validity. The first method is called Online Certificate Status Protocol (OCSP), which is a faster and more efficient way of checking certificate status because it queries the CA directly for information on the certificate, rather than requiring the download of a Certificate Revocation List (CRL).

The second method is called a CRL, which is a list that contains all revoked certificates and can be downloaded online. CRLs are typically updated frequently, with a new version available every week. Additionally, there is a type of CRL called a Delta CRL, which is a smaller list that is updated more frequently, often on a daily basis. The Delta CRL only contains the differences (entries added and removed) from the previous CRL, making it more efficient for checking the status of recently revoked certificates.

Both OCSP and CRL are important tools for ensuring the validity of digital certificates and for maintaining the security of the PKI. It is essential for users and applications to regularly check the validity of certificates before accepting them as authentic to prevent fraudulent use of certificates or other security breaches.

2.7 Importance of Transport Layer Security in data transmission

Transport Layer Security (TLS) plays a crucial role in safeguarding information as it moves across networks.

TLS encrypts data to prevent unauthorized access and ensuring that the data remains intact and unaltered during its transmission [28]. This encryption is achieved through a process of cryptographic algorithms where data is transformed into a format that is unreadable to anyone who does not possess the key to decrypt it. In addition to encryption, TLS also provides a mechanism for authenticating the identity of the parties involved in the data exchange. This is typically done using digital certificates, which are issued by trusted CAs.

The primary importance of TLS lies in its ability to secure sensitive data. In scenarios where data integrity, confidentiality and security is a must, TLS ensures that the information exchanged remains confidential and secure from potential attackers. By encrypting the data, TLS prevents the interception of sensitive information like credit card numbers, login credentials, and personal messages. Furthermore, the integrity of the data is maintained, ensuring that the information received is exactly as it was sent, without any unauthorized alterations.

Chapter 3.

Proposed solution

Contents

In accordance the DSRM methodology detailed in section 1.3 this chapter presents the developed solution and demonstrates it. It delves into the details of the developed POC aiming at creating a safer, easier and faster alternative for online authentication. It starts by establishing the high-level requirements each NFC authentication for E-ID cards system must follow to be safe. After it, the solution is dissected and each detail and decision taken is discussed.

[This page has been intentionally left blank]

Chapter 3
Proposed solution

Following the DSRM methodology details in section 1.3 the solution will presented.

3.1 High-level Requirements for a Safe NFC-Based Authentication Using E-ID Cards

This section delineates the essential requirements necessary for the establishment of an authentication system harnessing the combined potential of Electronic Identification (E-ID) cards and Near-field communication (NFC).

The E-ID card Composition The E-ID card must be equipped with NFC tag. Additionally, it must incorporate a secure microprocessor chip responsible for storing the certificate, as well as the private and public keys of the entity utilized in the authentication process. This microprocessor chip must be able to perform the necessary cryptographic operations for the authentication. For added security, the private key must be encrypted when stored within the card, ensuring its protection against unauthorized access.

NFC Initiator The NFC initiator, such as an NFC card reader or a smartphone with NFC support, establishes a secure connection with the authentication server. Once this connection is established, the initiator serves as the primary communication bridge between the E-ID card and the authentication server.

Server Behaviour in authentication The authentication server plays a pivotal role in verifying the authenticity of the E-ID card and ensuring the correct authentication of its owner. Upon receiving an authentication request, the server initiates a challenge, subsequently verifying the response alongside the certificate's authenticity and revocation status. Should the challenge be signed with a legitimate key and the certificate be confirmed as genuine and valid, the server is obliged to invoke multifactor authentication. The mandatory implementation of multifactor authentication significantly bolsters security, ensuring that even in scenarios where the E-ID card is misappropriated or an unauthorized NFC initiator is nearby and possesses knowledge about the local decryption pin, access remains restricted. The authentication server is integrated with a PKI server. This PKI server manages the CAs responsible for issuing and supervising certificates. Furthermore, it upholds the protocols and policies related to the issuance and revocation of digital certificates, ensuring the authentication server can consistently verify the status of these certificates.

Server Behaviour in Registration The registration phase is crucial for establishing a trust relationship between the E-ID card and the authentication server. During registration, it is imperative that the server is obliged to receive a CSR. A CSR is a standardized way of requesting a CA to issue a digital certificate for the public key enclosed in the request, while keeping the corresponding private key secure. The CSR contains vital information including the public key, identity information, and a digital signature, which are essential for the CA to issue a certificate. By mandating the submission of a CSR, the server ensures that the key pair is generated securely, and the private key remains confidential, never leaving the E-ID card or the NFC initiator. This practice significantly enhances the security of the registration process, ensuring that the cryptographic credentials used in the subsequent authentication processes are securely established and managed. Moreover, the use of CSR in registration aligns with best practices in PKI management, ensuring a standardized and secure approach to certificate issuance and management, which is fundamental for the robustness and reliability of the NFC-based authentication system.

User Interface and Experience For an authentication system to be embraced by its users, the interface must be as seamless as the technology behind it. The interface should be designed with the user in mind, ensuring that it's intuitive and straightforward. This entails clear prompts for user actions, feedback on successful or failed authentication attempts, and adherence to modern usability and accessibility standards. Given the sensitive nature of authentication, the interface must also maintain stringent security measures, ensuring that user data remains protected at all times.

3.2 Proposed Solution Design and Architecture

This section introduces a Proof of Concept (POC) for a novel authentication solution designed to validate E-ID card using NFC via a smartphone. This POC converges the benefits of NFC, the strong security features of E-ID cards, and the prevalence of smartphones, offering a new approach to digital identity verification. Figure 3.1 shows the overall system's structure.

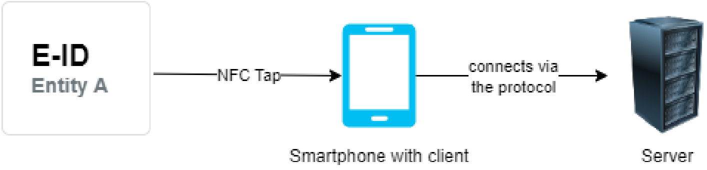


Figure 3.1: Overall system's structure

The core of this POC is a custom authentication server, created using C++, which includes a custom protocol constructed on the sturdy framework of Transport Layer Security (TLS). Renowned for providing secure communication over computer networks, TLS was chosen to ensure the safe transmission of sensitive authentication data.

On the client side, the application is developed in Kotlin, a statically-typed programming language favored for modern Android applications due to its concise syntax, safety features, and seamless interoperability with Java. The application is designed to interact with a custom native library, also written in C++, fostering tight integration and enabling direct access to low-level system APIs for optimized performance and extended functionality such as a custom emulator that emulates the behaviour of a real smart card for testing purposes.

To guarantee robust cryptographic operations, the OpenSSL C library, a comprehensive, open-source toolkit that implements the Secure Sockets Layer (SSL) and TLS protocols, is utilized. Incorporating OpenSSL affords this POC with state-of-the-art security features.

This section outlines the architecture of this POC, explores the details of the custom protocol, and provides insight into the design decisions and compromises made during the development process. The goal is to deliver a comprehensive understanding of the solution, its functionality, and its potential contribution to the field of secure digital authentication.

3.2.1 The Protocol

Integral to this POC is the protocol, custom-designed for securing the communication between the client application and the server. This protocol is pivotal in the process of E-ID card validation via a smartphone, facilitating structured and secure data exchange. This protocol adheres to the X.509 standard for the structure of public-key certificates since it is a widely adopted framework therefore ensuring a consistent and interoperable approach across various systems and applications.

The protocol is designed to accommodate three distinct types of requests, each of these requests serves a unique purpose in the authentication process, ensuring the system can conduct the necessary steps towards secure and successful E-ID card authentication.

1. **Certificate request**: This request issues a certificate for the client. It creates a critical piece of information necessary for the subsequent steps in the authentication process, representing the client's identity in digital form.

2. **Certificate validation request**: This request plays the role of verifying the validity of the issued certificate. It checks if the certificate was issued by a trusted Certification Authority (CA) and whether it remains active, i.e., it has not been revoked. This request essentially informs the client about the current status of their certificate. Keeping abreast of the certificate's validity is crucial in maintaining the integrity of the authentication process and avoiding potential misuse.

3. **Authentication request**: this request performs a challenge-based authentication using certificates. This request is the main entry point for the client when it seeks to authenticate. It includes validation checks and provides an interface for the client to authenticate securely.

It's crucial to note that while each request has a specific function, they are not necessarily sequential. The client can directly make an authentication request, as the requisite certificate validation processes are embedded within it. This design choice offers flexibility and ensures a streamlined and efficient authentication process.

The following sub-sections will delve deeper into each of these three request types, their roles in the protocol, and their contribution to the broader context of the authentication process. By comprehending the details of these requests, the intricacies of the protocol design and its robust security measures become evident. It will also explain the rational behind the choice of making a custom protocol.

In the following discussions on the protocol and its associated requests, it's imperative to underline a foundational assumption. All the outlined interactions, requests, and responses are predicated on the establishment of a secure communication channel, facilitated by a prior TLS handshake. This handshake acts as the bedrock of the system's security, ensuring that all data exchanges between client and server remain confidential and protected. The TLS protocol, with its robust encryption and mutual authentication capabilities, sets the stage for the subsequent steps in the authentication process. Its inclusion underscores the author's commitment to not only securing the higher-level protocol interactions but also ensuring that the very foundation of any communication remains secure.

3.2.1.1 Rationale for a Custom Protocol

The decision to create a custom protocol for the authentication system is driven by an array of reasons, aiming to address the distinct requirements and challenges inherent to E-ID card validation via smartphone. The following points explain the rationale behind the choice of developing a custom protocol:

1. **Enhanced Safety**: A custom protocol allows for the incorporation of security measures rigorously tailored to tackle the unique security challenges associated with E-ID card validation and NFC technology. This personalized approach to security ensures a robust defense against potential threats and vulnerabilities.

2. **Optimized Performance**: By engineering a protocol streamlined to handle the specific types of requests involved in the authentication process, the system can achieve optimized performance. This optimization reduces latency and ensures a seamless user experience, making the authentication process fast and efficient.

3. **Precision Tailoring to Needs**: The custom protocol is precisely done to meet the system's needs and requirements, devoid of any unnecessary features or bloat. This lean design not only makes the protocol lighter and faster but also ensures that every aspect of the protocol serves a defined purpose, aligning perfectly with the system's objectives.

4. **Ease of Maintenance**: With a custom protocol, the maintenance process becomes more straightforward. The protocol's operations and data formats are defined explicitly for this system, simplifying the task of identifying, debugging, and resolving issues. Moreover, as the system evolves, this custom protocol and libraries were done so that they can be adapted or expanded with easily, ensuring their longevity and relevance to the system's needs.

3.2.1.2 Certificate Request

In this request, the client sends a Certificate Signing Request (CSR) to the server. The server, acting as a trusted third party, issues a digital certificate after validating the client's request. This digital certificate, serving as a digital passport, is then registered in the server's database, a critical step that ensures traceability and accountability. Subsequently, the server transmits the certificate back to the client, fulfilling its request. This two-way transaction is a fundamental aspect of establishing secure communication within a network, offering a robust framework for entities to validate each other's identities, ensuring the integrity and confidentiality of their interactions. Figure 3.2 shows the request's diagram.

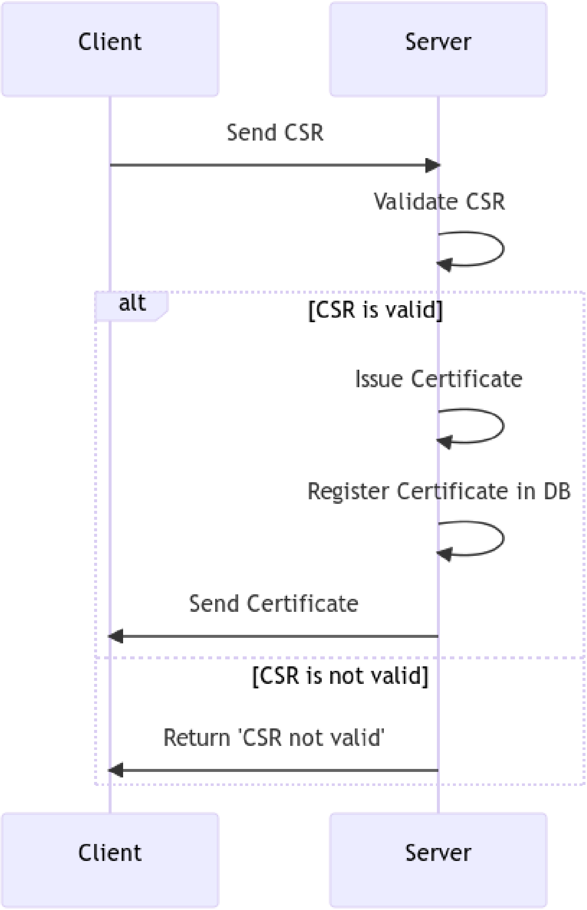


Figure 3.2: Certificate request diagram

3.2.1.3 Certificate Validation Request

In this request, the client sends its digital certificate to a server. The server then embarks on a verification process, a scrutiny of the certificate's origins and status. It checks if the certificate was indeed issued by a trusted CA and assesses the certificate's revoke status. This is a critical step in determining the legitimacy of the certificate and ascertaining whether it has been compromised or revoked. Upon completion of the verification process, the server communicates back to the client, confirming the validity of the certificate or indicating otherwise. Figure 3.3 shows the request's diagram.

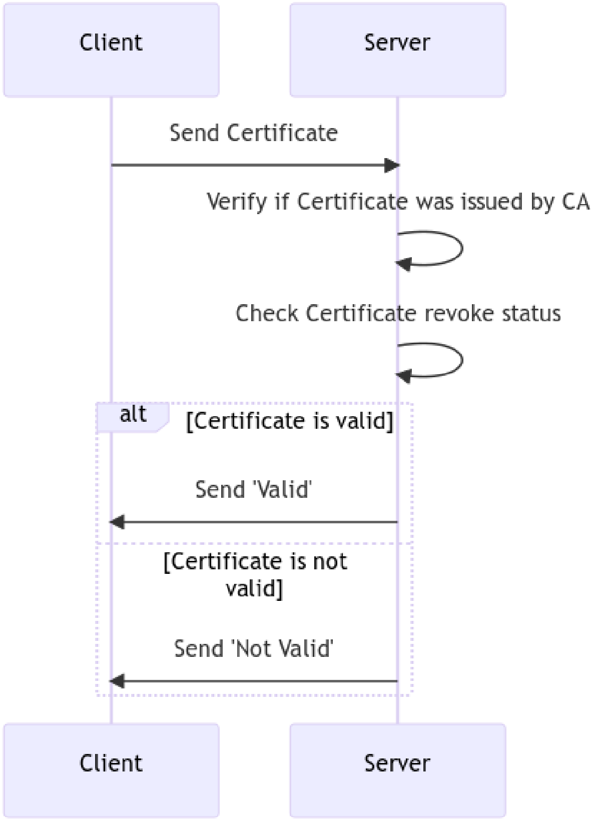

Client
Server
Send Certificate
Verify if Certificate was issued by CA
Check Certificate revoke status
alt
[Certificate is valid]
Send 'Valid'
[Certificate is not valid]
Send 'Not Valid'
Client
Server

Figure 3.3: Certificate validation request diagram

3.2.1.4 Authentication Request

In this request, a client first sends its digital certificate to a server. The server verifies the certificate's legitimacy by confirming whether it was issued by a trusted CA and assessing its revoke status. Should either of these checks fail, the client is deemed unauthenticated, and the process halts. However, if the server verifies these checks successfully, it escalates the process by issuing a challenge to the client. In response, the client signs this challenge, effectively using its private key to encode the server's message. The client then returns the signed challenge to the server. The server, possessing the client's public key from the initial certificate, verifies the signature. If the signature is found to be valid, the client is authenticated, but if the verification fails, the client is not authenticated. Figure 3.4 shows the request's diagram.

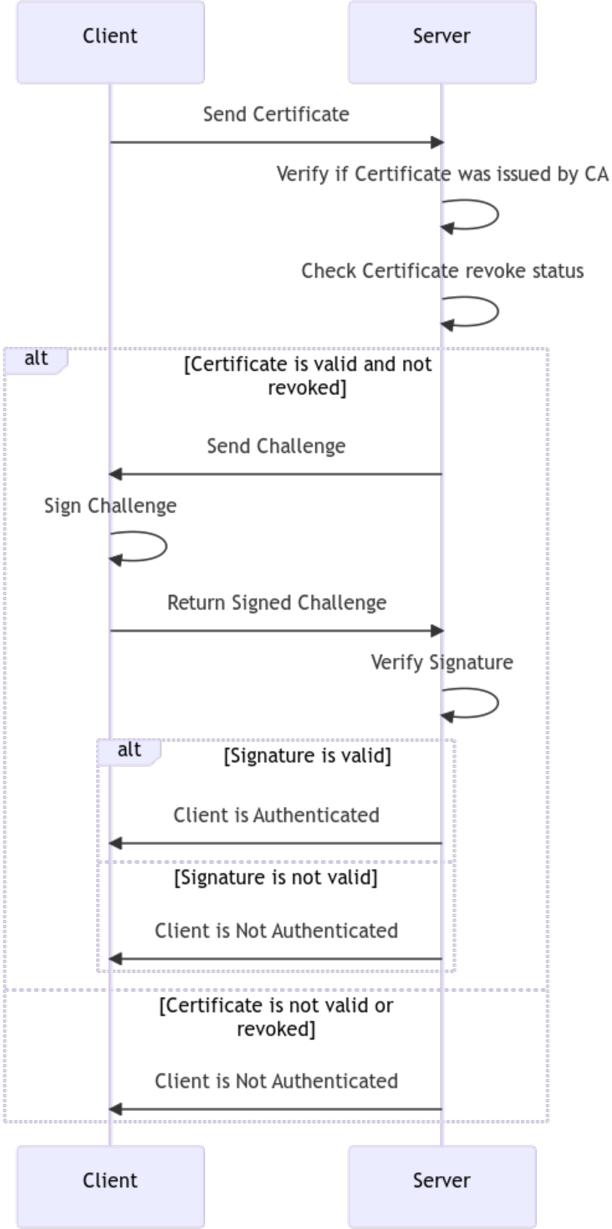

Client
Server
Send Certificate
Verify if Certificate was issued by CA
Check Certificate revoke status
alt
[Certificate is valid and not revoked]
Send Challenge
Sign Challenge
Return Signed Challenge
Verify Signature
alt
[Signature is valid]
Client is Authenticated
[Signature is not valid]
Client is Not Authenticated
[Certificate is not valid or revoked]
Client is Not Authenticated
Client
Server

Figure 3.4: Authentication request diagram

3.2.2 The Server Application

At the heart of the authentication system sits the server built entirely in C++. Tasked with handling and appropriately responding to the various types of protocol requests, the server holds a crucial role in guaranteeing the security and efficiency of the authentication solution.

This sub-section delves into the construction and operation of the server, examining its role in certificate issuance, validation, and challenge-based authentication. It underscores the meticulous design considerations and choices that went into creating a robust and secure authentication server, ensuring the integrity and confidentiality of the authentication process.

The server is designed with multi-threading support, a crucial feature for managing multiple simultaneous client connections efficiently. By handling each client connection in a separate thread, the server ensures responsiveness and swift processing, even under high request loads. Furthermore, for each client request, a distinct socket is established, opting against persistent connections. This design choice enhances security as it reduces the window of opportunity for potential attackers to exploit a session, and ensures that even if one session is compromised, others remain unaffected. This design choice also was made considering the peculiarities and requirements of such authentication system.

Integral to the server's operation is the implementation of the custom protocol previously detailed. This protocol sets the foundational guidelines for communication, ensuring structured and safe data exchanges between the server and client-side application.

Furthermore, the choice to integrate the OpenSSL library was deliberate. OpenSSL, a prominent and widely-utilized software library suite, is one of the most important tools that can be utilized to keep connections safe and assure the integrity of information. Some of the primary reasons for selecting OpenSSL include:

- **Industry Standard**: Recognized worldwide, OpenSSL has been adopted by numerous organizations to ensure communication security.

- **Open Source Nature**: Its open-source status means its code is continually reviewed by the global community, leading to prompt vulnerability identification and mitigation.

- **Comprehensive Cryptographic Tools**: OpenSSL provides a vast array of cryptographic algorithms and tools, allowing for diverse security mechanisms.

- **Active Development and Support**: With regular updates and wide community backing, OpenSSL guarantees that the server adheres to the latest cryptographic standards and is protected against emerging threats.

- **Flexibility and Portability**: OpenSSL is applicable across various platforms, which adds to its adaptability in diverse deployment scenarios. Notably, OpenSSL is also employed in the native library of the client application, ensuring consistent cryptographic standards and interactions between the client and server components.

Furthermore, the principles and needs of secure communication and cryptography also underpinning the decision to adopt OpenSSL are detailed in foundational texts on network security[29].

Delving into the intricacies of the server application, this sub-section elucidates its crucial role within the authentication framework, accentuating its indispensable contributions to the robustness, security, and efficiency of the E-ID card authentication via NFC with a smartphone using this POC. Through a thorough exploration of the server application, this sub-section unveils the crucial mechanisms that bolster the integrity, safety, and streamlined performance of this system overall. Figures 3.5, 3.6 and 3.7 show an UML diagram of the server application, as described in the standard object modeling language by Fowler [30].

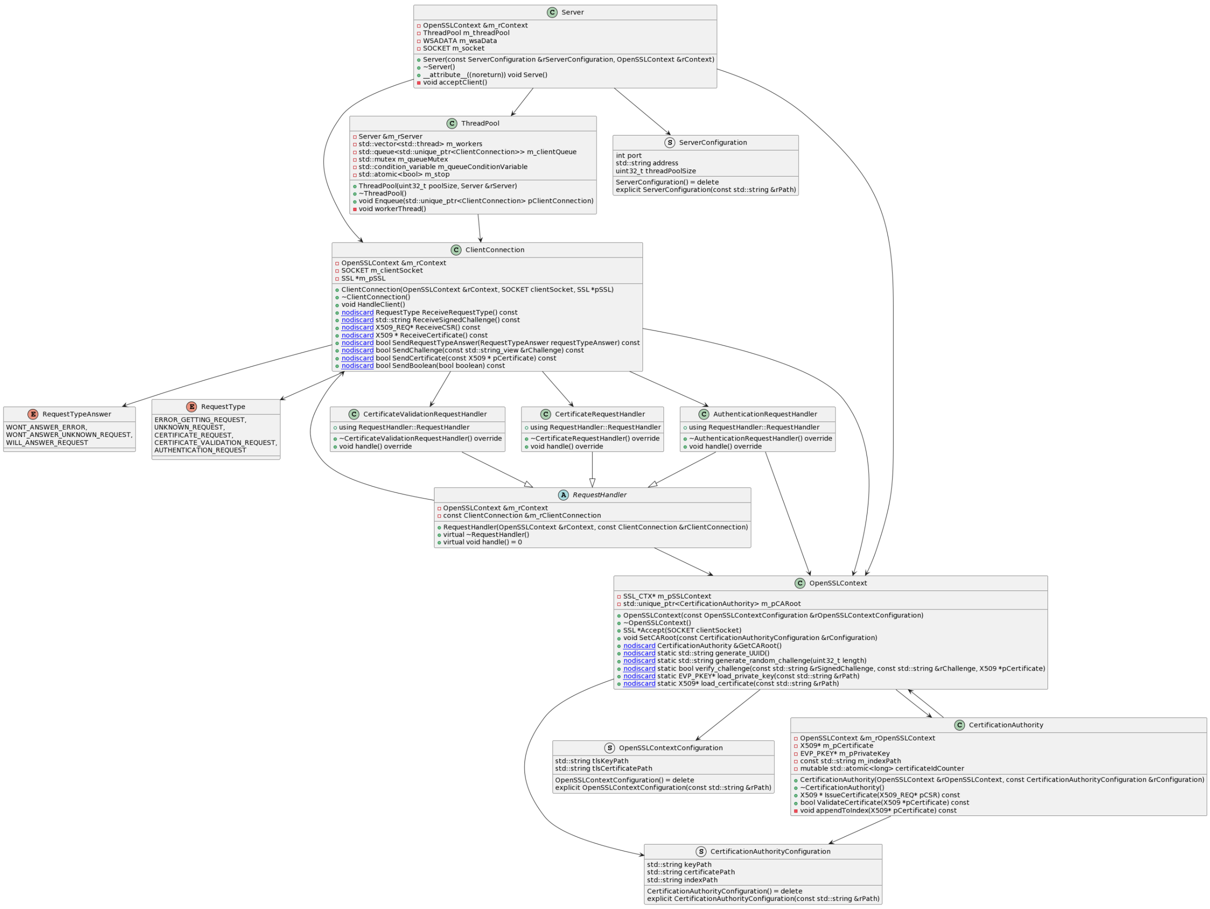

Server
OpenSSLContext &m_rContext
ThreadPool m_threadPool
WSADATA m_wsaData
SOCKET m_socket
Server(const ServerConfiguration &rServerConfiguration, OpenSSLContext &rContext)
~Server()
__attribute__((noreturn)) void Serve()
void acceptClient()
ThreadPool
Server &m_rServer
std::vector<std::thread> m_workers
std::queue<std::unique_ptr<ClientConnection>> m_clientQueue
std::mutex m_queueMutex
std::condition_variable m_queueConditionVariable
std::atomic<bool> m_stop
ThreadPool(uint32_t poolSize, Server &rServer)
~ThreadPool()
void Enqueue(std::unique_ptr<ClientConnection> pClientConnection)
void workerThread()
ServerConfiguration
int port
std::string address
uint32_t threadPoolSize
ServerConfiguration() = delete
explicit ServerConfiguration(const std::string &rPath)
ClientConnection
OpenSSLContext &m_rContext
SOCKET m_clientSocket
SSL *m_pSSL
ClientConnection(OpenSSLContext &rContext, SOCKET clientSocket, SSL *pSSL)
~ClientConnection()
void HandleClient()
nodiscard RequestType ReceiveRequestType() const
nodiscard std::string ReceiveSignedChallenge() const
nodiscard X509_REQ* ReceiveCSR() const
nodiscard X509 * ReceiveCertificate() const
nodiscard bool SendRequestTypeAnswer(RequestTypeAnswer requestTypeAnswer) const
nodiscard bool SendChallenge(const std::string_view &rChallenge) const
nodiscard bool SendCertificate(const X509 * pCertificate) const
nodiscard bool SendBoolean(bool boolean) const
RequestTypeAnswer
WONT_ANSWER_ERROR,
WONT_ANSWER_UNKNOWN_REQUEST,
WILL_ANSWER_REQUEST
RequestType
ERROR_GETTING_REQUEST,
UNKNOWN_REQUEST,
CERTIFICATE_REQUEST,
CERTIFICATE_VALIDATION_REQUEST,
AUTHENTICATION_REQUEST
CertificateValidationRequestHandler
using RequestHandler::RequestHandler
~CertificateValidationRequestHandler() override
void handle() override
CertificateRequestHandler
using RequestHandler::RequestHandler
~CertificateRequestHandler() override
void handle() override
AuthenticationRequestHandler
using RequestHandler::RequestHandler
~AuthenticationRequestHandler() override
void handle() override
RequestHandler
OpenSSLContext &m_rContext
const ClientConnection &m_rClientConnection
RequestHandler(OpenSSLContext &rContext, const ClientConnection &rClientConnection)
virtual ~RequestHandler()
virtual void handle() = 0
OpenSSLContext
SSL_CTX* m_pSSLContext
std::unique_ptr<CertificationAuthority> m_pCARoot
OpenSSLContext(const OpenSSLContextConfiguration &rOpenSSLContextConfiguration)
~OpenSSLContext()
SSL *Accept(SOCKET clientSocket)
void SetCARoot(const CertificationAuthorityConfiguration &rConfiguration)
nodiscard CertificationAuthority &GetCARoot()
nodiscard static std::string generate_UUID()
nodiscard static std::string generate_random_challenge(uint32_t length)
nodiscard static bool verify_challenge(const std::string &rSignedChallenge, const std::string &rChallenge, X509 *pCertificate)
nodiscard static EVP_PKEY* load_private_key(const std::string &rPath)
nodiscard static X509* load_certificate(const std::string &rPath)
OpenSSLContextConfiguration
std::string tlsKeyPath
std::string tlsCertificatePath
OpenSSLContextConfiguration() = delete
explicit OpenSSLContextConfiguration(const std::string &rPath)
CertificationAuthority
OpenSSLContext &m_rOpenSSLContext
X509* m_pCertificate
EVP_PKEY* m_pPrivateKey
const std::string m_indexPath
mutable std::atomic<long> certificateIdCounter
CertificationAuthority(OpenSSLContext &rOpenSSLContext, const CertificationAuthorityConfiguration &rConfiguration)
~CertificationAuthority()
X509 * IssueCertificate(X509_REQ* pCSR) const
bool ValidateCertificate(X509 *pCertificate) const
void appendToIndex(X509* pCertificate) const
CertificationAuthorityConfiguration
std::string keyPath
std::string certificatePath
std::string indexPath
CertificationAuthorityConfiguration() = delete
explicit CertificationAuthorityConfiguration(const std::string &rPath)

Figure 3.5: UML diagram of the server application

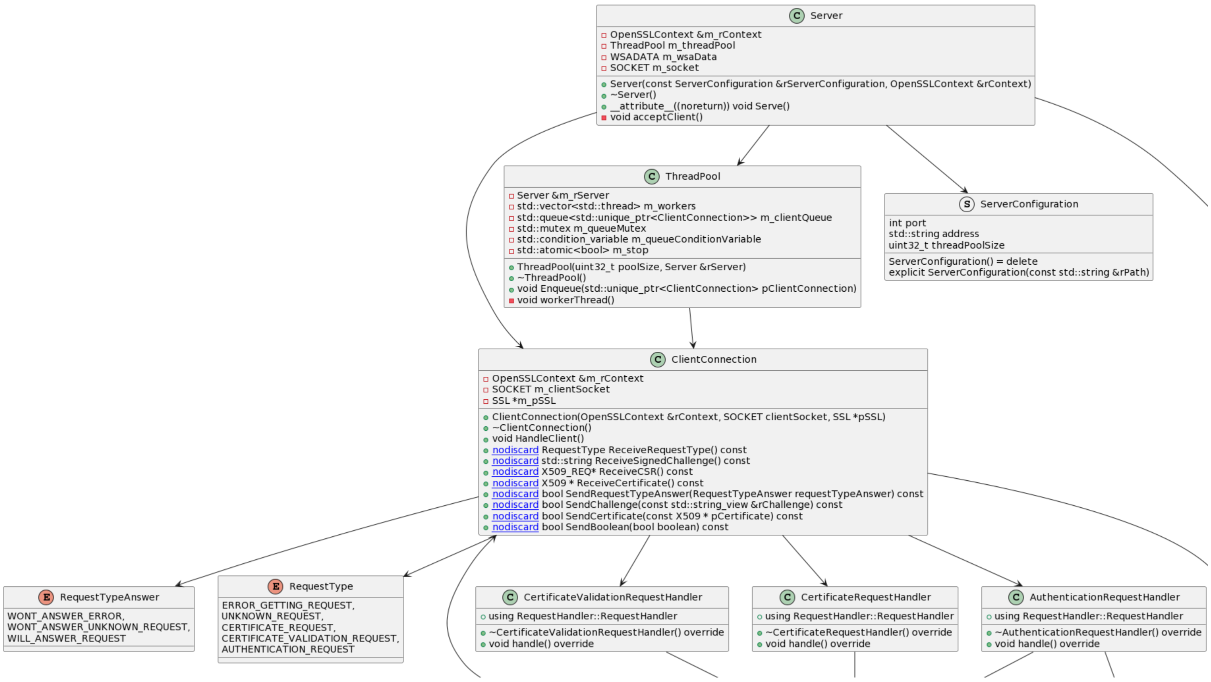

C Server
OpenSSLContext &m_rContext
ThreadPool m_threadPool
WSADATA m_wsaData
SOCKET m_socket
Server(const ServerConfiguration &rServerConfiguration, OpenSSLContext &rContext)
~Server()
__attribute__((noreturn)) void Serve()
void acceptClient()
C ThreadPool
Server &m_rServer
std::vector<std::thread> m_workers
std::queue<std::unique_ptr<ClientConnection>> m_clientQueue
std::mutex m_queueMutex
std::condition_variable m_queueConditionVariable
std::atomic<bool> m_stop
ThreadPool(uint32_t poolSize, Server &rServer)
~ThreadPool()
void Enqueue(std::unique_ptr<ClientConnection> pClientConnection)
void workerThread()
S ServerConfiguration
int port
std::string address
uint32_t threadPoolSize
ServerConfiguration() = delete
explicit ServerConfiguration(const std::string &rPath)
C ClientConnection
OpenSSLContext &m_rContext
SOCKET m_clientSocket
SSL *m_pSSL
ClientConnection(OpenSSLContext &rContext, SOCKET clientSocket, SSL *pSSL)
~ClientConnection()
void HandleClient()
nodiscard RequestType ReceiveRequestType() const
nodiscard std::string ReceiveSignedChallenge() const
nodiscard X509_REQ* ReceiveCSR() const
nodiscard X509 * ReceiveCertificate() const
nodiscard bool SendRequestTypeAnswer(RequestTypeAnswer requestTypeAnswer) const
nodiscard bool SendChallenge(const std::string_view &rChallenge) const
nodiscard bool SendCertificate(const X509 * pCertificate) const
nodiscard bool SendBoolean(bool boolean) const
E RequestTypeAnswer
WONT_ANSWER_ERROR,
WONT_ANSWER_UNKNOWN_REQUEST,
WILL_ANSWER_REQUEST
E RequestType
ERROR_GETTING_REQUEST,
UNKNOWN_REQUEST,
CERTIFICATE_REQUEST,
CERTIFICATE_VALIDATION_REQUEST,
AUTHENTICATION_REQUEST
C CertificateValidationRequestHandler
using RequestHandler::RequestHandler
~CertificateValidationRequestHandler() override
void handle() override
C CertificateRequestHandler
using RequestHandler::RequestHandler
~CertificateRequestHandler() override
void handle() override
C AuthenticationRequestHandler
using RequestHandler::RequestHandler
~AuthenticationRequestHandler() override
void handle() override

Figure 3.6: UML diagram of the server application (Detail A)

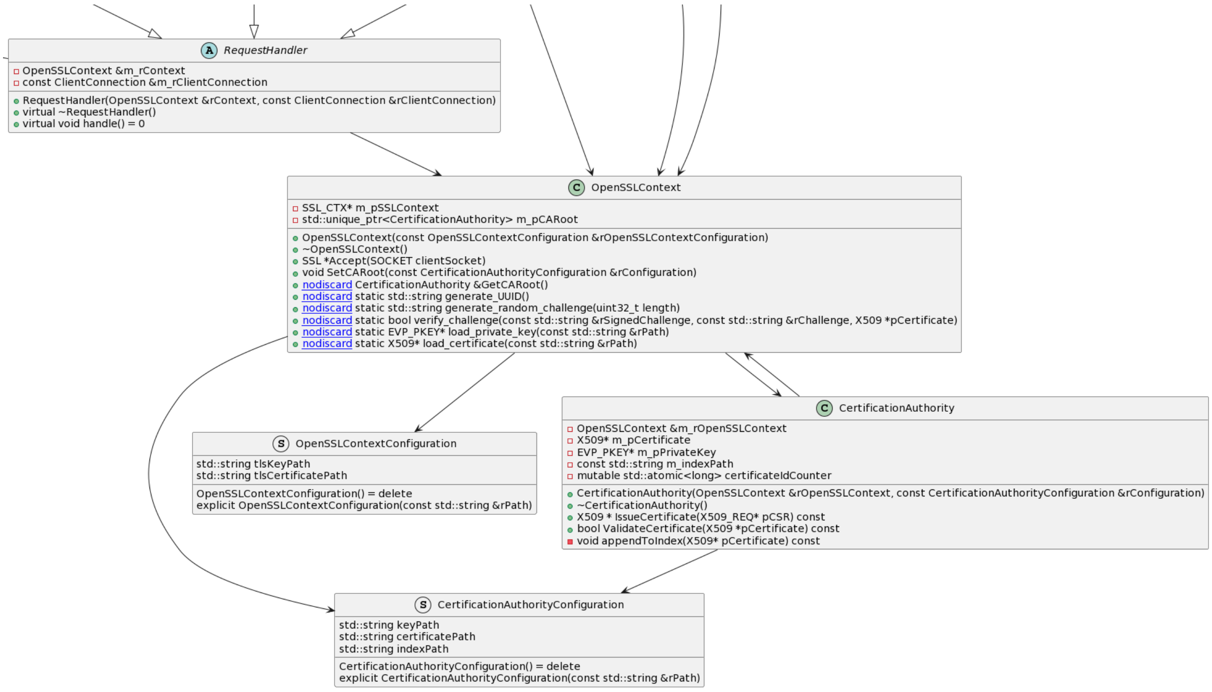

A RequestHandler
OpenSSLContext &m_rContext
const ClientConnection &m_rClientConnection
RequestHandler(OpenSSLContext &rContext, const ClientConnection &rClientConnection)
virtual ~RequestHandler()
virtual void handle() = 0
C OpenSSLContext
SSL_CTX* m_pSSLContext
std::unique_ptr<CertificationAuthority> m_pCARoot
OpenSSLContext(const OpenSSLContextConfiguration &rOpenSSLContextConfiguration)
~OpenSSLContext()
SSL *Accept(SOCKET clientSocket)
void SetCARoot(const CertificationAuthorityConfiguration &rConfiguration)
nodiscard CertificationAuthority &GetCARoot()
nodiscard static std::string generate_UUID()
nodiscard static std::string generate_random_challenge(uint32_t length)
nodiscard static bool verify_challenge(const std::string &rSignedChallenge, const std::string &rChallenge, X509 *pCertificate)
nodiscard static EVP_PKEY* load_private_key(const std::string &rPath)
nodiscard static X509* load_certificate(const std::string &rPath)
S OpenSSLContextConfiguration
std::string tlsKeyPath
std::string tlsCertificatePath
OpenSSLContextConfiguration() = delete
explicit OpenSSLContextConfiguration(const std::string &rPath)
C CertificationAuthority
OpenSSLContext &m_rOpenSSLContext
X509* m_pCertificate
EVP_PKEY* m_pPrivateKey
const std::string m_indexPath
mutable std::atomic<long> certificateIdCounter
CertificationAuthority(OpenSSLContext &rOpenSSLContext, const CertificationAuthorityConfiguration &rConfiguration)
~CertificationAuthority()
X509 * IssueCertificate(X509_REQ* pCSR) const
bool ValidateCertificate(X509 *pCertificate) const
void appendToIndex(X509* pCertificate) const
S CertificationAuthorityConfiguration
std::string keyPath
std::string certificatePath
std::string indexPath
CertificationAuthorityConfiguration() = delete
explicit CertificationAuthorityConfiguration(const std::string &rPath)

Figure 3.7: UML diagram of the server application (Detail B)

3.2.2.1 ThreadPool Class

The ThreadPool class is an integral part of the server application, designed to manage a pool of worker threads that handle client connections efficiently. This design ensures that the server remains responsive even under high request loads, further enhancing the robustness and efficiency of the authentication system. The class is tightly coupled with the Server class, ensuring seamless integration and operation. With its thread management, client connection queue, thread safety mechanisms, and graceful shutdown procedures, the ThreadPool class embodies the meticulous design considerations that went into building a secure and efficient server application.

3.2.2.2 Server Class

The Server class is the central entity responsible for managing client connections and ensuring secure communication. It is tightly integrated with the OpenSSLContext class, which provides cryptographic functionalities, and the ThreadPool class, which manages worker threads for handling client connections. The class is initialized with server configurations, including the IP address and port number, and sets up the necessary socket infrastructure for listening to client requests.

3.2.2.3 OpenSSLContext Class

The OpenSSLContext class is responsible for managing the SSL/TLS context, ensuring secure communication between the server and clients. It integrates with the OpenSSL library to provide cryptographic functionalities, such as SSL handshakes, certificate management, and challenge verification. The class initializes the OpenSSL library, sets up the SSL context, and loads the server's certificate and private key from specified paths. It also ensures proper cleanup of resources during destruction.

Key functionalities and features of the OpenSSLContext class include SSL handshake management, UUID generation for session management, random challenge generation for authentication, and challenge verification using provided certificates. The class also provides methods for loading private keys and certificates, essential for setting up secure communication. By leveraging the OpenSSL library, the OpenSSLContext class ensures that all communications are encrypted and secure, providing a robust foundation for building secure server applications that require SSL/TLS for communication.

3.2.2.4 ClientConnection Class

The ClientConnection class is a pivotal component in the server architecture, dedicated to managing individual client connections and ensuring secure data exchange. It interfaces directly with the OpenSSLContext, facilitating encrypted communication using the OpenSSL library. This class is designed to handle different client requests, such as certificate requests, certificate validation, and authentication requests.

Upon instantiation, the ClientConnection class is initialized with the OpenSSL context, client socket, and SSL structure. It logs the creation of a new client connection and ensures a graceful shutdown during destruction, releasing all associated resources.

The class provides a comprehensive set of methods to handle client interactions:

- **Receiving Data**: Methods like ReceiveRequestType(), ReceiveSignedChallenge(), ReceiveCSR(), and ReceiveCertificate() are designed to securely receive various types of data from the client, such as request types, signed challenges, Certificate Signing Requests (CSRs), and certificates.

- **Sending Data**: The class offers methods like SendRequestTypeAnswer(), SendChallenge(), SendCertificate(), and SendBoolean() to transmit data back to the client. This includes sending responses based on the type of request received, challenges for authentication, certificates, and boolean values.

- **Handling Client Requests**: The HandleClient() method is a central function that determines the type of client request and delegates the handling to the appropriate request handler. It ensures that each request is processed efficiently and securely.

- **Error Handling**: Throughout its operations, the class employs the spdlog library to log any potential issues, ensuring that any communication or data processing failures are immediately identified and logged.

In essence, the ClientConnection class plays a crucial role in managing client interactions, ensuring that each client connection is handled securely and efficiently. By integrating with the OpenSSLContext, it guarantees encrypted and secure communication, making it an indispensable component in the server's secure communication infrastructure.

3.2.2.5 CertificationAuthority Class

The CertificationAuthority class is pivotal for managing certification authority operations within the server, closely integrated with the OpenSSLContext class for cryptographic functionalities. It comprises private members such as a reference to the OpenSSL context, pointers to the certificate and private key of the certification authority, a string for the index file path, and an atomic counter for unique certificate IDs. The class offers methods to issue new certificates based on Certificate Signing Requests (CSRs) and validate existing certificates. The constructor initializes the OpenSSL context, loads the certification authority's certificate and private key, and sets the index path. The destructor ensures proper resource cleanup. Key functionalities include creating new certificates, setting their attributes, signing them with the CA's private key, verifying certificates against the CA's certificate, and logging details of issued certificates. Overall, the CertificationAuthority class ensures secure communication and trust establishment within the server infrastructure.

3.2.2.6 RequestType Enum

The RequestType enumeration defines the types of requests that the server can handle. It consists of five distinct values:

- ERROR_GETTING_REQUEST: Represents an error encountered while fetching the request.
- UNKNOWN_REQUEST: Indicates that the server received an unrecognized request type.
- CERTIFICATE_REQUEST: Denotes a request to obtain a certificate.
- CERTIFICATE_VALIDATION_REQUEST: Represents a request to validate an existing certificate.
- AUTHENTICATION_REQUEST: Signifies a request for authentication.

This enumeration aids in categorizing and processing client requests efficiently within the server framework.

3.2.2.7 RequestTypeAnswer Enum

The RequestTypeAnswer enumeration defines the possible responses the server can give based on the type of request it receives. It comprises three distinct values:

- WONT_ANSWER_ERROR: Indicates that the server won't respond due to an error.
- WONT_ANSWER_UNKNOWN_REQUEST: Signifies that the server won't respond because it received an unrecognized request type.
- WILL_ANSWER_REQUEST: Denotes that the server acknowledges the request and will provide an appropriate response.

This enumeration assists in streamlining the server's response mechanism, ensuring clarity and precision in communication with clients.

3.2.2.8 RequestHandler Class

The RequestHandler class serves as an abstract interface for handling different types of client requests. It establishes a foundational structure that other specific request handler classes can inherit and implement based on their unique requirements. The class encapsulates a reference to the OpenSSLContext and the ClientConnection, ensuring that derived classes have access to essential cryptographic functionalities and client connection details.

Derived classes that implement this interface are expected to provide concrete implementations for the handle() method, ensuring that each type of request is processed appropriately. This interface makes it straightforward to add new request types making the system highly customizable and escalable.

3.2.2.9 Configuration Structures

The server application utilizes structured configuration files to initialize and manage its various components. These configuration files, tin JSON format, ensure that each component is set up with the necessary parameters, promoting modularity and ease of maintenance.

The ServerConfiguration structure encapsulates settings related to the server's operational parameters. The configuration file for this structure might look like:

```
{
  "port": 8888,
  "address": "127.0.0.1",
  "thread_pool_size": 5
}
```

Where:

- `port`: Specifies the port on which the server listens for incoming connections.
- `address`: Defines the IP address of the server.
- `thread_pool_size`: Determines the number of threads in the server's thread pool.

The `OpenSSLContextConfiguration` structure holds settings pertinent to the OpenSSL context. The configuration file for this structure might appear as:

```
{
  "tls_key_path": "tls_private_key.pem",
  "tls_certificate_path": "tls_certificate.pem"
}
```

Key elements include:

- `tls_key_path`: Path to the server's private key used for TLS communication.
- `tls_certificate_path`: Path to the server's certificate used for TLS communication.

The `CertificationAuthorityConfiguration` structure encompasses settings related to the Certification Authority (CA). The configuration file for this structure might be:

```
{
  "key_path": "ca_private_key.pem",
  "certificate_path": "ca_certificate.pem",
  "index_path": "index.txt"
}
```

Important parameters are:

- `key_path`: Path to the private key of the CA.
- `certificate_path`: Path to the certificate of the CA.
- `index_path`: Path to the index file maintained by the CA, which keeps track of issued certificates.

3.2.2.10 Index File Structure

Each CA maintains an index file, named `index.txt`, which plays a critial role in the revocation status checking process. The `index.txt` file is structured in a tabular format where each row represents a certificate entry, and each entry is organized into several fields separated by tabs. Below is a breakdown of the structure of the `index.txt` file:

```
V   Sep  6 16:34:22 2024 GMT     01  unknown /CN=Test
R   Sep  6 16:58:35 2024 GMT     02  unknown /CN=Test1
```

Field Descriptions:

1. **Status Field**: The first field indicates the status of the certificate. The status is either `V` for Valid or `R` for Revoked.

2. **Expiration Date Field**: The second field denotes the expiration date of the certificate, formatted as `Mon DD HH:MM:SS YYYY GMT`.

3. **Serial Number Field**: The third field represents the serial number of the certificate, which is unique to each certificate issued by the CA.

4. **Revocation Reason Field**: The fourth field contains the reason for revocation if the certificate was revoked.

5. **Distinguished Name Field**: The fifth field contains the DN of the certificate owner, which in this example just has the common name filled in.

Revocation Status Checking The `index.txt` file is crucial in the revocation status checking process. When a revocation check is initiated, the system searches the `index.txt` file of the respective CA to determine the status of the certificate in question. If the status field of the certificate entry is marked as `V` (Valid), the certificate is considered valid; if marked as `R` (Revoked), the certificate is considered revoked, and the revocation date field will contain the date of revocation. Various ways of improving this structure and the revocation status checking process are discussed in the Future Work section in Chapter 5.

3.2.3 The Client Application

The client-side application plays a crucial role in this POC. As the point of interaction for users and their E-ID card, the client-side application is responsible for issuing requests to the server and presenting the authentication results in an intuitive and user-friendly manner. Designed in Kotlin and incorporating a custom C++ native library, this application conjugates the flexibility and ease-of-use of a high-level language with the power and control of a low-level language.

In the development of this POC, a key consideration was the practical challenge posed by the unavailability of a physical smart card during certain stages. An emulator was specifically designed to simulate the behaviors of an actual E-ID card. This emulator, meticulously crafted to reproduce the functions, responses, and interactions of a genuine smart card, played a pivotal role in facilitating uninterrupted development and testing. Acting as a surrogate for the real card, it provided a mechanism to validate the system's behavior and interactions in a controlled setting.

With the emulator in place, the focus shifted to the overall architecture of the client application, the handling of various protocol requests, the user interface design, security considerations, performance characteristics, and the testing and validation strategies applied. The aim was to illuminate the strategic and technical decisions made during the development, highlighting its role in the overall authentication process and how it contributes to the robustness and efficiency of the system. The emphasis was on the importance of user-centric design and efficient communication with the server, both of which paramount in the success of any authentication system. The following sub-section provides a comprehensive understanding of the client-side component of this system, elucidating the nuances and intricacies of its implementation.

3.2.3.1 Native android library

At the core of the Android client lies a C++ library. This library, distinct in its construction, follows the Resource Acquisition Is Initialization (RAII) [31] paradigm—a design pattern that binds the life cycle of a resource (memory, network connections, file handles, etc.) to the lifetime of an object. The choice of using the RAII paradigm brings several advantages:

- **Deterministic Resource Management**: With RAII, resources are acquired during object creation and released during object destruction. This deterministic behavior reduces the chance of resource leaks, a prevalent issue in long-running applications.
- **Exception Safety**: The paradigm inherently ensures that resources are correctly cleaned up even when exceptions are thrown, making the system more resilient and stable.
- **Simplified Code**: By automating resource management, RAII reduces manual cleanup code, leading to cleaner, more readable, and less error-prone software.
- **Improved Performance**: Deterministic deallocation means that resources are released as soon as they are no longer required, leading to potentially better resource usage and system performance.

A noteworthy component of this library is the built-in emulator, designed to replicate the behaviors of a physical smart card. As previously discussed, this emulator serves as a critical tool in testing and demonstration stages, given the constraints of physical hardware availability.

Furthermore, the integration of the OpenSSL toolkit stands out. This choice wasn't arbitrary; it mirrors the same cryptographic library used on the server-side application. By employing OpenSSL on both ends, a consistent and secure cryptographic environment is established, reinforcing the trustworthiness of the authentication system. This decision was motivated by OpenSSL's widespread recognition for secure communications and its ability to assure strong cryptographic measures aligning with modern security standards.

Lastly, the library is imbued with the custom protocol designed for the authentication system. This implementation ensures that the Android client seamlessly interacts with the server, fostering a reliable and efficient authentication process unsing an NFC E-ID card with a smartphone.

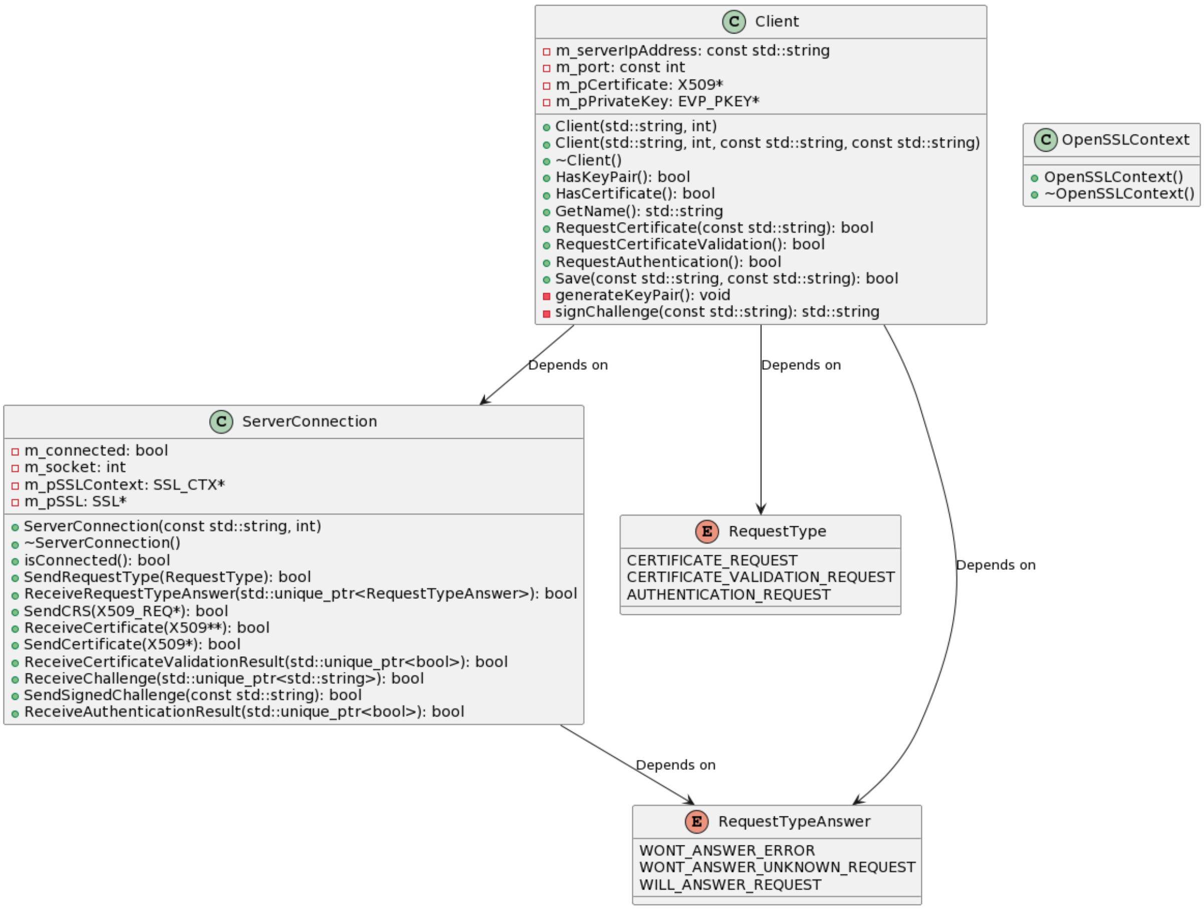

C Client
m_serverIpAddress: const std::string
m_port: const int
m_pCertificate: X509*
m_pPrivateKey: EVP_PKEY*
Client(std::string, int)
Client(std::string, int, const std::string, const std::string)
~Client()
HasKeyPair(): bool
HasCertificate(): bool
GetName(): std::string
RequestCertificate(const std::string): bool
RequestCertificateValidation(): bool
RequestAuthentication(): bool
Save(const std::string, const std::string): bool
generateKeyPair(): void
signChallenge(const std::string): std::string
C OpenSSLContext
OpenSSLContext()
~OpenSSLContext()
Depends on
Depends on
Depends on
C ServerConnection
m_connected: bool
m_socket: int
m_pSSLContext: SSL_CTX*
m_pSSL: SSL*
ServerConnection(const std::string, int)
~ServerConnection()
isConnected(): bool
SendRequestType(RequestType): bool
ReceiveRequestTypeAnswer(std::unique_ptr<RequestTypeAnswer>): bool
SendCRS(X509_REQ*): bool
ReceiveCertificate(X509**): bool
SendCertificate(X509*): bool
ReceiveCertificateValidationResult(std::unique_ptr<bool>): bool
ReceiveChallenge(std::unique_ptr<std::string>): bool
SendSignedChallenge(const std::string): bool
ReceiveAuthenticationResult(std::unique_ptr<bool>): bool
E RequestType
CERTIFICATE_REQUEST
CERTIFICATE_VALIDATION_REQUEST
AUTHENTICATION_REQUEST
Depends on
E RequestTypeAnswer
WONT_ANSWER_ERROR
WONT_ANSWER_UNKNOWN_REQUEST
WILL_ANSWER_REQUEST

Figure 3.8: UML diagram of the native android library

Delving deeper into the library's architecture, shown in figure 3.8, we encounter the 'Client' class, tailored for secure communication with a server. This class encompasses functionalities such as generating key pairs, requesting certificates, validating certificates, and authenticating with the server.Furthermore, it emphasizes proper memory management, evident from the destructor that ensures the release of allocated resources. The class employs the OpenSSL library for cryptographic operations, such as generating RSA key pairs, reading PEM files, and signing challenges.

The 'ServerConnection' class's constructor is responsible for establishing a connection to the server using the provided IP address and port. It initializes the socket, sets up the SSL context, and establishes the SSL connection. The destructor ensures that all resources, including the SSL context, SSL object, and socket, are properly released.

This class plays a crucial role in ensuring secure and efficient communication between the Android client and the server, making it an indispensable component of the authentication system.

Another integral component within the library is the 'OpenSSLContext' class. This class is dedicated to initializing and cleaning up the OpenSSL library, ensuring that all necessary algorithms and error strings are loaded when an instance of the class is created. The class's constructor is responsible for adding all algorithms and loading cryptographic error strings, while the destructor ensures the proper cleanup of error strings and the EVP (Envelope) cleanup.

The inclusion of the ‘OpenSSLContext‘ class underscores the library’s commitment towards providing a robust and secure environment for cryptographic operations, further enhancing the security and reliability of the authentication system.

To further streamline the communication process between the client and the server, the library employs two enumerations: ‘RequestType‘ and ‘RequestTypeAnswer‘.

The ‘RequestType‘ enumeration defines the types of requests that can be made to the server:

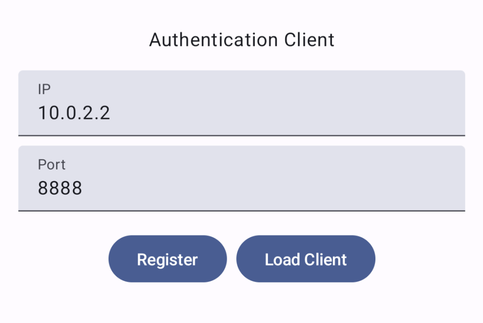
- **CERTIFICATE_REQUEST**: Represents a request for a certificate.
- **CERTIFICATE_VALIDATION_REQUEST**: Represents a request to validate a certificate.
- **AUTHENTICATION_REQUEST**: Represents a request for authentication.

On the other hand, the ‘RequestTypeAnswer‘ enumeration defines the possible answers the server can provide in response to a request:

- **WONT_ANSWER_ERROR**: Indicates an error in answering the request.
- **WONT_ANSWER_UNKNOWN_REQUEST**: Indicates that the server does not recognize the request type.
- **WILL_ANSWER_REQUEST**: Indicates that the server will answer the request.

These enumerations play a pivotal role in ensuring clear and structured communication between the client and the server and its custom protocol, enhancing the efficiency and reliability of the authentication process.

3.2.3.2 User Interface - Client Entry Point



Authentication Client

IP
10.0.2.2

Port
8888

Figure 3.9: Client entry point page

The client entry page, shown in figure 3.9, serves as the initial interface for users upon launching the client-side application. It’s designed with the intent of establishing the initial connection to the authentication server, allowing for subsequent operations like client registration and loading.

The page primarily consists of two input fields:

1. **Server Address**: An input field to specify the IP address or hostname of the authentication server.

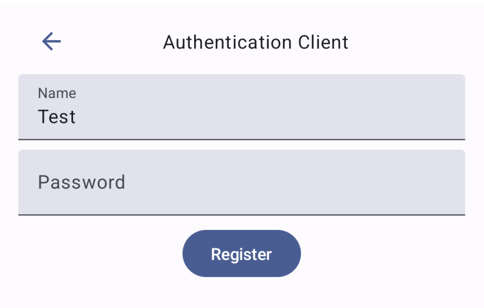
2. **Port**: A dedicated field for users to input the communication port of the authentication server.

Once the server's details are provided, users have two primary actions available:

- **Register Button**: Directs users to the registration page. It's worth noting that this page and button are included predominantly for testing purposes. In a real-world application, the issuance of smart cards, along with their certificates and keys, would negate the need for a client-based registration interface.
- **Load Client Button**: This button leads to a subsequent page where a client's previously registered details are loaded, facilitating the initiation of the authentication process.

The layout and flow of the client entry page highlight the application's focus on simplicity and user-centric design, setting the stage for the in-depth discussion of the registration process in later sections.

3.2.3.3 User Interface - Client Registration



Authentication Client

Name
Test

Figure 3.10: Client registration page

The registration page, shown in figure 3.10, stands as a temporary interface designed primarily for testing purposes in the current version of the client-side application. As such, it's vital to understand its nature and the rationale behind its inclusion.

The page is laid out with simplicity in mind, featuring two primary input fields:

1. **Username**: An input field allowing users to define a unique identifier. This username serves as an identification mechanism for the subsequent communication with the server.
2. **Password**: Directly below the username field, users are required to input a password. This password serves a dual purpose in the testing environment; it aids in encrypting the locally saved certificate and key files and also establishes a local credential for the user.

Beneath the input fields, there is a single **Register Button**. When activated, this button triggers a certificate request embedded with the provided username. This request is then sent to the server for further processing.

However, it is paramount to note some key distinctions:

- In a real-world, production environment, this registration page wouldn't exist. The rationale is that client certificates and keys would typically be preloaded onto smart cards, obviating the need for self-registration.

- Given the emulation environment in this application, the certificate and keys are stored locally. This storage is encrypted using the provided password, ensuring the security of the data even in this test scenario.

- It's emphasized that in a production scenario, the encryption keys and certificates would reside securely within the smart card itself, and any associated password or PIN would be provided directly to the client upon receipt of their smart card.

The existence of this registration page underscores the application's flexibility to cater to both testing and potential real-world scenarios, while also stressing the importance of adopting differing strategies based on the deployment environment.

3.2.3.4 User Interface - Client Load

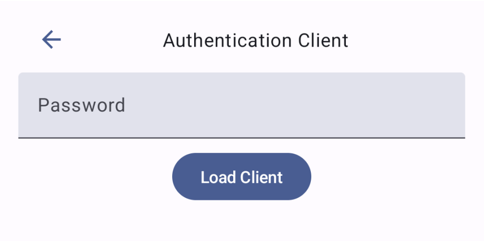


Figure 3.11: Client load page

The client load page, shown in figure 3.11, is a crucial step in the authentication process. It is designed to securely provide access to the client's encrypted certificate and key files, which are pivotal for authentication.

The layout of this page is minimalistic, emphasizing quick and secure access:

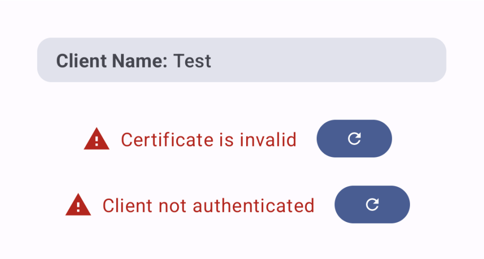
1. **Password Field**: This input field allows users to provide the password required to decrypt their stored certificate and keys.

Upon providing the correct password, users can proceed by selecting:

- **Load Client Button**: This action triggers the decryption process of the locally saved, encrypted certificate and keys. Upon successful decryption, the client is primed for the authentication process.

In a production environment, this stage would typically involve a client scanning their smart card using the smartphone's NFC sensor. This physical action, combined with the password, would be used to access and decrypt the smart card's embedded certificate and keys. However, given the constraints of the current setup and the usage of a custom emulator, the password inputted here is used to decrypt certificate and key files saved locally on the device.

3.2.3.5 User Interface - Client



Client Name: Test

⚠ Certificate is invalid

⚠ Client not authenticated

Figure 3.12: Client page

The Client Page, shown in figure 3.12, serves as the primary dashboard for users once they've logged in. It's the hub from which various authentication-related actions can be initiated and offers users feedback on their authentication status.

1. **Loaded Client Name Display**: This section prominently displays the name of the client that has been loaded, ensuring clarity for the user regarding the active profile.

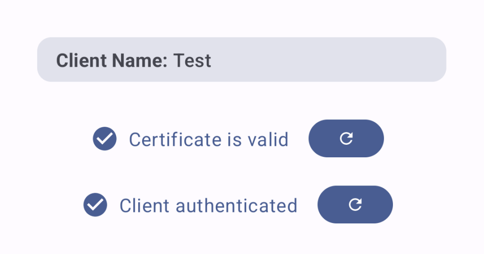
2. **Certificate Validation Button**:

   - Functionality: Initiates a certificate validation request to the server.
   - UI Feedback: The interface updates in real-time, reflecting the server's response regarding the certificate's validity. This direct feedback ensures users are immediately aware of the certificate's status.

3. **Authentication Button**:

   - Functionality: Initiates an authentication request to the server.
   - UI Feedback: The interface updates post-request, indicating the outcome of the authentication process based on the server's response. This helps the user understand whether the authentication was successful or if there were any issues.

The design of the Client Page is both functional and user-centric, focusing on clarity and ease of use. With immediate feedback mechanisms, users are always informed about the status of their authentication and can navigate the authentication process with confidence. Figure 3.13 shows the client page UI after the client is authenticated.



Client Name: Test

✓ Certificate is valid

✓ Client authenticated

Figure 3.13: Client page (client authenticated)

[This page has been intentionally left blank]

Chapter 4.

Results Analysis

Contents

Following the DSRM methodology detailed in section 1.3, this chapter evaluates the results derived from the proposed NFC-based authentication solution using E-ID cards. It revisits the research objectives and questions, providing a thorough analysis of the system's performance in enhancing online authentication. Through a comparative analysis with existing methods, the chapter highlights the advancements of the created POC in terms of security, efficiency, and user experience.

[This page has been intentionally left blank]

Chapter 4
Results Analysis

This chapter delves into the analysis of the results derived from the implementation of the NFC-based contactless authentication system for E-IDs, explained in detail in chapter 3. The analysis is structured around the research questions outlined in the introductory chapter to evaluate the effectiveness, advantages, and benefits of the proposed solution. The proposed research questions in chapter 1 were the following:

[Q.1] How can contactless authentication through an E-ID card using NFC technology be well implemented (safe, easy and fast)?

[Q.2] What are the potential advantages for public administration and private companies in adopting contactless authentication via E-ID cards?

[Q.3] How will citizens benefit from the contactless authentication feature of their E-ID cards?

4.1 Analysis of Research Question 1: Implementation of NFC-based Authentication

This POC was designed with three core principles in mind: safety, ease of use, and speed. The following points elaborate on how these principles were achieved in the implementation:

- **Safety:** The stipulated requirement for the system to incorporate Multi-Factor Authentication (MFA) significantly enhances security [3]. Additionally, the need of physical card presence further bolsters the security, making it a safer alternative to the Chave Móvel Digital (CMD) method which does not necessitate the card's presence. The encryption of the keys and certificate within the E-ID card protect it against unauthorized access [2].

- **Ease of Use:** Unlike systems requiring a smart card reader, this solution simplifies the authentication process by necessitating just a single tap followed by E-ID card decryption and MFA, alongside the installation of a mobile application. This streamlined process enhances user experience by reducing the steps required for authentication.

- **Speed:** Speed was a paramount concern during the development of this system. The single-tap feature, coupled with a swift MFA process, ensures a quick yet secure authentication experience for the users.

4.2 Analysis of Research Question 2: Advantages for Public Administration and Private companies

This POC presents several advantages not only for public administration but also for private companies. The following points elaborate on the benefits:

- **Reduced Workload on Government Employees:** The streamlined authentication process significantly reduces authentication-related queries, thereby reducing the workload on government employees. This leads to improved operational efficiency within public administration, allowing government personnel to focus on other critical tasks.

- **Enhanced Government Service Safety:** The safety features inherent in the NFC-based authentication system, such as the requirement of physical card presence, enhance the security of government services. This added layer of security is crucial in safeguarding sensitive information and ensuring trustworthy interactions between the government and the citizens.

- **Safe Authentication Option for Companies:** Private companies now have a proven safe method to authenticate employees. This POC provides a reliable and secure option for companies looking to improve their authentication processes, thereby enhancing overall organizational security.

This POC, therefore, not only aligns with the security and operational efficiency goals of public administration but also provides private companies with a robust and reliable authentication option. This dual advantage showcases the versatility and potential of the NFC-based authentication system in covering a wide range of authentication needs across different sectors.

4.3 Analysis of Research Question 3: Benefits for Citizens

This POC brings an array of benefits for citizens, enhancing the overall user experience while ensuring a high level of security. The following points elaborate on these benefits:

- **Smartphone Authentication:** The ability to authenticate using a smartphone significantly simplifies the process for citizens. Smartphones are widespread and an integral part of daily life, making them a convenient tool for authentication. This feature eliminates the need for additional hardware or devices, thereby promoting ease of use.

- **Elimination of Smart Card Reader:** Traditional authentication methods often require a smart card reader, which can be difficult and may not always be readily available. The E-ID NFC-based authentication system eradicates the need for a smart card reader, further simplifying the authentication process and reducing the associated costs and inconveniences.

- **Enhanced Safety:** The requirement of physical card presence for authentication, along with the secure protocols employed in the NFC-based system, significantly enhances the safety of the authentication process. This feature is crucial in protecting citizens' sensitive information and ensuring a secure digital environment.

The aforementioned benefits collectively contribute to a more user-friendly, cost-effective, and secure authentication experience for citizens. This POC successfully addresses the common challenges associated with traditional authentication methods, showcasing the potential of NFC technology in promoting a safer and more efficient digital interaction landscape for citizens.

4.4 Conclusion of Results Analysis

The analysis of the research questions provides a comprehensive insight into the many benefits and effectiveness of the developed POC for an NFC-based authentication system using E-ID cards. The core principles of safety, ease of use, and speed were not only well-implemented but also significantly contributed to the enhanced authentication experience for all stakeholders involved - public administration, private companies, and citizens.

This POC, therefore, successfully addresses the outlined research questions, demonstrating a significant step towards modernizing and securing digital authentication processes with E-ID cards. The findings from this analysis not only affirm the viability of the proposed solution but also create a solid foundation for further explorations and enhancements in the domain of contactless authentication technology.

[This page has been intentionally left blank]

Chapter 5.

Conclusion

Contents

This chapter encapsulates the research journey, summarizing the key findings and contributions made towards enhancing online authentication using NFC technology and E-ID cards. The chapter also acknowledges the limitations encountered and suggests avenues for future solution improvements, particularly focusing on expanding the its capabilities.

[This page has been intentionally left blank]

Chapter 5
Conclusion

There are a lot of potential improvements in how people interact with government services thanks to the digital age. This thesis objective was to explore the potential of Near-field communication (NFC) technology in enhancing the authentication process of Electronic Identification (E-ID) cards. Through a thorough examination of the existing authentication mechanisms in the Portuguese context, a novel system architecture, that uses NFC technology, was proposed. The creating of this Proof of Concept (POC) showcased a promising avenue for making online authentication more secure, efficient, and user-friendly.

The evaluation of the proposed system revealed significant improvements in the authentication process. The integration of NFC technology not only streamlined the authentication process but also improved the security measures, thereby addressing the prevalent concerns associated with the existing authentication mechanisms. Furthermore, the user-centric approach used in this research highlighted the importance of creating digital solutions tailored to meet the needs and preferences of the end-users, ensuring a seamless and intuitive user experience.

Contacting with companies in the field showed there is a growing tendency on the adoption of NFC, and therefore, there is a strong potential for developed countries like Portugal to include NFC in their E-ID cards and because of this, there is good change for the Portuguese state to develop a solution aligned with the findings of this thesis in the near future. Thus, it would be desirable for future development to take into consideration the issues raised and solutions proposed in this thesis. The integration of NFC technology with E-ID cards represents a significant step towards a more secure and efficient interaction between citizens and governmental entities. The continuation of this work could play a crucial role in the evolution of digital authentication practices in Portugal, promoting a smooth transition towards more robust and user-centered solutions. Additionally, this could serve as a precursor for other EU states to start similar developments, fostering a harmonized approach towards digital authentication across the European Union.

This work and its consequential findings were published and presented at the Industry Sciences & Computer Sciences Innovation 2023 (ISCSI) conference, further contributing to the discourse on enhancing digital authentication mechanisms and fostering a more secure and user-centric digital ecosystem.

5.1 Future Work

Looking ahead, the successful implementation of this POC lays a solid foundation for further exploration and refinement. Future work could delve into optimizing the system architecture, exploring additional security measures, and conducting extensive user testing to garner more insights into the user experience. Moreover, expanding this research to encompass other forms of digital identification and authentication could unravel new dimensions of enhancing digital governance and online security.

One notable aspect that was not implemented and is in the system requirements of this POC is Multi-Factor Authentication (MFA), which could significantly bolster the security framework of the proposed system. Implementing MFA would provide an additional layer of security, ensuring that users are authenticated through multiple methods before gaining access.

Additionally, the implementation of an Online Certificate Status Protocol (OCSP) responder was not realized in this POC. An OCSP responder is crucial for checking the revocation status of certificates, thereby enhancing the reliability and trustworthiness of the authentication process. Integrating an OCSP responder in future iterations of this project could further improve the system by providing a standardized method for real-time revocation checking, aligning with common practices in digital certificate management.

Furthermore, this POC's custom protocol could be enhanced by introducing new request types, such as a revocation request. A revocation request would allow for the revocation of a certificate through a server request in case it has been compromised or is no longer needed, adding an extra layer of security and control over the authentication process.

Moreover, a significant area for improvement lies in the current method of data storage for issued certificates. This POC utilizes an `index.txt` file to store information about the issued certificates, which is not scalable or efficient for handling a large number of certificates. Transitioning to a more robust and structured data storage solution, such as a database using SQL, would not only improve data management and retrieval but also enhance the system's scalability and performance.

The process of changing the environment of digital authentication is ongoing. The insights garnered from this research contribute to the expanding body of knowledge in the realm of digital authentication and set a precedent for leveraging emerging technologies to create a more secure and user-centric digital ecosystem. The repercussions of this study could extend beyond the realm of citizen card authentication, inspiring innovative solutions that could redefine the way individuals interact with digital platforms and governmental services.

Bibliography

[1] M. Tavares, A. Guerreiro, C. Coutinho, F. Veiga, and A. Campos. "WalliD: Secure your ID in an Ethereum Wallet." In: *2018 International Conference on Intelligent Systems (IS)*. 2018, pp. 714–721. DOI: 10.1109/IS.2018.8710547.

[2] M. M. Mahinderjit Singh, K. Adzman, and R. Hassan. "Near Field Communication (NFC) Technology Security Vulnerabilities and Countermeasures." In: *International Journal of Engineering and Technology* 7 (Dec. 2018), pp. 298–305. DOI: 10.14419/ijet.v7i4.31.23384.

[3] N. Alyousif and S. Alhabis. "The Necessity of Multi Factor Authentication." In: *International Journal of Computer Science and Information Technology Research* 10.2 (Apr. 2022). Research Publish Journals (Publisher), Website: www.researchpublish.com, International Journal of Computer Science and Information Technology Research, ISSN 2348-1196 (print), ISSN 2348-120X (online), pp. 46–49. DOI: 10.5281/zenodo.6472757. URL: https://doi.org/10.5281/zenodo.6472757.

[4] L. A. Meyer, S. Romero, G. Bertoli, T. Burt, A. Weinert, and J. L. Ferres. *How effective is multifactor authentication at deterring cyberattacks?* 2023. arXiv: 2305.00945 [cs.CR].

[5] K. Abhishek, S. Roshan, P. Kumar, and R. Ranjan. "A Comprehensive Study on Multifactor Authentication Schemes." In: *Advances in Computing and Information Technology*. Ed. by N. Meghanathan, D. Nagamalai, and N. Chaki. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 561–568. ISBN: 978-3-642-31552-7.

[6] K. Peffers, T. Tuunanen, C. E. Gengler, M. Rossi, W. Hui, V. Virtanen, and J. Bragge. "Design Science Research Process: A Model for Producing and Presenting Information Systems Research." In: *CoRR* abs/2006.02763 (2020). arXiv: 2006.02763. URL: https://arxiv.org/abs/2006.02763.

[7] J. Budurushi, S. Neumann, and M. Volkamer. "Smart Cards in Electronic Voting - Lessons learned from applications in legally binding elections and approaches proposed in scientific papers." In: Jan. 2012, pp. 257–270. ISBN: 9783885792994.

[8] *O Cartão de Cidadão*. Portuguese. URL: https://www.autenticacao.gov.pt/web/guest/o-cartao-de-cidadao. (accessed: 08/11/2022).

[9] *Autenticação com Cartão de Cidadão*. Portuguese. URL: https://www.autenticacao.gov.pt/cartao-cidadao/autenticacao. (accessed: 06/11/2022).

[10] *Carta e códigos PIN do Cartão de Cidadão*. Portuguese. URL: https://www.autenticacao.gov.pt/web/guest/cartao-cidadao/codigo-pin. (accessed: 06/11/2022).

[11] *Request a duplicate of the pin letter - ePortugal.gov.pt*. English. URL: https://eportugal.gov.pt/en/servicos/pedir-a-segunda-via-da-carta-pin. (accessed: 06/11/2022).

[12] *SmartCardOnMobile – SmartCardOnMobile*. Portuguese. URL: https://scom.pt/. (accessed: 06/11/2022).

[13] *sdk – SmartCardOnMobile*. Portuguese. URL: https://scom.pt/sdk/. (accessed: 06/11/2022).

[14] *Ativar a Chave Móvel Digital*. Portuguese. URL: https://www.autenticacao.gov.pt/web/guest/cmd-pedido-chave. (accessed: 06/11/2022).

[15] *Autenticação com Chave Móvel Digital*. Portuguese. URL: https://www.autenticacao.gov.pt/web/guest/chave-movel-digital/autenticacao. (accessed: 06/11/2022).

[16] J. León-Coca, D. Reina, S. Toral, F. Barrero, and N. Bessis. "Authentication Systems Using ID Cards over NFC Links: The Spanish Experience Using DNIe." In: *Procedia Computer Science* 21 (2013). The 4th International Conference on Emerging Ubiquitous Systems and Pervasive Networks (EUSPN-2013) and the 3rd International Conference on Current and Future Trends of Information and Communication Technologies in Healthcare (ICTH), pp. 91–98. ISSN: 1877-0509. DOI: https://doi.org/10.1016/j.procs.2013.09.014. URL: https://www.sciencedirect.com/science/article/pii/S1877050913008077.

[17] S. Dominikus and M. Aigner. "mCoupons: An Application for Near Field Communication (NFC)." English. In: *21st International Conference on Advanced Information Networking and Applications Workshops (AINAW'07)*. Vol. 2. 2007, pp. 421–428. DOI: 10.1109/AINAW.2007.230.

[18] P. Lathiya and J. Wang. "Near-Field Communications (NFC) for Wireless Power Transfer (WPT): An Overview." In: *Wireless Power Transfer*. Ed. by M. Zellagui. Rijeka: IntechOpen, 2021. Chap. 6. DOI: 10.5772/intechopen.96345. URL: https://doi.org/10.5772/intechopen.96345.

[19] *What is NFC and how does it work? Here's everything you need to know*. English. URL: https://www.androidauthority.com/what-is-nfc-270730. (accessed: 22/10/2022).

[20] A. Research. "Worldwide NFC Technology Use Surges Over Last 24 Months." English. In: (2022). URL: https://nfc-forum.org/news/2022-07-worldwide-nfc-technology-use-surges-over-last-24-months/. (accessed: 26/10/2022).

[21] *The Complete Guide to NFC*. English. URL: https://www.bluebite.com/nfc. (accessed: 24/10/2022).

[22] Y. Wurmser. *US Mobile Payment Users 2019 - Insider Intelligence Trends, Forecasts Statistics*. English. 2019. URL: https://www.insiderintelligence.com/content/us-mobile-payment-users-2019. (accessed: 25/10/2022).

[23] V. Seth. "Why NFC is a rising star in digital ID." In: *Biometric Technology Today* 2021.9 (2021), pp. 5–7. DOI: 10.1016/S0969-4765(21)00094-1.

[24] S. J. Henk C. A. van Tilborg, ed. *Encyclopedia of Cryptography and Security*. English. 2nd ed. Springer New York, NY. ISBN: 978-1-4419-5905-8. DOI: 10.1007/978-1-4419-5906-5.

[25] C. Adams, S. Lloyd, and C. Adams. *Understanding PKI: Concepts, Standards, and Deployment Considerations*. Boston: Addison-Wesley, 2003. URL: https://searchworks.stanford.edu/view/5376191.

[26] *Internacional Telecommuncation Union - X.509*. English. URL: https://www.itu.int/rec/T-REC-X.509. (accessed: 09/03/2023).

[27] *PKI cartão de cidadão*. Portuguese. URL: https://pki.cartaodecidadao.pt/#. (accessed: 06/03/2023).

[28] K. Bhargavan, C. Fournet, M. Kohlweiss, A. Pironti, P.-Y. Strub, and S. Zanella-Béguelin. "Proving the TLS Handshake Secure (As It Is)." English. In: *Advances in Cryptology - CRYPTO 2014*. Lecture Notes in Computer Science. 34rd International Cryptology Conference, Crypto 2014 ; Conference date: 17-08-2014 Through 21-08-2014. Springer, Aug. 2014, pp. 235–255. ISBN: 978-3-662-44380-4. DOI: 10.1007/978-3-662-44381-1_14. URL: https://www.iacr.org/conferences/crypto2014/index.html.

[29] W. Stallings. *Network Security Essentials: Applications and Standards*. 7th. Hoboken, NJ, USA: Pearson Education, Inc., 2017. ISBN: 978-0-13-444428-4.

[30] M. Fowler. *UML Distilled: A Brief Guide to the Standard Object Modeling Language*. Addison-Wesley object technology series. Addison-Wesley, 2004. ISBN: 9780321193681. URL: https://books.google.pt/books?id=nHZslSr1gJAC.

[31] B. Stroustrup. *The C++ Programming Language*. 4th. Boston, MA, USA: Addison-Wesley Professional, 2013. ISBN: 978-0-321-56384-2.

[This page has been intentionally left blank]

iscte
UNIVERSITY
INSTITUTE
OF LISBON

Enhancing E-ID cards authentication with NFC

Tariq Youssef