University of Minho
School of Engineering

Mário Simão Dias Costa

Reducing fraud in authentication systems using attribute certificates

Master's Dissertation

Integrated Master's in Telecommunications and Informatics Engineering

Research oriented by:

Henrique Manuel Dinis dos Santos

and

Sérgio Adriano Fernando Lopes

October 2017

DECLARATION

Name: Mário Simão Dias Costa

Email: simaodcosta@gmail.com

Telephone: +351 918 399 915

Citizen ID Number: 14003283

Title of Dissertation: Reducing fraud in authentication systems using attribute certificates

Supervisors: Professor Doctor Henrique Manuel Dinis dos Santos and Professor Doctor Sérgio Adriano Fernandes Lopes

Year of conclusion: 2017

Master Programme: Masters in Telecommunications and Informatics Engineering

School: University of Minho - School of Engineering

University of Minho, _____/_____/_____

Signature: _____

ACKNOWLEDGEMENTS

I am using this opportunity to express my gratitude to everyone who supported me throughout these years and in particular this thesis period.

First and foremost, I must thank AET Europe in the person of Gonçalo Hermenegildo. Without their assistance and dedicated involvement in every step throughout the process, this thesis would have never been accomplished. I would like to thank you very much for your support and understanding over this period. I would also like to thank DigitalSign for the initial involvement in this thesis and for the partnership on certificates infrastructure.

I would like to express the deepest appreciation to my research supervisors, Professor Henrique Santos and Professor Sérgio Lopes, who helped me to coordinate my project especially in writing this thesis.

I also would like to thank my university friends for the stimulating discussions, for the sleepless nights we have had working together before exams and deadlines, and for all the fun. Thank you, guys, for the strong friendship that will stand forever.

Finally, I must express my very profound gratitude to my parents and to my girlfriend for providing me with unfailing support and continuous encouragement throughout my years of study and through the process of researching and writing this thesis. This accomplishment would not have been possible without them. Thank you.

ABSTRACT

With sophisticate ways to forge authentication systems such as ticketing, there is a growing interest in a way to reduce fraud in these systems. To do so, attribute certificate's technology will be used, more specifically, in a scheme where will be employed QR code to support its transport, to share and to present the attribute certificate for better identification so that a user has a harder task when trying to forge its identity.

The proposed solution will make authentication systems more secure; validations will be available in both online and offline schemes; and since people these days are using and abusing smartphones, and companies need to reduce financial losses due to fraud, our system could easily be delivered with no extra cost for final users.

Keywords: Attribute Certificates; X.509; Android; PHP; Authentication; Identification; Access Control; On-line Authentication; Offline Authentication

RESUMO

Com sofisticadas formas de forjar sistemas de autenticação como a bilhética, existe um enorme interesse para a redução de fraude nestes sistemas. Para isso, a tecnologia de certificados de atributo será usada, e mais especificamente numa forma onde serão empregues códigos QR para suportar o transporte, a partilha e a visualização do AC numa melhor identificação para que o individuo tenha uma tarefa complicada se pretender forjar a sua identidade.

A solução proposta fará com que os sistemas de autenticação se tornem mais seguros; validações estão disponíveis nos modos online e offline; e, assumindo que nestes dias as pessoas usam e abusam dos smartphones, para a redução de custos nas empresas devido à fraude, este sistema pode ser empregue sem custo extra para o utilizador final.

Palavras-chave: Certificados de Atributo; X.509; Android; PHP; Autenticação; Identificação; Controlo de Acessos; Autenticações Online; Autenticações Offline

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS

AA	Attribute Authority
AC	Attribute Certificate
ASN.1	Abstract Syntax Notation One
BER	Bit Error Ratio
CA	Certificate Authority
CRL	Certificate Revocation List
DAO	Data Access Object
DSA	Digital Signature Algorithm
ECDSA	Elliptic Curve Digital Signature Algorithm
GCM	Google Cloud Messaging
ID	Identifier
OTP	One-Time Password
PHP	Personal Home Page
PIN	Personal Identification Number
PKC	Public-Key Certificate
PKI	Public-Key Infrastructures
PMI	Privilege Management Infrastructure
QR Code	Quick Response Code
RFC	Request for Comment
RSA	Rivest–Shamir–Adleman
SDK	Software Development Kit
SHA	Secure Hash Algorithm
S/MIME	Secure Multi-Purpose Internet Mail Extensions
SSL	Secure Sockets Layer
TLS	Transport Layer Security

URL	Uniform Resource Locator
X.509	Standard for Defining Digital Certificates

1. INTRODUCTION

1.1 The Subject Matter

Trust and confidence has always been a part of the human needs. Nowadays, the number of systems that suffer with fraud is enormous. Every year, big sums of money are lost by companies [1] due to fraudulent schemes.

In some services or products, students, disabled or even senior people, can get a discount and pay less than others. Frequently, in services that need to perform a quick ticket issuing, such as for buses/trains and entertainment shows, it is vital to have a fast and an efficient way to prove which ones are eligible to which discounts. What currently we could observe is a high number of fake evidences from people forging those statuses. For example, in Brazil a simple paper is given to the students to prove their rights and it's quite easy to forge due to absence of validation services.

To provide a stronger identification and secure authentication for access control, a combination between Public-Key certificate (PKC) and Attribute Certificate (AC) could be applied. While PKC binds one identity to a public key and proves the related private key ownership through digital signature, an AC contains attributes that specify group membership, role, security clearance, or other authorization information associated with the holder identity. A combination of those two certificates can be used within multiple services, including access control, data origin authentication, and non-repudiation.

Giving this idea, a related work by David Pastor Maestre [2] propose an authentication method which aims to provide an improved secure authentication method using QR codes and PIN code. It works in both online and offline schemes with an authentication method developed using two-factor authentication: a trusted device (a mobile phone) that will read a QR code and that will act as a token, and a password known by the user. In our scenario, instead of a token, it will be presented an attribute certificate to authenticate.

1.2 Motivation and Goals

In a world where fraud is a big problem, it is essential that security schemes can get even more secure. Nowadays, every service has an authentication system by using tickets, tags, QR codes or any other type of digital IDs. With modern and advanced ways to scam, the criminals can bypass some of those methods. In this dissertation, it will be demonstrated how it is possible to reduce fraud in authentication systems using QR code technology and certificates by getting a modern, easy and cheap alternative to authenticate a user. For this purpose, some components must be studied and implemented, for instance: how to use digital and attribute certificate, the process used to read/validate a certificate, how to add an additional factor (something you are or something you have) and the conception of a prototype to authenticate.

This dissertation aims to design and implement an authentication scheme using digital and attribute certificates for either online or offline scenarios. To reach this goal, an Android application will be developed to allow the users to grant access to something (e.g. a ticket or service). For this implementation, the QR code technology will be used.

The main goals of this thesis are:

- Study the different approaches to authenticate via attribute certificates and compare this mechanism with other suitable authentication techniques;
- Develop android application extended to support AC;
- Make adjustments / extensions on an already existing Authentication system;
- Dummy service to request the Authentication system that subsequently invokes the Android App;
- Guarantee the efficiency for the both online and offline solutions.

1.3 Research Methodology

Was followed the subsequent methodology as a guideline in the development of this thesis:

Research on the current state of the art on cryptographic techniques in authentication, X.509 Attribute Certificates structure, standards/publications and existing market solutions.

Definition of an architecture that provides attribute certificate operations to ensure authentication / authorization.

Implementation of the defined architecture that can be used to fulfil one of the main goals.

Deployment of an implemented architecture in real scenarios in order to use as proof of concept usage.

Analyse the developed research, developed work, architecture and protocols done in the scope of this thesis using a Plan–Do–Check–Act cycle.

1.4 Document Structure

The remaining chapters on this document are organised as follows:

Chapter 1 (Introduction) where is presented the framework which clarifies the subjects that will be discussed in this dissertation, followed by the motivation to conduct this thesis and soon after, the goals to be achieved.

Chapter 2 (Cryptographic techniques in authentication) present and discuss the foundations of the research work with an analysis on important topics and technologies that were used.

Chapter 3 (Implementation) describes the main goals on this thesis with architecture and the related implementations.

Chapter 4 (Validations and tests) presents a prototype solution of this thesis with results and analysis for both online and offline features.

Chapter 5 (Conclusion and future work) is presented the conclusion of the developed solution, implementation limitations and future work to be made.

2. CRYPTOGRAPHIC TECHNIQUES IN AUTHENTICATION

This chapter contains the theoretical foundations regarding Authentication Methods, Cryptography, Public Key Cryptography, Public Key Infrastructure and Attribute Certificates. Furthermore, we also review some standards used to manage/issue attribute certificates. We also made an analysis existent market solutions that can be related with this thesis. Finally, we will describe the result of a literature study which is the basis for following chapters.

2.1 Access Control and Authentication

Access control mechanisms are a mandatory and fundamental set of techniques, which is directly related with authentication, authorization and accounting. A secure system should guarantee that only with proper authentication and authorization is possible to access to certain information, whenever we talk about people, devices or processes.

Authentication is a process to present an identity among two parties. Usually, this authentication happens between a client and a server and it is made with credentials, for example: password-based, tickets, tokens, biometric and certificate-based authentication. One differentiator of certificate-based authentication is that unlike some solutions that only work for users, such as biometrics and passwords, the same solution can be used for all endpoints – users, machine, devices and even the growing Internet of Things (IoT). Certificates are natively supported by many applications and networks.

Authorization defines which rights and permissions a user has. After being authenticated, the authorization process determines what user can, or cannot access.

Accounting is the act of collecting information on resource usage for trend analysis, auditing, billing, cost allocation or for security reasons [3].

2.2 Cryptography

Cryptography is a technique that grant integrity and confidentiality of a message from the sender to receiver. To hide original message from third-parties, the plain-text is converted into a cipher-text though an operation known as encryption and later, converted to plain-text again, known as decryption.

Cryptographic systems can assure one or more of following properties and operations [4]:

- Confidentiality – Is the property of ensuring that data is not made available or disclosed to unauthorized people;
- Integrity – Is the protection of information from damage or deliberate manipulation. In plain language, integrity insures that data hasn't been modified;
- Non-repudiation – Is the term used to describe the inability of a person to deny or repudiate the origin of a signature or document, or receipt of a message or document. A person can also prove that it is the author of a valid message.

2.2.1 Symmetric Cryptography

This is considered the simplest form of encryption because only one key is applied in the process. The key is shared between the sender and receiver and it is used to encrypt and decrypt the message. This form of cryptography, raises some problems: guarantee a secure-key storage and to use a proper and safe exchange mechanism with the receiver. It also has the disadvantage of having to generate $n * \frac{n-1}{2}$ keys to n interlocutors [5]. This means that every different communication needs a new shared key. Hence, this creates a new problem with keeping all these keys in a safe storage. As the secret key is shared between two parties, usually we don't apply symmetric cryptography to the authentication. Although, the main advantage is the efficiency and processing speed of encryption and decryption operations [6].

Figure 2.1 represents this symmetric cryptography between sender and receiver.

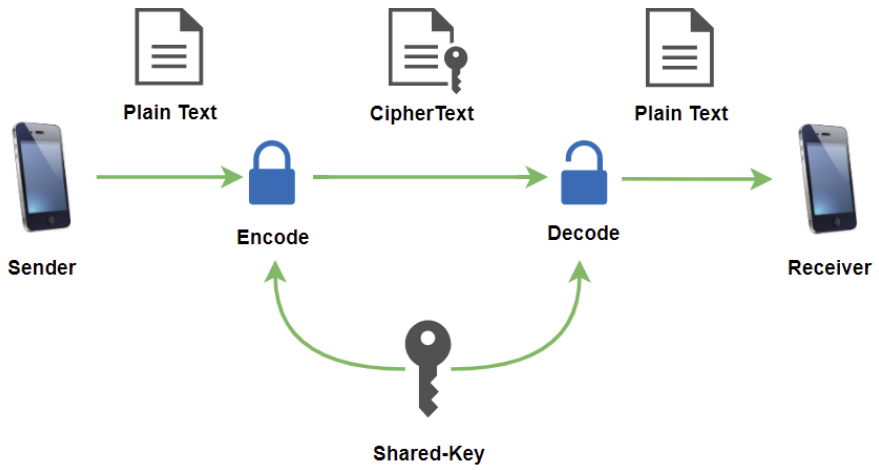

Figure 2.1 – Representation of Symmetric Cryptography

2.2.2 Asymmetric Cryptography

This kind of cryptography, which is also known as Public-key cryptography, works with pairs of keys: public and private keys, that are mathematically related. Everything that is encrypted with one key, it can be decrypted with the other one. For example: if a plain text message is encrypted with the private key, only who possess the associated public key can decrypt it and get access to the plain text. Somehow, if the private-key fell into wrong hands, this cryptography gets threatened [6].

When the message is encrypted with private key, also known as digital signature, we want to ensure the authenticity and non-repudiation. The destination will then be able to decrypt it using the public key, validating the origin and the integrity of the message. When the message is encrypted with public key, the plain text can only be revealed using the private key, ensuring the communication confidentiality. This flow can be seen in Figure 2.2.

Due to requiring higher computer resources, it can be less viable compared to single-key encryption. In spite of that, when compared with symmetric cryptography, it is easier to manage the multiple keys given to the users and only the protection of private key must be ensured.

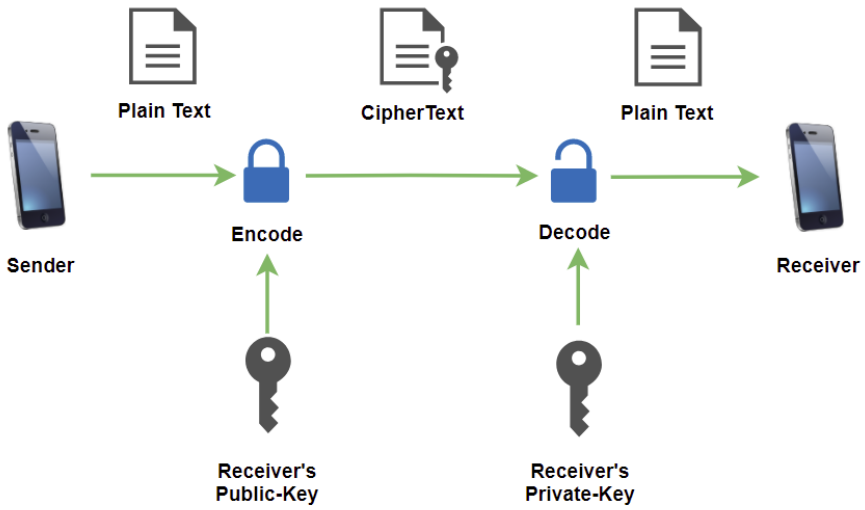

Figure 2.2 – Representation of Asymmetric Cryptography

2.2.3 Hybrid Cryptography

Since both symmetric and asymmetric encryption have different advantages, the combination of this two appears to reduce their disadvantages. SSL/TLS, S/MIME and EFS encryptions are examples of using this hybrid method [7].

First, using symmetric encryption, a secret key, previously generated, is used to encrypt the plain text. That secret key will then be encrypted using asymmetric encryption through the public key. Both encrypted data (original plain text and symmetric key) is then safely sent to the destination. The receiver, through asymmetric cryptography using his private key, will decrypt the symmetric key. The next step is to decrypt the plain text using the symmetric key. This way we get the symmetric encryption power and speed allied to the ease of the asymmetric keys management [8].

The secure exchange of information generally applies a hybrid format of symmetric and asymmetric encryption to preserve confidentiality, integrity and non-repudiation. In Figure 2.3, it is displayed a representative example of this hybrid method.

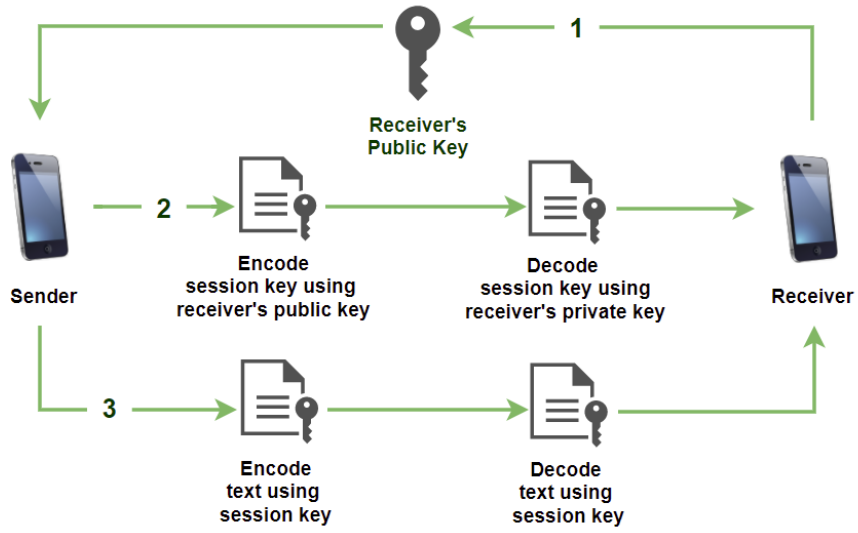

Figure 2.3 – Representation of Hybrid Cryptography

2.3 Digital Signature

Digital signatures have the potential to possess greater legal authority than handwritten signatures because they may provide a higher degree of nonrepudiation and authenticity than their handwritten documents. For example, if a five-page document is signed by hand on the fifth page, one cannot be sure that the first four pages have not been altered. However, if an electronic document is signed with a digital signature, a third party can verify that not one byte of the document has been changed.

The digital signature process is normally held with asymmetric cryptography. The most efficient way to create a valid signature is to use a private key. The signature is unique to a message. On each hashed message, the sender's private key produces different signatures.

Using the corresponding the public key to decrypt the plain text hash, it is made a comparison between the message hash with the deciphered one. If they hashes are the same, the signature is authentic and valid.

There are three key types of algorithms that are used for digital certificates: RSA, DSA and ECDSA [6].

2.4 Public Key Infrastructure

The Public Key Infrastructure (PKI) is the combination of hardware, software, processes, entities, hierarchies and people responsible for establishing trust in the modern electronic world using public key cryptography. PKI is responsible for the creation, distribution, identification, update and revocation of public key certificates and all related activities. PKI relates all communicating parties and certificate validations and verifications.

2.4.1 Digital Certificate and Public Keys

A digital certificate is an electronic document that binds an identity to a public key. A digital certificate, also known as public key certificate or identity certificate, is an identification in digital format, as well as the passport or citizen card are in physical format. Together with the certificate public key, a private key was generated and can be used to perform cryptographic operations. Being a digital credential, this certificate contains important information such as identification data, public key and an identification relative to certificate authority (CA), which is the root of a typical PKI infrastructure. The CA issues and digitally sign the digital certificate and it should be a highly reliable and responsible entity that must ensure the authenticity of the information present in the certificate. As in physical documents, the validity is also an important part in digital certificates. Validity is represented by the issuing and expiry date and are related with the policy for the certificate usage as well as the security assurance given from the key sizes (longer validity requires longer keys).

For a certificate to be useful, it must be clearly structured so later the existing information could be easily collected and perceived. Using passports as an example, it is clearly visible that the structure is very explicit because the information contained in is sensible. In the case of digital certificates, the same strategy applies. Regardless of who issues it, a structure must be followed by applying the one of the existing standards, for example: X.509 or PGP.

Digital certificates can be used in many devices, not only in personal computers. They may also be used in mobile devices and smartcards, such as citizen card. This way digital certificates are easily transportable. They should also be protected with authentication factor (e.g.: password), otherwise, anyone can use it.

2.4.2 Certificate Authority (CA)

A CA is a third party that is responsible to receive the certificate requests. Issues digital certificates with associated public key, validates the requester identity and manage its life cycle. Also sets and defines Certificate Policy (CP) and Certification Practice Statement (CPS) which describes their practice for issuing and managing PKCs.

2.4.3 Registration Authority (RA)

This entity is responsible to identify the certificate requestor and assure the authenticity for each applicant. The registration authority (RA) firstly checks that requests are valid and comply with the CPS and relevant CP. It then authenticates the identity of the user in accordance with any requirements in the CPS and CP. Once satisfied, the RA on forwards the request to the Certification Authority to sign and issue a Digital Certificate to the intended certificate holder. Highly reliable, it also exists to relieve the CA functional load and is closer to the requesting party.

2.4.4 Certificate Revocation List

The Certificate Revocation List (CRL) is a list containing the serial number for all certificates that are revoked. A digital certificate can be revoked at any time for some reasons, for example: in case of lost or stolen private key, malicious use of certificate, security breach on any of the algorithms, or even by mutual ending of contracts. As result of the revocation action, the CRL is published.

The CRL also contains the reason for the revocation, its own validity and the next planned update and it is also digitally signed by the CA.

2.4.5 X.509

X.509 is a ITU-T standard that defines the format of digital/public key certificates [9]. Currently in version 3, the X.509 standardizes public key certificates formats, extensions and attributes

for the certificate. It is inspired in a tree where the elements can be identified by their Distinguished Name (DN). For asymmetric cryptography based applications, the X.509 public key certificate is the most common format. Figure 2.4 shows the PKC used by Google in its main page.

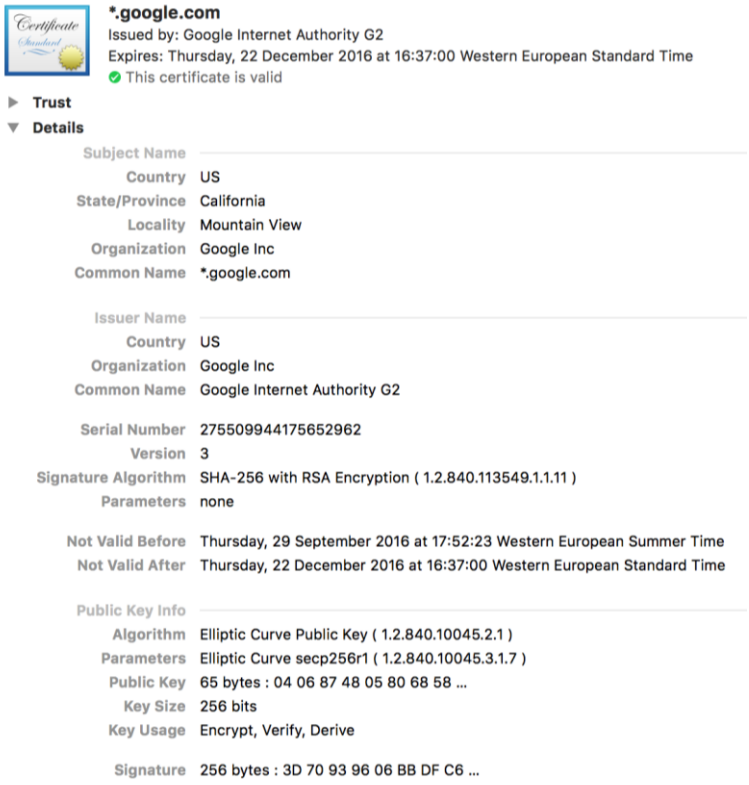

Figure 2.4 – Google's Public Key Certificate

The X.509 standard specifies that a digital certificate must have standardized information. The X.509 version 3 certificates contain the following fields:

- Version – The standard X.509 version being used. Currently it is version 3;
- SerialNumber – This field should be unique for each CA and identifies the digital certificate (on that CA). If subsequently the certificate is revoked, this number can be found in CRL, issued by the CA;
- Signature Algorithm – Specifies the algorithm and hash function used by the CA when signing the certificate (e.g.: SHA-256WithRSAEncryption);
- Issuer – Describes the entity name that issued and signed the certificate. Using this certificate implies trusting the entity that signed it. Note that the CA certificate is auto signed. The name must be unique across the Internet. This field holds the CA

distinguished name (DN). For example: CN=Simão Costa Certificate Authority, OU= Telecommunications Department, O=University of Minho, C=PT (Those acronyms mean: “Common Name”, “Organization Unit”, “Organization” and “Country”, respectively);

- Validity – The date intervals where the certificate is valid: Not valid before and not valid after;
- Subject – Holder's name or entity. Main visible property that represents the holder identity as a distinguished name (e.g. CN=Simão Costa, OU= Telecommunications Department, O=University of Minho, C=PT).
- subjectPublicKeyInfo – Holder’s public key present in this certificate;
- issuerUniqueIdentifier (Optional) – This field is only used to identify the issuer in case of name reuse;
- subjectUniqueIdentifier (Optional) – This field is only used to identify the holder in case of name reuse;
- extensions (Optional) – Complementary field with additional and personalized information. A CA can use extensions to issue a certificate only for a specific purpose (e.g. only to access to a specific web service). The following Figure 2.5, displays the extensions field of Google’s PKC.

Extension	Key Usage (2.5.29.15)
Critical	NO
Usage	Digital Signature
Extension	Basic Constraints (2.5.29.19)
Critical	YES
Certificate Authority	NO
Extension	Extended Key Usage (2.5.29.37)
Critical	NO
Purpose #1	Server Authentication (1.3.6.1.5.5.7.3.1)
Purpose #2	Client Authentication (1.3.6.1.5.5.7.3.2)
Extension	Subject Key Identifier (2.5.29.14)
Critical	NO
Key ID	2D 05 39 D4 9C C9 4F AA 33 32 D2 B1 28 DA 68 4E A0 2E 8E 5B
Extension	Authority Key Identifier (2.5.29.35)
Critical	NO
Key ID	4A DD 06 16 1B BC F6 68 B5 76 F5 81 B6 BB 62 1A BA 5A 81 2F
Extension	Subject Alternative Name (2.5.29.17)
Critical	NO
DNS Name	*.google.com
DNS Name	www.goo.gl
...	...
DNS Name	youtube.com
Extension	Certificate Policies (2.5.29.32)
Critical	NO
Policy ID #1	(1.3.6.1.4.1.11129.2.5.1)
Policy ID #2	(2.23.140.1.2.2)
Extension	CRL Distribution Points (2.5.29.31)
Critical	NO
URI	http://pki.google.com/GIAG2.crl
Extension	Certificate Authority Information Access (1.3.6.1.5.5.7.1.1)
Critical	NO

Figure 2.5 – Extensions Field of Google's PKC

2.5 X.509 Attribute Certificates

A data structure, digitally signed by an Attribute Authority, that binds attribute values to the holder (through the holder's identification information). AC is standardized in X.509. RFC 5755 [10], further specifies the usage for authorization purpose in the Internet.

2.5.1 Digital Certificate *versus* Attribute Certificate

Although the attribute certificates have a structure like public key certificates (PKC), the AC has a different functionality. Attribute certificates are able to support and implement a significant part of the authorization process. However, a major difference is that an attribute certificate does not contain a public key (neither the associated private key). It contains attributes that specify access control information associated with the AC holder (such as group membership, role, security clearance).

An analogy can by use to clarify both certificates:

- The public key certificate (PKC) could be seen as a passport: it identifies the holder, usually with short validity and not easy to obtain – centrally issued;
- An AC is like a cinema ticket: is typically issued by different authorities, not valid for a long period of time, easy to obtain and it has no legal validity.

An AC may be used in various security systems, including access control, authentication and non-repudiation. It is a good approach to separate authorizations from PKC; however, it is fundamental to link the authorization to an identity or entity. Although that AC has its own signature, it also provides this connection linking the PKC to the attribute certificate through the subject/holder distinguished name (keeping the same DN on both certificates).

2.5.2 Attribute Certificate Structure

The attribute certificate is defined as presented in Figure 2.6:

```
AttributeCertificateInfo ::= SEQUENCE {
  version                 AttCertVersion, -- version is v2
  holder                  Holder,
  issuer                  AttCertIssuer,
  signature               AlgorithmIdentifier,
  serialNumber            CertificateSerialNumber,
  attrCertValidityPeriod  AttCertValidityPeriod,
  attributes              SEQUENCE OF Attribute,
  issuerUniqueID          UniqueIdentifier OPTIONAL,
  extensions              Extensions OPTIONAL
}
```

Figure 2.6 – ASN.1 standard for Attribute Certificate

- Version – Specifies the version of the attribute certificate. Currently it must be v2;
- Holder – Describes the AC's holder which can be defined in three different ways;
    - BaseCertificateID – If present, shall identify a public-key certificate that will be used to authenticate the identity of this holder;
    - EntityName – One or more names used to identify the holder;
    - objectDigestInfo – Used to directly authenticate the holder, for example: an executable.
- Issuer – Indicates the entity of the Attribute Authority (AA) that issued the certificate;
- Signature – Describe a cryptographic algorithm used to sign the certificate;
- SerialNumber – Uniquely serial number which identifies the attribute certificate;

- attrCertValidityPeriod – Specifies certificate validity period;
- Attributes – It contains information (usually attributes) about AC's holder. When used for authorization, typically specify their privileges. Organized in OID with associated value.
- IssuerUniqueID (Optional) – May be used to identify the AC's issuer. This field should only be used by the issuer.
- Extensions (Optional) – Additional fields with additional and personalized information.

2.5.3 Attributes

In this field, it is possible to set all permissions to be used in a business context or not. Here are defined the existing roles associated to each holder, represented by object identifiers (OIDs). For example, we can see that a CEO could have full access to all company sectors while an employee only to the tasks which was hired for.

In RFC 5755 [10] it is possible to find the attribute structure shown in Figure 2.7 – ASN.1 structure of Attributes.

```
Attribute ::= SEQUENCE {
  type      AttributeType,
  values    SET OF AttributeValue
    -- at least one value is required
}

AttributeType ::= OBJECT IDENTIFIER

AttributeValue ::= ANY DEFINED BY AttributeType
```

Figure 2.7 – ASN.1 structure of Attributes

The available types of attributes presented in RFC 5755 [10] are:

- Service Authentication Information – The SvceAuthInfo attribute identifies the holder of the attribute certificate to the server or service with its name. Typically, this contain a username and password (for a legacy application). If this field contains sensitive information, encryption should be used;
- Access Identity – The accessIdentity attribute has the function to provide information about AC's holder, that later may be used in authorizations;

- Charging Identity – The chargingIdentity attribute identifies the holder of the attribute certificate for charging purposes. For example, the ChargingIdentity attribute maybe used to identify the holder's company which may be charged for some service;
- Group – The group attribute brings information about group memberships of the holder of the attribute certificate;
- Role – The role attribute shows information about role allocations about attribute certificate's holder;
- Clearance – The clearance attribute, contains security information about the holder. PoliceID field is used to identify a security policy of the information. The basic security classification hierarchy is (in ascending order): *unmarked*, *unclassified*, *restricted*, *confidential*, *secret* and *top-secret*.

2.5.4 Extensions

As previously mentioned, extensions field is optional. However, it allows attribute certificates to provide methods for associating additional attributes with holders.

It is possible to find the following extensions in RFC 5755 [10].

- Audit Identity – For audits purposes, this field is used so that the AC's holder can stay anonymous. The identity may be revealed only by the entity that issued the certificate;
- AC Targeting – It is used to target an AC by specifying many servers/services. It defines a group that records all those who have permission to access attribute certificate content. An (honest) AC verifier who is not amongst the named servers/authorized services, should reject the attribute certificate. If this extension is not present, it means the AC can be accepted by everyone;
- Authority Key Identifier – It should be used to support AC's verifier checking the signature of the AC. If baseCertificateID option was used in holder's field, this extension is not necessary because the link with the digital certificate is already known.
- Authority Information Access – This extension aims to assist the AC's verifier checking its revocation status;
- CRL Distribution Points – If used, assists the AC's verifier checking the AC revocation status;

- No Revocation Available – Allows the issuer of the attribute certificate to indicate that any information will not be available about the revocation of this AC.

2.6 Standards

IETF RFC 5280 – Internet X.509 PKI Certificate and CRL

This Request for Comments (RFC) 5280 [11], profiles the X.509 v3 certificate and X.509 v2 certificate revocation list to use on the Internet. The X.509 v3 certificate format is described in detail, with additional information regarding the format and semantics of Internet name forms. Standard certificate extensions are described and two Internet-specific extensions are defined. A set of required certificate extensions is specified. The X.509 v2 CRL format is described in detail along with standard and Internet-specific extensions. An algorithm for X.509 certification path validation is described.

IETF RFC 5755 – An Internet Attribute Certificate Profile for Authorization

This Request for Comments 5755 [10], specifies the profile to use X.509 Attribute Certificates on Internet Protocols for authorization. The target of this document is to create a standard for generic applications requiring extensive interoperability. It defines a way to "target" an AC at one or a small number of servers. Otherwise, the wide servers will reject the AC. It also defines a mechanism so that ACs can either be "pushed" and "pulled" by the client to a server.

ISO/IEC 9594-8 – Public-key and attribute certificate frameworks

ISO/IEC 9594-8:2014 [9] defines a framework for public-key certificates. This framework includes the specification of data objects used to represent the certificates themselves, as well as revocation notices for issued certificates that should no longer be trusted. The public-key certificate framework defined in ISO/IEC 9594-8:2014, while defining some critical components of a public-key infrastructure (PKI) and Privilege Management Infrastructure (PMI), does not completely define PKI or PMI. However, it provides the foundation upon which full PKIs/PMIs and their specifications would be built.

Information objects for holding PKI and PMI objects in the directory and for comparing presented values with stored values are also defined.

ISO/IEC 9594-8:2014 also describes a framework for the provision of authentication services by the directory to its users.

ISO/IEC 18004 – QR Code

ISO/IEC 18004:2015 [12] defines the requirements for the symbology known as QR Code. It specifies the QR Code symbology characteristics, data character encoding methods, symbol formats, dimensional characteristics, error correction rules, reference decoding algorithm, production quality requirements, and user-selectable application parameters.

2.7 Market Solutions and Publications

Although there is no existing project like the one we are proposing in this thesis, there are some different solutions that can be related as they use the same technologies.

Apple Wallet

Released in 2012, Apple Wallet (previously known as Passbook) was developed to allow users to store coupons, boarding passes, event tickets, and since iOS 8.1, banking cards via Apple Pay. In this scheme, the pass files are stored on disk as zipped with the pkpass file extension. This pass packages contain text, images, barcode, signature and some relevant keys as passTypeIdentifier and a unique serialNumber [13]. For any reason, if this file is shared or stolen, the data privacy stays in risk.

Access Control based on AC for Medical Intranet Applications

This project [14] attempts to contribute by addressing an important, and as yet not fully solved, challenge to healthcare: how to exploit Internet technologies to improve quality of care, while protecting patient privacy and the confidentiality and integrity of sensitive medical data. The focus here is on authentication mechanisms using digital certificates not only for user identification but also for access control decisions and authorizations.

An NFC E-Ticket System with Offline Authentication

Wei Jeng Wu and Wei Hsun Lee [15] describe an e-ticket system which works in both online and offline schemes. The proposed solution for offline authentication requires a NFC phone (Android OS 4.4+) with the capability to receive e-ticket, to store it in a secure element and to transfer it to a NFC reader. A digital certificate can be applied for verification. On the assumption to use cryptography, the security part of the ticket is encrypted with public key of the issuer. Then, during validation, a reader decrypts the security part with private key to check its authenticity.

QRP: An improved secure authentication method using QR codes

Another solution is proposed by David Pastor Maestre [2] which aims an improved secure authentication method using QR codes and PIN codes. As previous, this also works in both online and offline schemes but in this case, an authentication method using two-factor authentication was developed: a trusted device (a mobile phone) that will read a QR code and that will act as a token, and a password known by the user.

3. IMPLEMENTATION

This chapter describes in detail both online and offline components that have been developed. The main idea consists in development PIN-protected Attribute Certificate's Wallet to store and manage certificates. Due to the lack of support in Attribute Certificate with BER-encoded ASN.1 format of the existing Java and PHP libraries, a library has been developed to parse certificates in this file format; its Application Programming Interface (API) is described.

3.1 Main Idea

As mentioned before, to provide a stronger identification and secure authentication for access control, one of the obligations is to grant a safe certificate storage. To that, it was developed an AC Wallet using a strong database encrypted in 256-bit AES. This solution was employed on Consent ID app created by AET Europe. This app already enables easy and secure access to applications using PKCs.

Assuming a user already owns a PKC to authenticate and access into Consent ID software, it allows to import an Attribute Certificate via QR Code or URL. This import will allow the user to have access to Attribute Certificates in both online or offline states, since it was stored in his/her Android device.

Two different approaches are proposed to perform the authentication:

- First, a person can request for authentication using a web solution; for example, a student, to buy movie ticket with 50% discount, can apply the Attribute Certificate issued by its educational institution.
- Second, and following the previous example scenario, to use the bought ticket, the movie theatre validator must only read the student's Attribute Certificate via QR Since the validator has access to the student's AC and photo, located in server, she/he can use that data to check the student's identity.

With this process, it is possible to identify the person and to ensure a strong authentication using Attribute Certificates. With two-factor authentication, a PIN-protected AC Wallet and a certification-based authentication, we can say that our goal – fraud reduction – can be achieved.

3.1.1 System Architecture

The main goal of this work is to provide a secure solution to authenticate via web browser in checking Attribute Certificates. AET Europe servers are used for the provisioning of PKC to register in this Consent ID software and to issue Attribute Certificates. Also, Google Cloud Messaging (GCM) is used to establish a notification mechanism between both ends: Web Consent ID client and Android Consent ID client.

In Figure 3.1, is shown which components are used by different applications and its workflow.

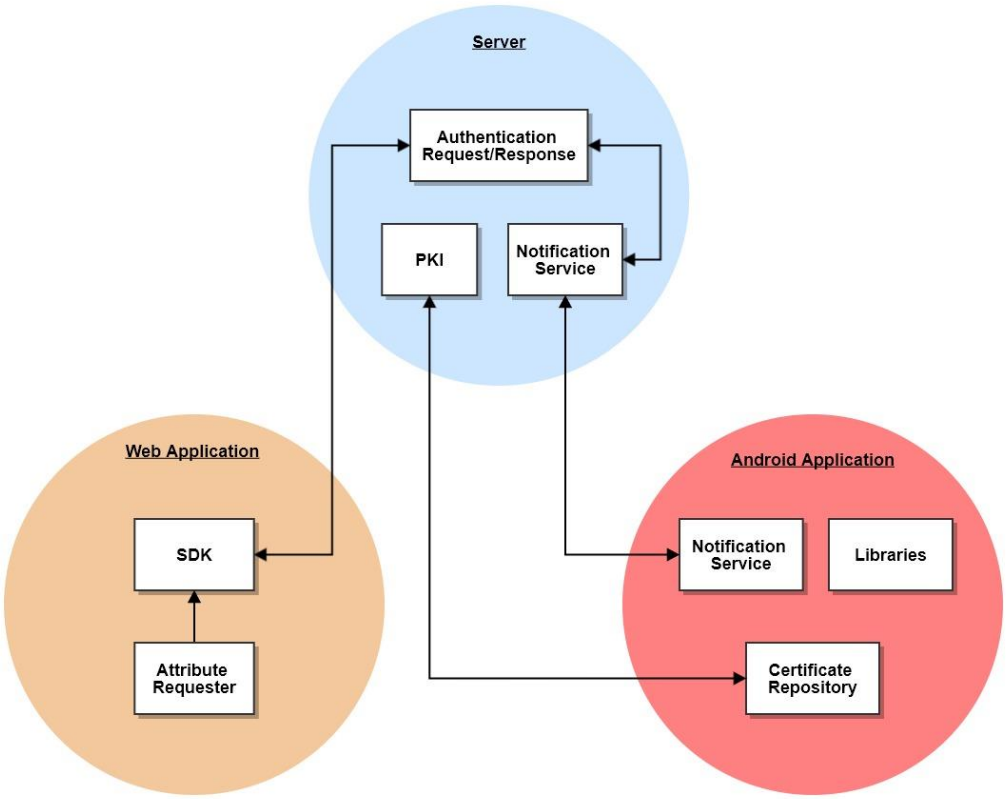

Figure 3.1 – System Architecture

Figure 3.2 presents the process of this system architecture for authorization requests using Attribute Certificates. At first stage, Web Consent ID client sends an Attribute Certificate request to the Consent ID server, which redirects it to GCM. Google servers store the received message and wait for the Android Consent ID client availability. As soon as the device is available, it will receive the notification.

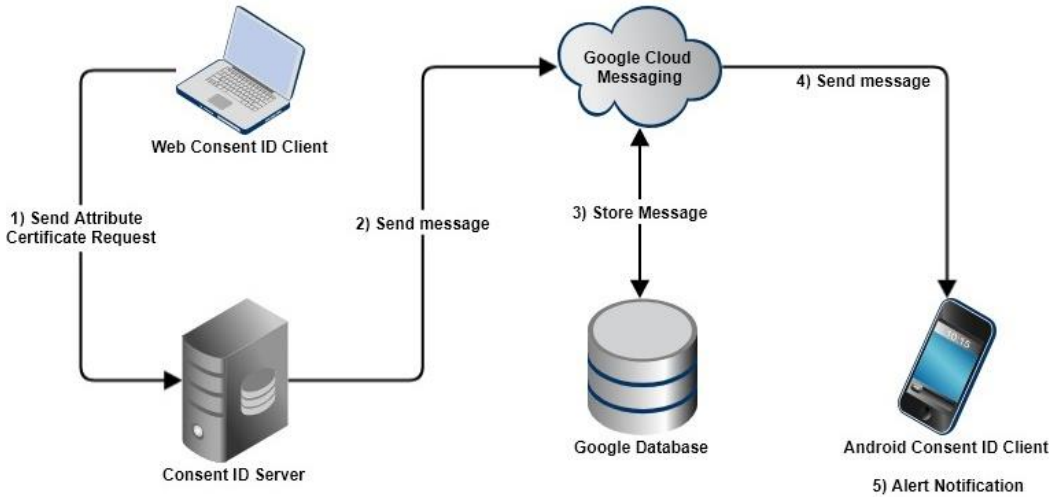

Figure 3.2 – Attribute Certificate Request on Web Client

After getting the notification, the Android application displays an Attribute Certificate list which is the same as AC's Wallet. By selecting one of the displayed ACs, the flow will follow the same logic as given in previous figure. In Figure 3.3, a system architecture for authorization response using Attribute Certificates is presented. Initially, our Android Consent ID application sends the message to GCM servers that will redirect it to Consent ID server, which will forward to Web Consent ID Client. Finally, the web application processes the received message and displays the authorization status.

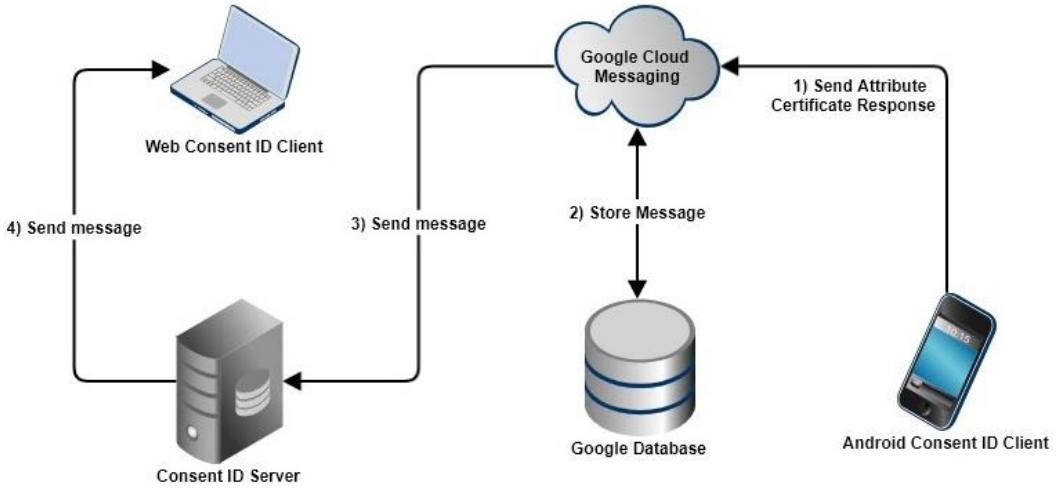

Figure 3.3 – Attribute Certificate Response on Android Client

In these last two architectures, it was presented the whole architecture workflow for authorization on the developed web application.

To support AC validations using Android application, it was implemented a mechanism so that a validator can read user's Attribute Certificate QR Code and then makes some local validations, such as: certificate AC integrity, issuer path verification, validity period, etc. When connected to internet, the validator has also access to revocation status and the user's face photography retrieved in Consent ID server.

When Attribute Certificate is issued, some attributes are set for later examination. In Figure 3.4, the validator checks the Attribute Certificate details and later will verify if an authorization can be granted given the attributes' content.

Figure 3.4 – Attribute Certificate Validation

3.2 Technologies Applied

3.2.1 Programming Languages and Tools

In this prototype two major platforms were used. First one, the well-known Internet and the second, Android OS by Google.

For web application, PHP, HTML, CSS and JavaScript were the programming languages applied, while the software Adobe Dreamweaver CC 2017 was an important tool to support the code development. The use of the free and open-source XAMPP was intended to run the web code and to make its results available either on the Internet or intranet.

For mobile application, Java, XML and JSON were used to develop an Android application using Google's official tool – Android Studio.

3.2.2 Libraries

Multiple set of libraries across different languages were extremely important to support, protect and provide multiple features that were implemented in this prototype.

Web application:

- Phpseclib [16] – PHP-based library that is responsible for reading X.509 certificates and to parse the ASN.1 standard structure (encoded in BER) into a readable information which was handled on our developed API.

Mobile application:

- Spongycastle – is the stock Bouncy Castle libraries with a few changes to adapt into Android environment. Bouncy Castle package is a java implementation of cryptographic algorithms, was used to parse X.509 certificates and to extend the developed API.
- ZXing – is an open-source, multi-format 1D/2D barcode image processing library implemented in Java. Useful to read and to create QR Codes.
- Greendao – is an open source Android ORM (object/relational mapping) making development for SQLite databases easier. It is used to update, delete, and query for java objects using this simple object-oriented API.
- Sqlcipher – extends the SQLite database library to add security enhancements that make it more suitable for encrypted local data storage such as on-the-fly encryption, tamper evidence, and key derivation. Used in our databases to provide a transparent 256-bit AES encryption of database files.

3.2.3 QR Code

A QR Code is a two dimensions' type barcode introduced by Japanese company Denso-Wave in 1994. [17] This type of barcode was designed to be decoded quickly and with high precision. It can support more than 3k characters in a tiny space and it has a special attribute for error correction. When QR Code is created, this can be generated in four different ways, by introducing some redundancy [18]:

- Level L (Low) – up to 7% of redundancy;
- Level M (Medium) – up to 15% of redundancy;
- Level Q (Quality) – up to 25% of redundancy;
- Level H (High) – up to 30% of redundancy;

As we can observe in Figure 3.5, using the same text in different types, when introduced some redundancy, the dimensions of barcode increased. Which means that the possibility of retrieving the content, when introduced an error, is also improved.

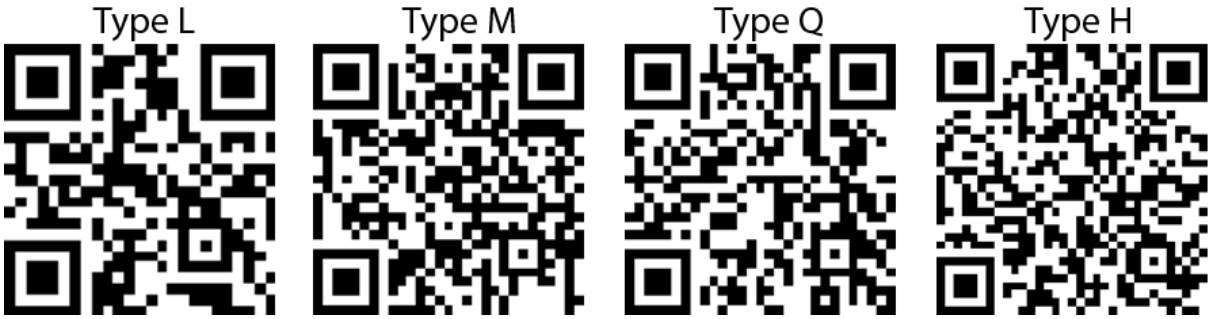

Figure 3.5 – Comparison between error correction types in QR Code

In Figure 3.6, we can observe that was introduced an error on different types of QR Codes, however, the content was successfully retrieved due to its redundancy.

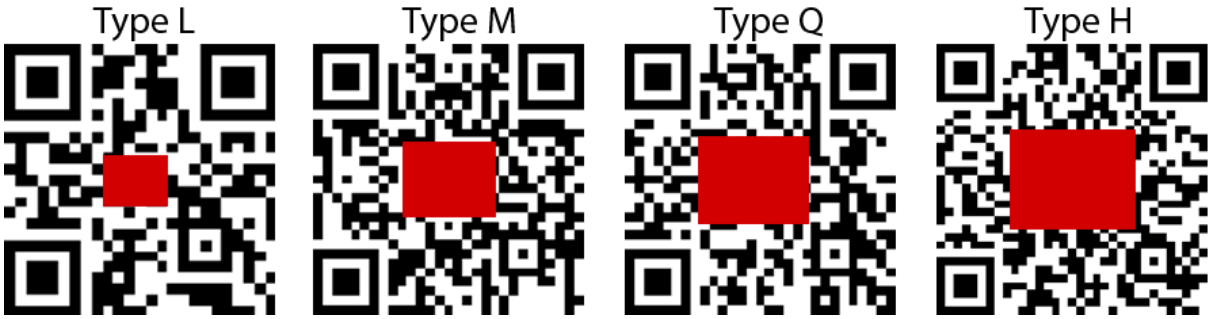

Figure 3.6 – Error introduced on different error correction types in QR Code

3.3 Android Application

3.3.1 Architecture

The Android application provides an Attribute Certificate Wallet with operations such as: importing ACs, delete ACs, check ACs status and connection with authentication services.

Google Cloud Messaging

At first, a messaging mechanism was used to swap notifications between server and client applications. Using this service, as shown in Figure 3.7, an Android application sends Application ID and Sender ID to GCM to register with its servers. When registered, the application receives a registration ID and sends it to the application server. This unique ID will be further used to recognize and to send a notification to the client device. [19]

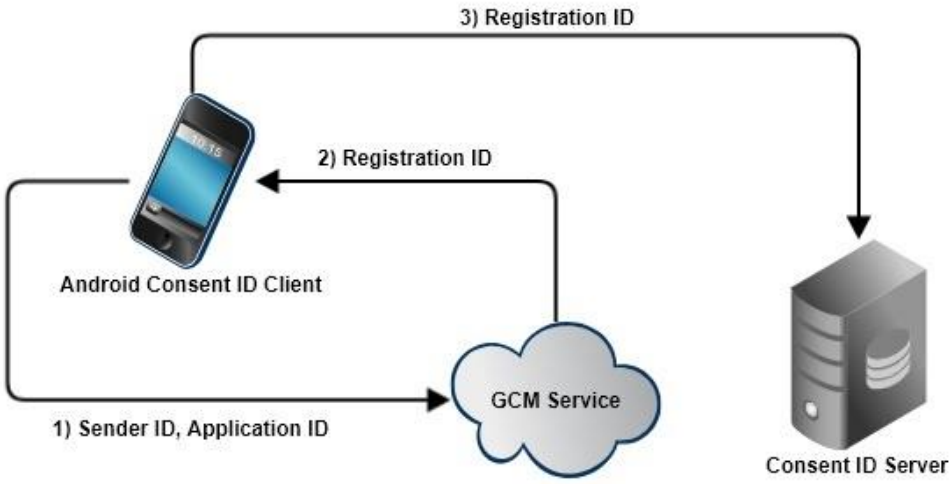

Figure 3.7 – Registration on GCM Service

Importing Attribute Certificates

To build the attribute certificates' Wallet, two different ways were settled to import them into the Android application. The first one is the cheaper and widely used QR Code, illustrated in Figure 3.8. Taking advantage of the fact that almost every Android device has an integrated camera, which allows for an easy reading from paper prints, screens, etc., this is the main way to import supported attribute certificates. Other benefit is that it can also be used in both online or offline scenarios.

Figure 3.8 – Import AC via QR Code

The second one, represented in Figure 3.9, is exclusively for online connections, consists in importing attribute certificates via a server. This method requires user/password for further connections to servers.

Figure 3.9 – Import AC via Server

3.3.2 Implementation

In this section, it is explained how the components are implemented. As the Android application is one of the main goals of the project, some figures are provided for better understanding. Figure 3.10 is presented a diagram describing the workflow features of the developed android application.

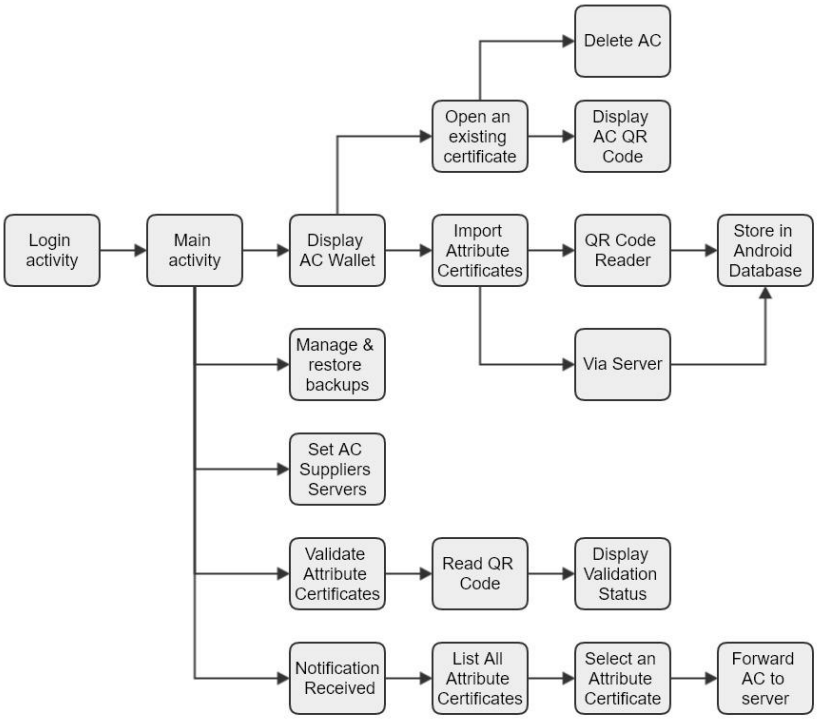

Figure 3.10 – Android Application Workflow

Databases

To implement the databases, the fastest ORM for Android [20] is used, the GreenDAO open-source library. The Backend Database stores certificates, to support and manage AC servers' suppliers. These database uses a 256-bit AES encryption to protect its files stored in Android phone. It consists of two tables presented in Figure 3.11.

Table AC is used to store certificates. The field acPath stores the entire attribute certificate using encoded in base64. This table is also used to provide a certificate alias to be later displayed in AC Wallet list. This alias is retrieved on holder's name given in attribute section of AC or fetch giving holder's CN.

Table ACServers stores information about end-entities data session's, to be later used in connections between AC suppliers' servers and Android device. The Boolean isChecked field is used to check if an AC Server is enable or not for connection. The fields, URL, port, username and password are used to establish that connection.

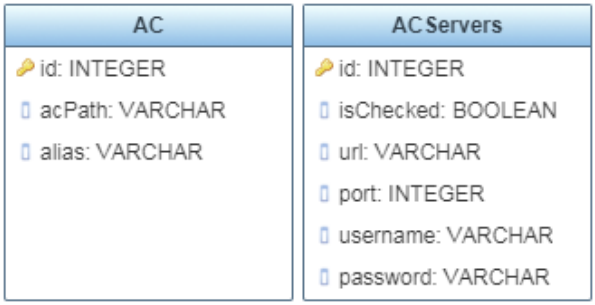

Figure 3.11 – Backend Databases

GreenDAO requires to create the schema of the table in the form of a class with greenDAO annotations. Figure 3.12 presents an example for AC table used in this implementation.

```
@Entity
public class AC {
    @Id
    private Long id;

    @NotNull
    private String acPath;

    @NotNull
    private String alias;
}
```

Figure 3.12 – AC Database: Generator Class

To explain some of the annotations used:

- @Entity: It defines the table name in the database;
- @Id: It defines the primary key of the table;
- @NotNull: It defines a not null constrain to the database column

In order to be able to work with the database objects, the project has to be compiled first. By doing that, the DAO classes and helper classes get created as shown in Figure 3.13.

```
@Entity
public class AC {
    @Id
    private Long id;

    @NotNull
    private String acPath;

    @NotNull
    private String alias;

    @Generated(hash = 771645180)
    public AC(Long id, @NotNull String acPath, @NotNull String alias) {
        this.id = id;
        this.acPath = acPath;
        this.alias = alias;
    }

    @Generated(hash = 1727253655)
    public AC() {
    }

    public Long getId() {
        return this.id;
    }

    public void setId(Long id) {
        this.id = id;
    }

    public String getAcPath() {
        return this.acPath;
    }

    public void setAcPath(String acPath) {
        this.acPath = acPath;
    }

    public String getAlias() {
        return this.alias;
    }

    public void setAlias(String alias) {
        this.alias = alias;
    }
}
```

Figure 3.13 – AC Database: Generated Class

Importing Attribute Certificates

As illustrated in Figure 3.1 – System Architecture, two ways are implemented to import attribute certificates into the application.

Importing by reading the barcode with the device's integrated camera is implemented by using ZXing library. Before initializing the scan, some parameters are defined, such as some text to be displayed when reading the barcode, a specific camera to be read (back camera) and disable beep sound when reading. The process to read QR Code is described in Figure 3.14.

Figure 3.14 – Importing via QR Code

The other way to import attribute certificates is to fetch them via the server. This is implemented by making a SOAP request to our server using URL, port, user and password given in AC Supplier settings. But since we use a SSL authentication to our server via PKC, firstly we must provide our client application with the certificate to establish the connection. When succeeded, it is necessary to parse all the information received and import it to our databases. In Figure 3.15, a diagram is given to explain this import.

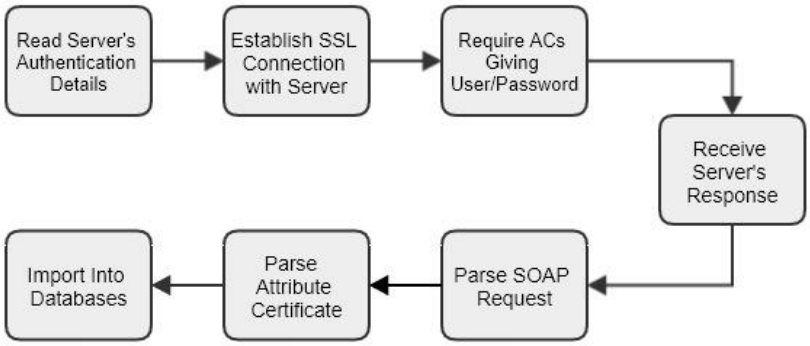

Figure 3.15 – Importing via Server

API

Due to lack support of Attribute Certificates using Abstract Syntax Notation One (ASN.1) encoded with BER format, an API was designed for Android applications. Existing libraries (e.g.: spongycastle) are only able to parse the entire certificate decoding from BER format to a

"readable" one. The API developed in this work extends spongycastle's to support a better and sustainable attribute certificate parsing solution. It was also added a list for known OID values and descriptions for a better user-experience.

The documentation for developed Android API can be found in 57Appendix A.

Backup and Restore AC Wallet and AC Supplier Settings

Another feature of this application allows users to create backups and restore from backups created earlier. For this to be achieved it is required an extra permission: to write on external storage. A dialog is displayed for such permission to be conceded. Figure 3.16 presents

To restore, it is displayed a dialog list with backups in memory, located in AET folder. When chosen, an alert is shown saying that the user is about to replace the current AC Wallet and AC Supplier' settings.

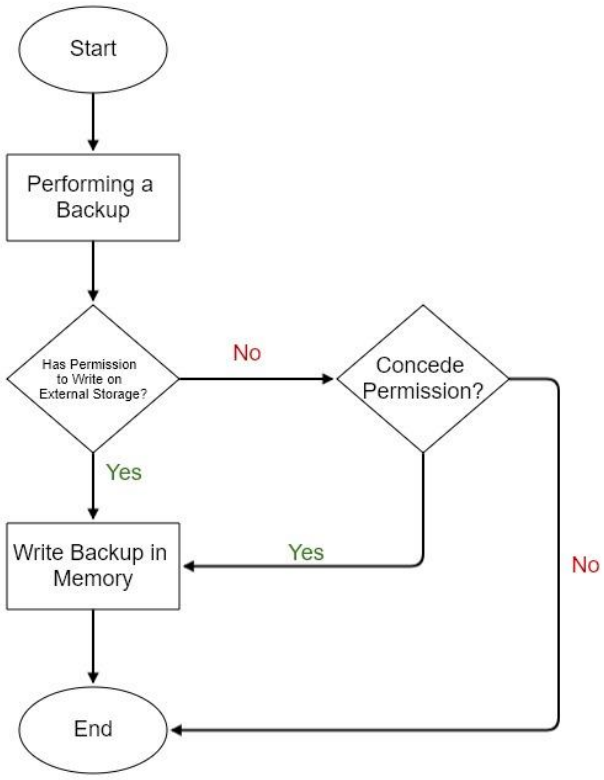

Figure 3.16 – Performing Backup Flowchart Diagram

Certificate Validations

Section 5 of RFC 5755 – An Internet Attribute Certificate Profile for Authorization [10] – defines a basic set of rules that all valid ACs must satisfy. In this Android application, those

rules are verified before any AC validation. When connected to the internet, these validations are instantly made once an attribute certificate is selected from the AC Wallet.

Online Mode

With internet connection, the user can experience all implemented features. One of the most important is attribute certificate revocation status. When an attribute certificate is selected from AC Wallet, its serial number is sent to the server for validation. The server implements two distinct behaviours: if the certificate is valid (positive response) it attaches the certificate owner's photo (assuming it is available) in the response, if negative, it sends an error message giving an informative error code. There are different error codes, such as "AC revoked", "this AC was not found on server's database" and "this AC has expired or is not valid yet".

Also, with internet connection, the retrieving of new certificates is available along with authentication requests to online services.

Offline Mode

Without internet connection, the user can afford the AC Wallet to check AC's information, to add new certificates via QR Code, to display or remove them from the list. This AC Wallet is still protected with the personal PIN code to access it.

Validation Mode

Regardless of user's internet connection, for QR Code validations only the validator must be connected to the internet. To validate any supported QR Code on Android application, the validator can use "Validate" option given in options menu. This validation method applies the same logic as used when an attribute certificate is opened from AC Wallet (with internet connection), which locally some validations are made and then the AC's serial number is sent to the server for extra validations.

3.4 Web Application

3.4.1 Architecture

For this component, it was developed a web application where a user could send an attribute certificate request to Android application via Consent ID servers to possibly authenticate in services. In this prototype, we didn't create any extra service (e.g.: bank account login) which

could take advantage of our implementation, but instead, we grant support for future services which want to use our authentication method.

Since this web application was intended to support online authentications, a workflow between web browser and Android application was developed to send a request to mobile client for authentication and to receive its response.

Authorization Request-Response

Giving a set of parameters, the request is generated and is sent to server which will process the message and will forward to some different servers before reaching the Android application as shown in Figure 3.2 – Attribute Certificate Request on Web Client.

Its response flows with same logic as previous as explained in Figure 3.3 – Attribute Certificate Response on Android Client.

3.4.2 Implementation

In this section, it is explained how the components are implemented. As the web application is the other main goal of the project, some figures are provided for better understanding.

Figure 3.17 presents an overview of the web application workflow.

Figure 3.17 – Web Application Workflow

API

Due to lack of support by existent PHP libraries of Attribute Certificates using ASN.1 encoded with BER format (like it happens with Android libraries), an API was designed to support them. This API extends phpseclib library with certificate parsing.

The documentation for the developed PHP API can be found in Appendix B.

Authorization Request

Before submitting any request, several parameters must be set. To reach our Android device, we must introduce the username so that the server can identify the target device and which attribute is being requested for verification. With that, some other extra validations are

available, such as: issuer certification, given OID and value to conform, mandatory OIDs, serial number and certificate hash. Once the request is sent, the application redirects to a new page to wait for response.

Authorization Response

In Figure 3.18, the authorization response flow is described. As soon as Android application send a response, the PHP functions parse it to validate the attribute certificate. First, it is made an attribute certificate parsing complying with the standards set in RFC 5755 – An Internet Attribute Certificate Profile for Authorization [10]. Then, some validations are made, such as: issuer PKC certification path and signature verifying. Finally, was implemented a function where a soap request is set and the connection is established with server to retrieve the AC associated photo. If it is valid, the web client displays the owner's photo along with all attribute certificate properties.

Figure 3.18 – Authorization Response Flow

4. VALIDATIONS AND TESTS

In this chapter, the results for the tests applied to the implemented features are presented, along with the work necessary to set up AC Wallet to authenticate into a service. Finally, the results are discussed and figures are given to explain what was accomplished.

The developed solution can be used in two possible scenarios: the user has internet connection and is able to communicate with server or the user is offline and cannot communicate with server. Each situation originates different behaviours in the Android application. Hence, it is analysed how it is possible to authenticate in both cases.

4.1 Online Analysis

Starting the Service

In order to allow user registration on the existing Consent ID system, a User ID and a Registration Code must be provided by the service owner or supplier company.

This initial enrolment process, is only available with internet connection as it requires access to several online services. The standard operation mode is composed by the following actions:

- Register the user into the Consent ID service
- Register the user into a CA
- Fetch a digital certificate (public key certificate – PKC) from the CA
- Associate the PKC to the user account

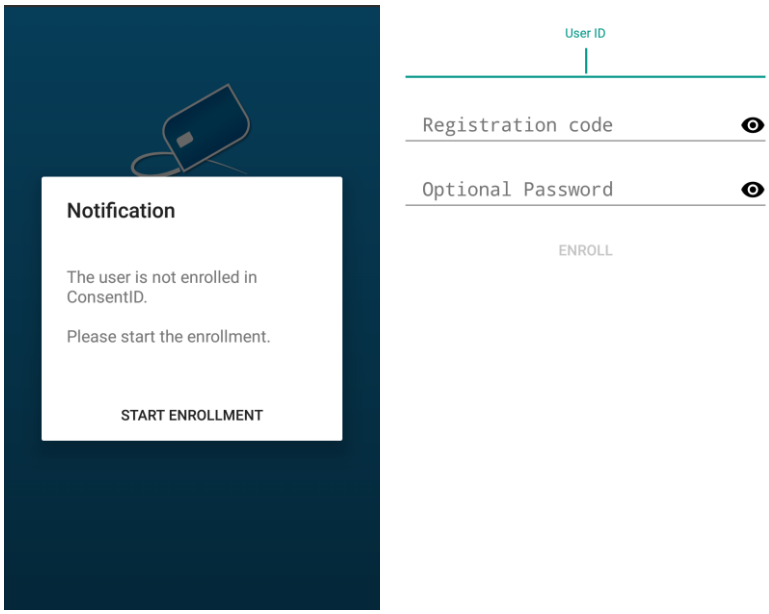

Figure 4.1 – Registration Process

In our prototype, we did not develop any PKI or any other certificate issuing service. Those services were made available by AET Europe and DigitalSign who are specialized in digital trust and security solutions. Using the provided registration details, the user is now ready to start the enrolment process and get access to the Android application.

The next step requires the user to set a PIN code that is needed to protect the application from unauthorized access. So, every time that this application is initialized, the same PIN code must be introduced. The same behaviour applies when an online service sends a notification to process the authorization. To be able to accept the attribute request and to choose an attribute certificate, the user must introduce the PIN code. If the user forgets its PIN code, unless he made a backup, all information contained-in cannot be recovered. The Consent ID app already has security mechanisms to avoid brute force PIN code attacks.

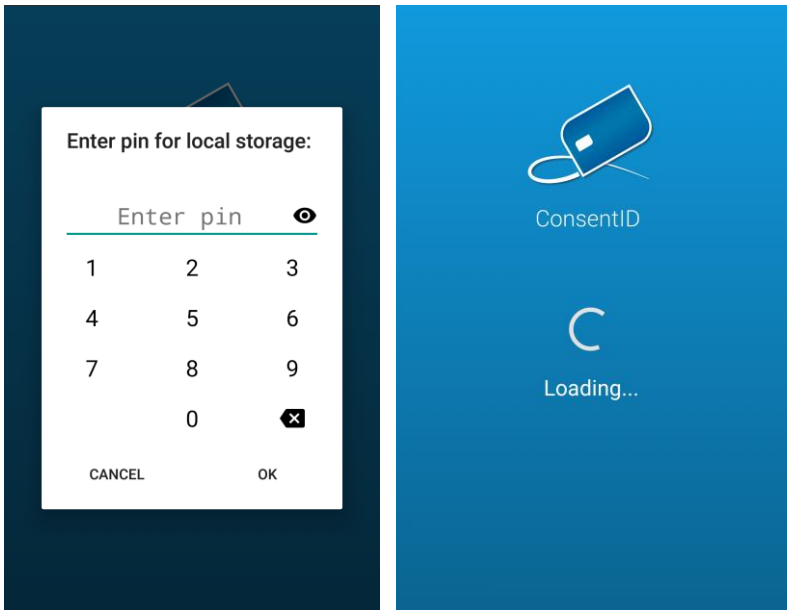

Figure 4.2 – Setting a Password Followed by Splash Screen Login

At this point, the user will be able to explore the different menus available in our Android application.

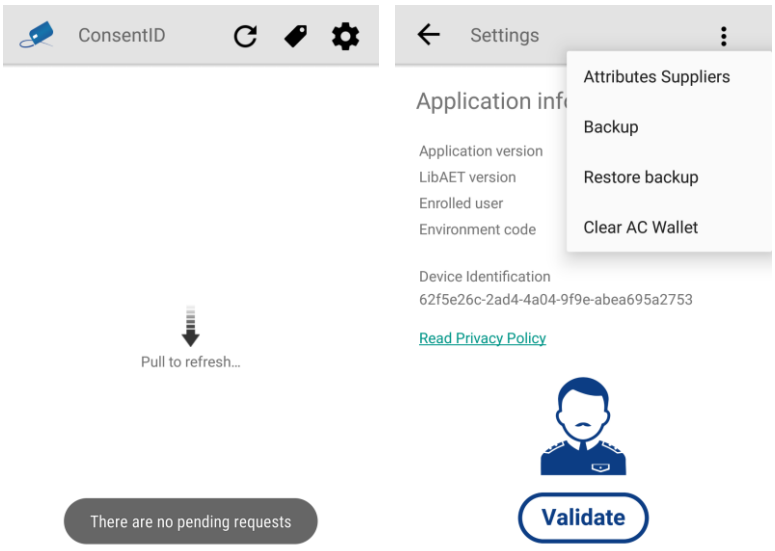

Figure 4.3 – Main and Settings Activities

Importing Attribute Certificates

Since there is no Attribute Certificates on our AC Wallet, first, it is essential to import one. As previously said, there are two different ways: importing via server or via QR Code.

When a service provider entity manages and delivers our services, to allow the user to import attribute certificates via server, it should first deliver to the client a set of detailed information so the user can download its attribute certificates that should be stored online. Another available alternative is to provide a printed QR Code ticket (which can be used for services like theatre ticketing), that can also be imported to AC Wallet. Using the QR Code to import the AC, removes the necessity for an online connection.

In Figure 4.4 is a working example of a client login details to retrieve attribute certificates from server.

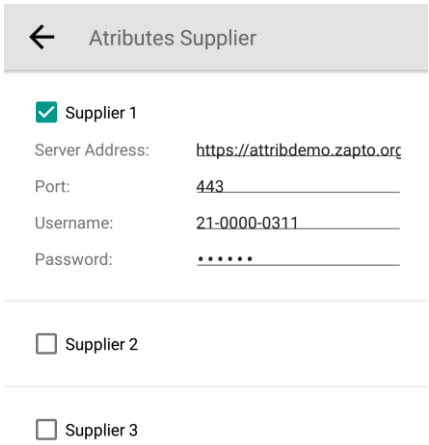

Figure 4.4 – Filling Attributes Supplier' Server Details

Now a user is up to download attribute certificates from server. Since AC Wallet is still empty and when clicked to access, a different activity is displayed so that the user can add new ACs.

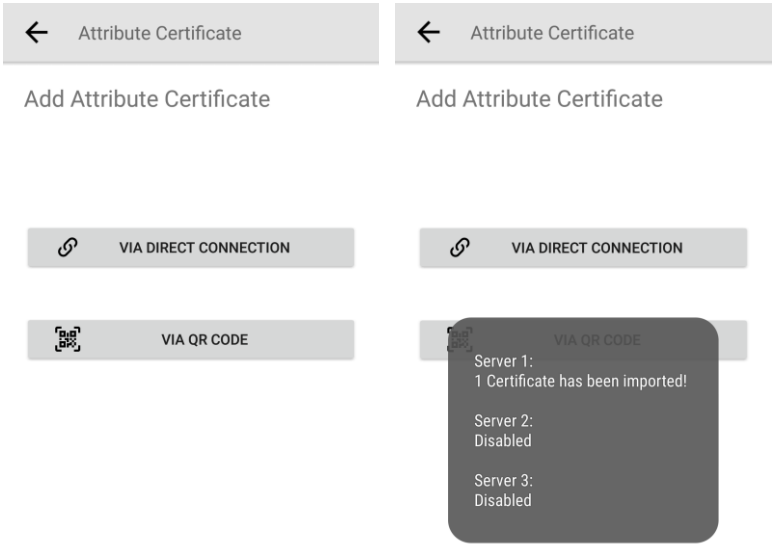

Figure 4.5 – Add New Attribute Certificates' Activity and Importing AC Via Server

By clicking “Via Direct Connection” the app will fetch new attribute certificates via server. It was also implemented an error control to warn the user when the server details are incorrect or if the imported AC already exists.

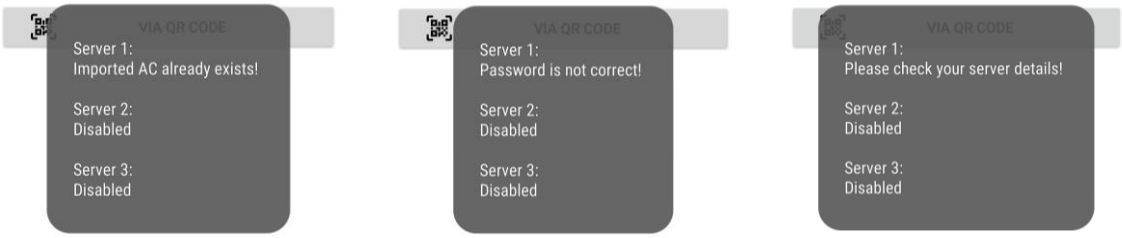

Figure 4.6 – Error Control When Importing AC via Server

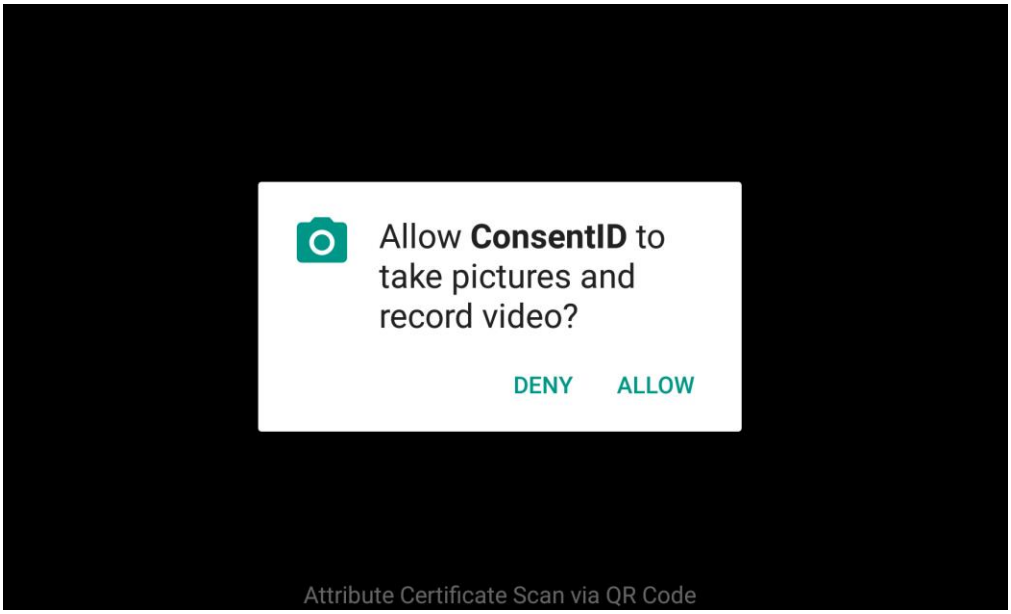

Figure 4.7 – Dialog to Grant Permission to Use the Device's Camera

The alternative QR Code import (which can also be used in offline mode), requires an extra Android permission to allow the application to read the barcode.

Figure 4.8 – Scanning an Attribute Certificate QR Code

When importing via QR Code, a verification on application database is made to assure the imported AC does not exists yet. If it already exists, an alert is shown, warning the user about this. Otherwise, the user will be redirected to a new activity where the user is asked to confirm the AC import and where it's possible to see the AC properties.

Figure 4.9 shows the alert dialog to give a possibility for user to analyse the attribute certificate before importing it into AC Wallet.

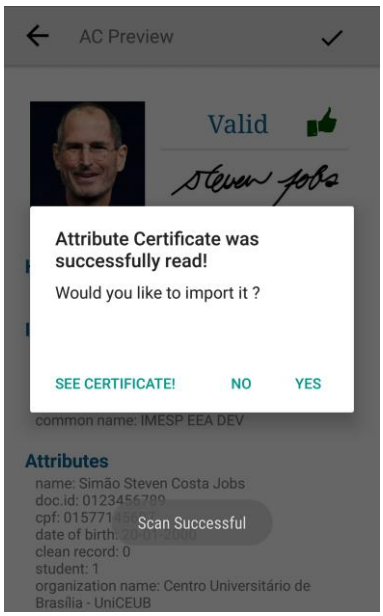

Figure 4.9 – AC Preview When Imported Via QR Code

Attribute Certificate Wallet

At this moment, after starting the service and importing the AC, the user has activated his AC Wallet.

As we can see in Figure 4.10, anytime that urge to import a new attribute certificate, the user should press the “Add Button” on top right and it will redirect to “Add Attribute Certificate” activity.

Figure 4.10 – AC Wallet with Imported AC Via QR Code

When an Attribute Certificate is selected, certificate properties are presented in detail. On AC properties, the user can check the following fields: holder, issuer, attributes, extensions, version, serial number, validity period and signature algorithm. As a prototype, those are the important fields to display. For a final solution, some display alternatives could be considered so the user can choose from different information levels (e.g. All, Short info, Only attributes, Validation info, ...) depending on their knowledge or needs.

In Figure 4.11, we can observe two features that are available inside each AC. Firstly it was pressed the “Delete” button and immediately a dialog window is shown asking if the user wants to remove the attribute certificate from his wallet. The second option is to display the attribute certificate QR Code to be used in further validations.

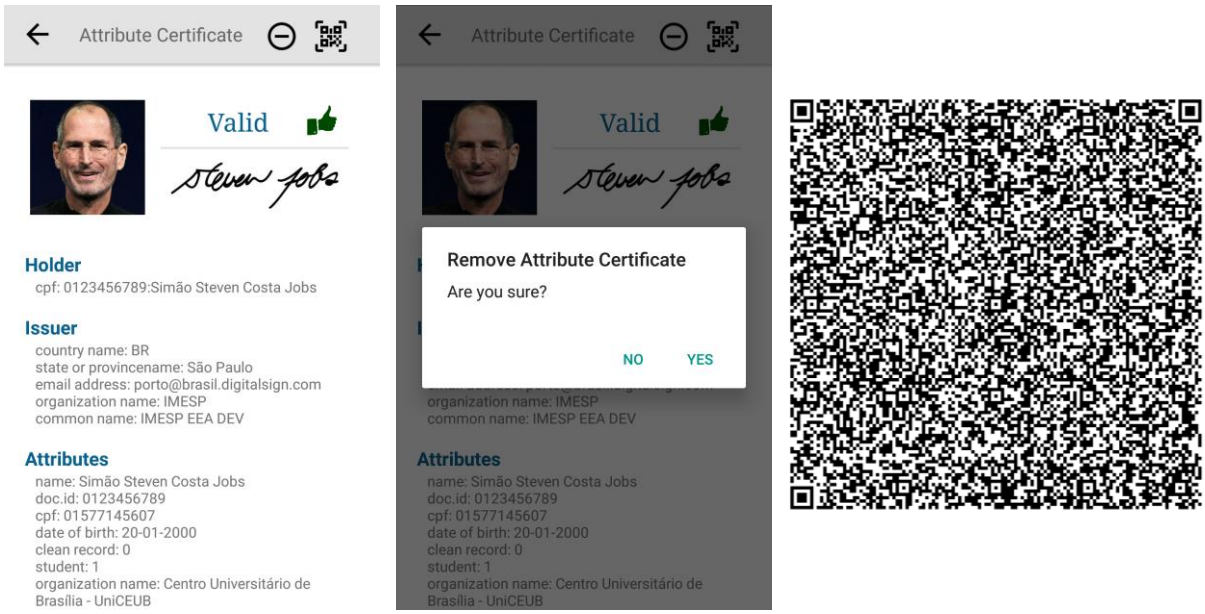

Figure 4.11 – Attribute Certificate Example Activity and its Features

Like every certificate, an attribute certificate is expected to be used for its entire validity period. This means that if a certificate is beyond from issuing and expiring dates, it is not valid for use and the validation should fail. However, a certificate can be invalidated before its current expiration date. When that happens, the CA adds the AC serial number and broadcast it through Certificate Revocation List (CRL). Once added into the CRL, a certificate should no longer be trusted by any entity. The CRL is requested by the Android application and the server sends it for the AC validation as we can see in Figure 4.12.

Another possible error could occur when an attribute certificate was imported whose issuer is not recognized by our Android application.

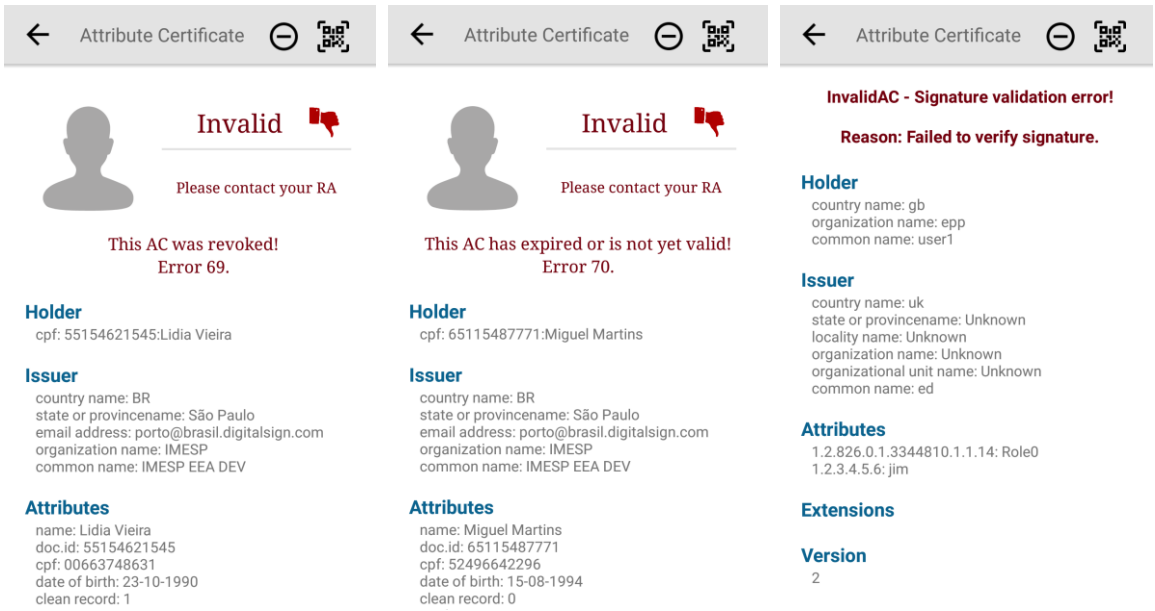

Figure 4.12 – Different Attribute Certificate Invalidations

Backup and Restoring

As required in Android environments, we also need to set up a dialog asking the user to grant permission to write and read from external storage. Our Android application creates a folder which will be used to store all the backups previously made. Figure 4.13 shows the dialog and after conceding the requested permission, a message is displayed informing the backup was successfully created.

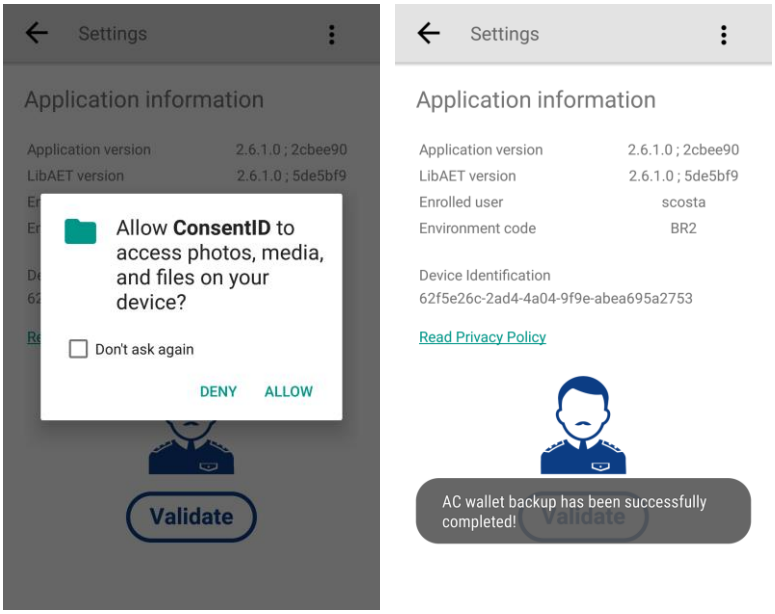

Figure 4.13 – Performing a Backup

Aligned with backup feature, we correspondingly hand out the restoring feature. The user will be capable to restore into the application an AC Wallet and AC Suppliers' settings. Evidently, when restoring, all existent data will be replaced.

In Figure 4.14, it is displayed all backups that one device has. Using a file explorer, we can observe that in AET folder exists four backups with filename prepared to be parsed by our application.

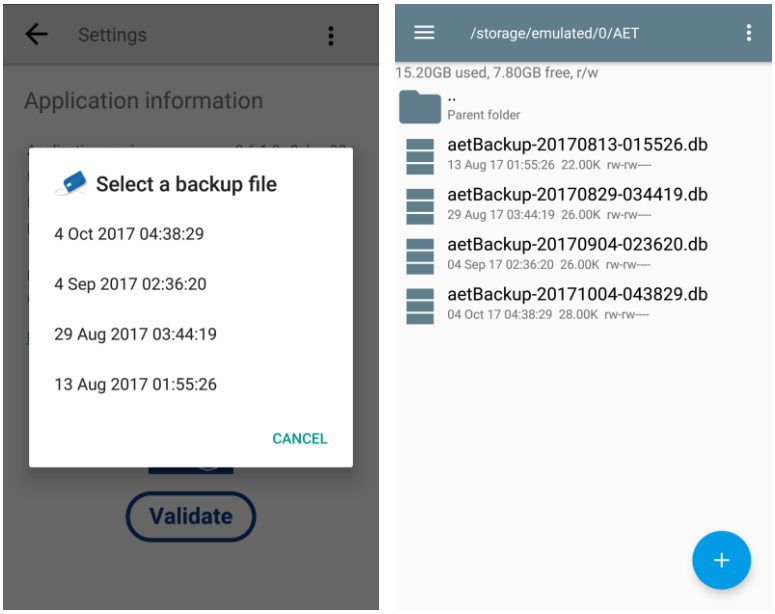

Figure 4.14 – Displaying All Backups in Memory's Device

Delete All Attribute Certifiactes

When the wants to stop using the attribute certificates, we allow the user, if desired, to wipe the AC Wallet. Note that this action will not affect the AC supplier settings, which will remain intact.

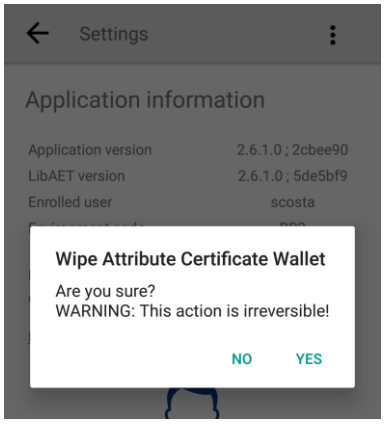

Figure 4.15 – Resetting AC Wallet

Checking Attributes for Authorization

To extend our Android application to support web services so they can use the Android's AC Wallet for authorization, we designed a PHP sample webpage where a user sends a request by specifying some fields, shown in Figure 4.16. In this example, we will verify the age majority of a given attribute certificate with no extra validations defined. In Figure 4.17 are presented all different validations options introduced for this sample webpage.

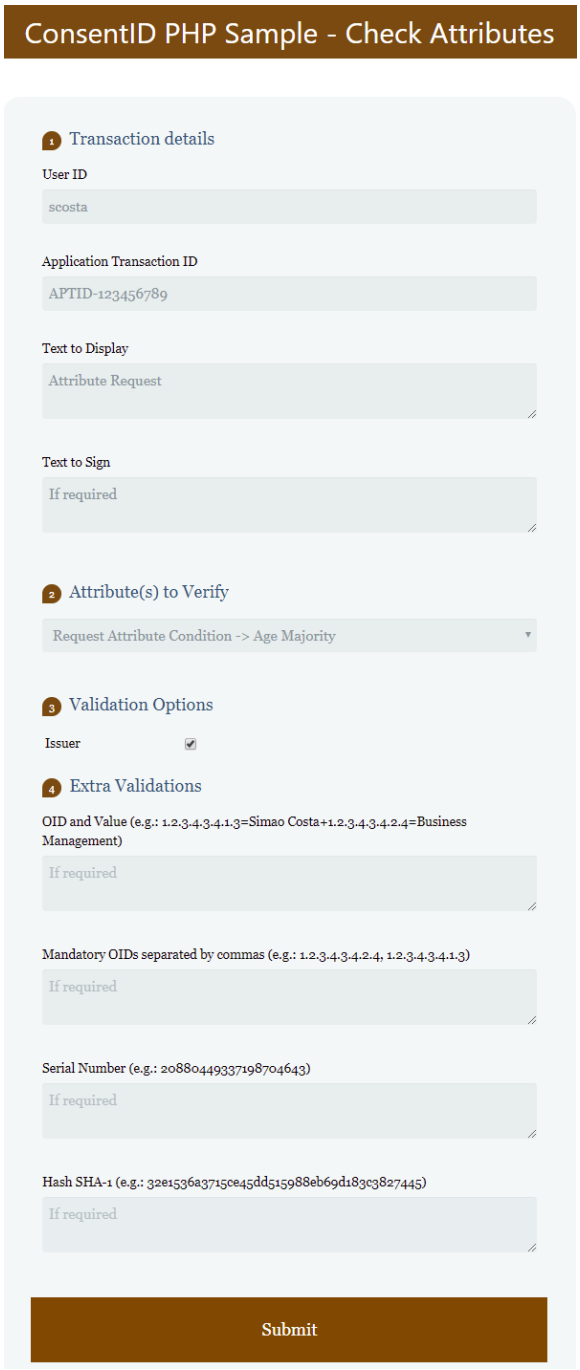

Figure 4.16 – Submit a Request for Attribute Verification

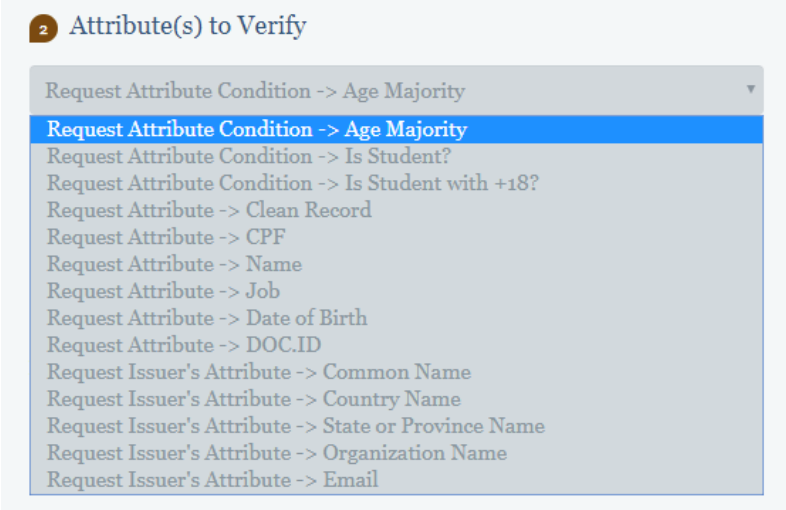

Figure 4.17 – Choosing an Attribute Field or Condition to be Requested

When the form is submitted on web application, automatically, our Android application receives a notification alert informing that there is an attribute request waiting for authorization. Conceding this authorization, will display a dialog where are shown all AC Wallet list entries. After choosing the proper AC, the authentication response and the selected certificate are sent to the Consent ID service and immediately to the PHP Sample service. Both certificate and attribute response are forward for security reasons. The double verification (Android app, and Web app) of an attribute request can avoid some fraudulent schemes in Android phone. For example: if a certain response by Android app is sniffed and changed, the later verification on the web application will reject the authorization giving an error message.

Figure 4.18 – Status Bar Displaying a Notification Alert

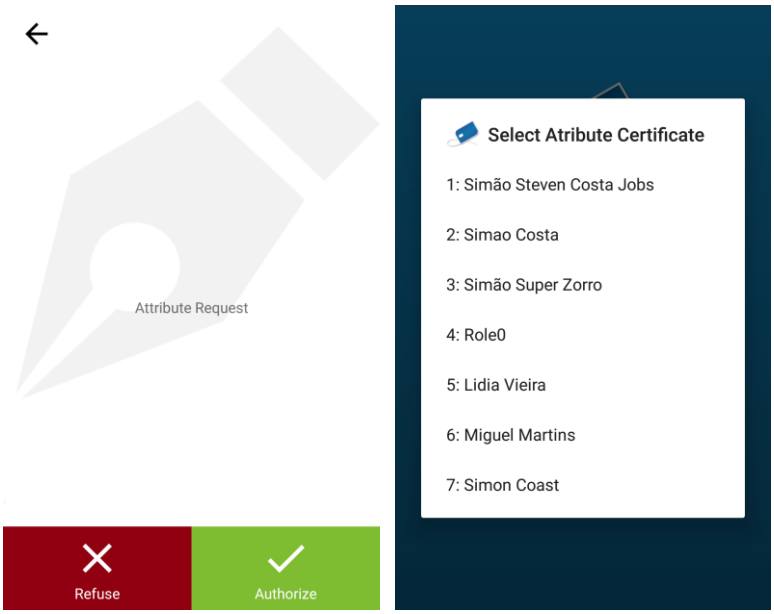

Figure 4.19 – Authorization Dialog and ACs List

Once received and handled by our web PHP-based mechanisms, it is shown the validation message. As we can observe in Figure 4.20, this attribute certificate is valid and checked against the attribute request. It was asked for owner's age majority and the final processed response was: Under 18. Along this information, on this web prototype, it can be displayed all attribute certificate details with enlarged personal photo, QR Code and signature by clicking on each picture.

Depending on Android device specifications and computer's, a single authentication could take 20 seconds, since the authorization is required from web browser, until the user acceptance in Android app when selecting an AC to authenticate.

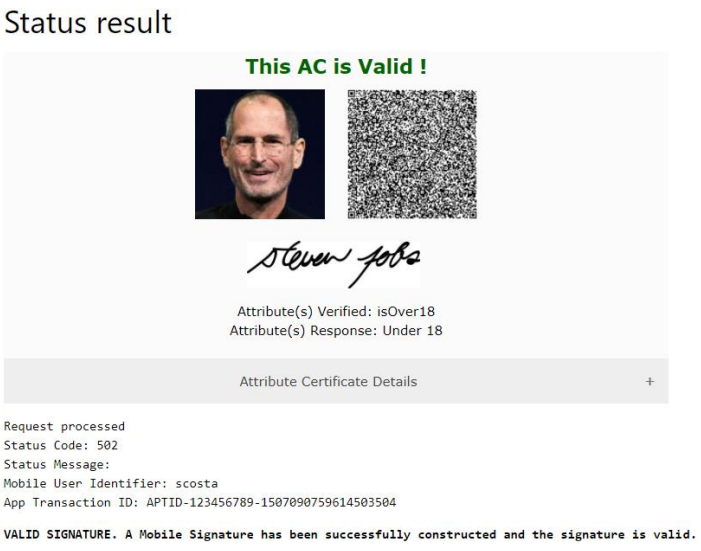

Figure 4.20 – Age Majority Validation Result

Error Handling in Authorization

Like every authorization system, here, an authorization can also be rejected. To demonstrate the rejection, we give a few examples where those denied requests are treated in our solution. In case of an empty AC Wallet and since every authorization requires an AC, when an attribute request is accepted by user, it is immediately displayed a message saying that AC Wallet is empty. Then, the error is automatically sent to PHP service which will display the error message, like shown in Figure 4.21.

Figure 4.21 – AC Wallet Empty When Verifying

In Figure 4.22, some AC validations errors are given. On these three examples, it was tested an AC that is revoked (which means that its serial number is on server's CRL list), an expired AC and an AC which the issuer information is not recognized by our implementation.

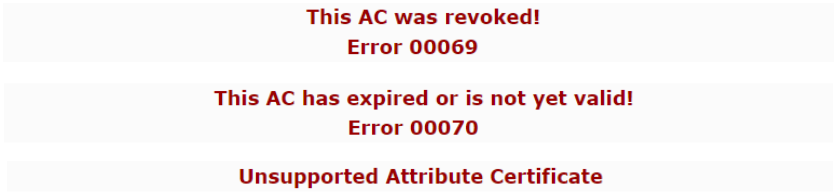

Figure 4.22 – Different AC Errors When Verifying

When asking for extra validations, there is a possibility that requested parameters does not match. In Figure 4.23, we give an example where it was set a mandatory OID and value which could not be found on certificate. In that scenario, an alert is displayed saying the given values doesn't exist and for that reason, any other validation will be done.

Figure 4.23 – Error When Specifying a Non-Existing Value/OID

Android Certificate Validation

To provide a validation feature using the same Android application, an option was developed so validators can read QR Codes from people whose possess attribute certificates. It can be used to concede a permission by checking an AC.

Figure 4.24 – AC Validate Option

4.2 Offline Analysis

To run an authentication in offline mode, only ones which its AC will be verified, can be disconnect from the internet.

This is a very particular process that involves a trust checking in the AC or its issuer. Since every single detail can be vital to grant any authentication, a photo or a signature evidence becomes very important in each validation. As such information is only located on servers, this means that the AC validator must be connected to the internet. However, when using offline scenarios, and taking this thesis title strictly, the main idea is to reduce fraud. And our work shows that fraud could be reduced if, by example, the service provider inside the proper time frame, fetch the CRLs and perform AC digital signature validation will surely reduce the attack rate. Additionally, a cache feature could be added to the android application, so it could store user photo and relate the image with the attribute certificate.

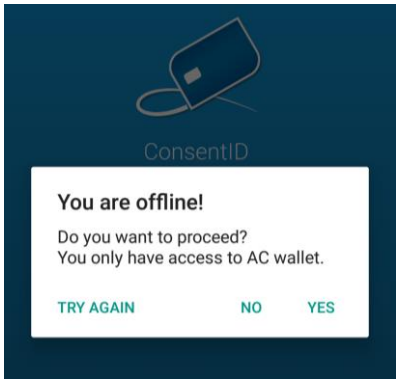

Figure 4.25 – Offline Login on Android Application

The application workflow is similar to the online mode. Being offline, some menus will be hidden from user. When starting, with same behaviour as online, a dialog is displayed to type the PIN code. After granted this authentication, instead of redirecting to main activity, this application goes immediately to AC Wallet where all attribute certificate is listed. And there, a message is displayed informing the user about offline mode. If the device gets internet connection, a dialog is displayed asking if the user wants to go online, or remain offline, as shown in Figure 4.26.

The user is still able to add attribute certificates to its wallet via QR Code, to remove them or to display any AC's QR Code.

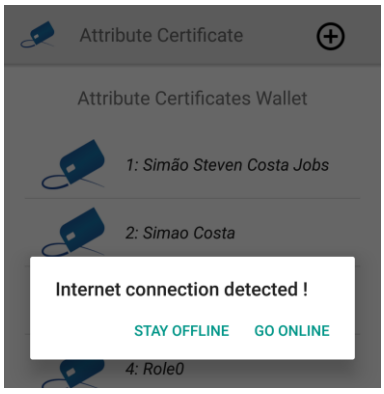

Figure 4.26 – Internet Connection Detected

4.3 Discussion of Results

With fraud reduction for authentication systems in mind, we developed Android and web based-PHP applications to demonstrate how to achieve better user authentication controls. For the

obtained results from our experiments and tests, it is safe to say that our solution can ensure an increased trust authentication system. Requesting users to have an attribute certificate associated with their digital certificate, will ensure proper and non-forgeable user identification.

With this solution, we demonstrated during the tests that an unknown certificate issuer could automatically be denied by our system, displaying a message alerting for the unsupported attribute certificate (or its issuer). Since in every authentication a certificate signature validation is made, to assure its integrity, any attempt to use a forged certificate (e.g. to change any field or data) will immediately be noticed and result in an authentication failure.

Both online and offline schemes offer features so a user can grant authentication to something. With internet connection, an authentication can be granted for services such as online banking, email and financial portals. Without internet connection, we can only support local authorizations via QR Code for services like ticketing and access control.

When reading an attribute certificate from QR Code, ZXing libraries found to be very efficient, precise and fast showing great results in its performance.

Since this thesis was developed with AET Europe, to ensure availability and reliability in authentications, we rely on them servers. During all this development period, it was not encountered any problem with their services.

Finally, and like any secure system, we cannot guarantee that our solution is unbreakable, but we can say that we present a strong authentication approach that increases trust and help reduce fraud. There are plans to productize this solution and some specific markets already demonstrated interest to test this solution.

5. CONCLUSION AND FUTURE WORK

In this thesis, a literature study is made regarding cryptography and Attribute Certificates, more specifically aiming to provide a secure authorization using X.509 Attribute Certificates. To achieve that goal, it was applied the basics of cryptography and researched about digital certificates, attribute certificates, PKI and authorization services through web services. During the research, we were capable of gathering good ideas were gathered from the related solutions and RFCs reported in the "Cryptographic Techniques in Authentication" chapter. Based on that research a solution was conceived for the abovementioned goal. The solution comprises three components – Consent ID Service, the Android application and the PHP demo platform, using the existing solution from AET Europe. An extended version of Consent ID application was developed adding functionalities and implementing the proposed architecture as explained in the "3. Implementation" chapter.

The developed prototype can be useful in near future since it is provided on top of an existing service, using a configurable and improvable authentication control. It can easily be enhanced and adapted to the market requirements so that it can be used in large scale for end-entities.

This work includes an API to read and parse attribute certificates that was not foreseen from the beginning. When deepening the study on these matters, it was found that the existing libraries for Java and PHP, do not support the studied format of attribute certificates.

5.1 Implementation Limitations

The developed solution does not consider any exceptions or problems like server load and high availability. As said, it relies on AET Europe service and assumes the base framework to work properly and manage any unexpected behaviours.

When the web application requests for an authorization, the elapsed time between sending a request and receiving the response is a little high, and it can be unpractical for authentications. This happens mainly due to the tests environment resources allocation (few CPU and available memory), however, for demonstrating purposes it is acceptable.

5.2 Future Work

For the developed software to be more than a prototype, some improvements should be made. There are some points that can be implemented in the future.

Link AC with PKC

It is helpful to consider this operational mode, as the workload is reduced and somehow trivial, however, due to operational limitations, such as no access to all systems' features (Consent ID service and application libraries), it was not possible to implement the desired deep linkage between the public key certificate (PKC) and the attribute certificate (AC). This feature would improve trust on the system and it is easy to extrapolate the usability results. For this to occur, the PKC should be used to request the AC. The AC will then be issued for the exactly same entity present in the PKC subject, so the identity is only one. This could prevent an inappropriate QR Code or unauthorized offline usage from anyone else. Also, it could be used to sign documents when specifying certain attributes.

Use CRL lists in offline scenarios

Since CRL/OSCP link is already included in the attribute certificates that have been researched, an appropriate application for those lists can be an important use for offline scenarios. Mainly for the validator (but also to apply for the client) when it does not have an internet connection available all the time, a synchronization mechanism could be implemented to fetch the CRL for later usage (e.g. daily sync in the morning when the Bus remains in the Bus station).

Extended backups to every personal information

Although is not strictly necessary for the purposes of this thesis, and as said in Implementation chapter, the backup feature only backups AC Wallet and AC Supplier list. There are other important data that can also be backed up, such as PIN code with associated PKC used for login and GCM Token used to communicate with Google servers for notifications.

Protect backups with password

If the backup feature comprises only the AC, as it is public and should be linked to the PKC, there is no big risk to have the backup in plain text (not protected by password). In the future, or if it is planned to backup also the PKC, password verification will be essential to ensure data protection. Even with an existent encrypted database, it does not protect databases from being stolen and used. With PIN code, will prevent from being restored and used in undesirable devices.

5.3 Conclusions

In this thesis, it was proposed a system to reduce fraud in authentication systems using attribute certificates. A solution is proposed to resolve the problem and it is included it in two major developments delivering functionalities to end-entities: the service requestor extension and the end-user mobile application. We have shown that this technology can be integrated on top of a traditional authentication procedure, taking advantage of the well-known, and every time more widespread, Public Key Infrastructure.

The initial configuration, mandatory in order to use the developed solution is dependent on the user registration on a Certification Authority (e.g. DigitalSign) to acquire a digital certificate. followed by registering into the AET service (Consent ID). After AET registration, a user name and a registration code will be provided. At this moment, the user will be "enabled" and ready to use and to authenticate using the developed application. We think that any person, without special IT skills, can understand how to use the system.

Security flaws may exist in the developed solution and it is not perfect yet. However, to be more than a prototype, some improvements should be made. The idea is to reduce fraud and, as is always important on the security side, keep to improving the solution, by increasing its security.

All the objectives proposed in the beginning of this work have been reached, reducing fraud by providing a secure and efficient way to authenticate in systems for both online and offline schemes and validating attribute certificates.

During the writing of this thesis, a similar solution was presented in Brazil [21], which consists in a physical driving license with QR Code printed in to avoid fraud. This QR Code can be imported into an application which displays its data contents. This experiment will start to be used in February 2018 [22]. However, this solution does not support any authentication feature. At same time, there are some European countries raising questions about the same issues that lead into this research. This solution was already demonstrated and very well accepted as a possible solution by AET Europe.

REFERENCES

[1] Intuit QuickBooks, "Fraud Statistics Every Business Should Know." [Online]. Available: https://quickbooks.intuit.com/r/trends-stats/fraud-statistics-every-business-should-know/. [Accessed: 06-Jul-2017].

[2] D. P. Maestre, "QRP: An improved secure authentication method using QR codes," *Univ. Oberta Catalunya*, pp. 1–11, 2012.

[3] R. Sandhu and P. Samarati, "Authentication, access control, and audit," *ACM Comput. Surv.*, vol. 28, no. 1, pp. 241–243, 1996.

[4] S. a. Thomas, "Security in Telecommunication and Information technology," 2003. [Online]. Available: http://www.itu.int/itudoc/itu-t/85097.pdf. [Accessed: 04-Oct-2017].

[5] L. Delgrossi and D. Ferrari, "The design of supranet security mechanisms," *7th IEEE Intell. Netw. Work. Proc.*, no. April 1998, pp. 167–173, 1998.

[6] S. Burnett, S. Paine, and O. Mcgraw-hill, *RSA Security ' s Official Guide to Cryptography*. .

[7] Microsoft, "Cryptography and Microsoft Public Key Infrastructure." [Online]. Available: https://technet.microsoft.com/en-us/library/dd277320(d=printer).aspx. [Accessed: 31-Oct-2017].

[8] P. Dornbusch, M. Möller, and A. Buttermann, "IT - Security in Global Corporate Networks," p. 292, 2002.

[9] ISO/IEC, "Information technology — Open Systems Interconnection — The Directory — Public-key and attribute certificate frameworks," vol. 2014, 2014.

[10] S. Farrell, S. Turner, and R. Housley, "RFC 5755 - An Internet Attribute Certificate Profile for Authorization," *Internet Eng. Task Force*, pp. 1–50, 2010.

[11] D. Cooper, S. Santesson, S. Farrell, S. Boeyen, R. Housley, and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile," *Netw. Work. Gr.*, pp. 1–151, 2008.

[12] ISO/IEC, "Information technology -- Automatic identification and data capture techniques -- QR Code bar code symbology specification," vol. 2015, 2015.

[13] Apple, “PassKit Web Service Reference,” 2015. [Online]. Available: https://developer.apple.com/library/content/documentation/PassKit/Reference/PassKit_WebService/WebService.html. [Accessed: 05-Oct-2017].

[14] I. Mavridis, C. Georgiadis, G. Pangalos, and M. Khair, “Access Control based on Attribute Certificates for Medical Intranet Applications,” no. 1, pp. 1–10.

[15] Wei Jeng Wu and W. H. Lee, “An NFC E-Ticket System with Offline Authentication,” *Information, Commun. Signal Process. 2013 9th Int. Conf.*, 2013.

[16] V. Khatavkar, “Phpseclib: Securely Communicating with Remote Servers via PHP,” 2016. [Online]. Available: https://www.sitepoint.com/phpseclib-securely-communicating-with-remote-servers-via-php/. [Accessed: 31-Oct-2017].

[17] Denso ADC, “QR Code Essentials,” *Denso Adc*, 2011. [Online]. Available: http://www.nacs.org/LinkClick.aspx?fileticket=D1FpVAvvJuo=&tabid=1426&mid=480. [Accessed: 04-Oct-2017].

[18] Qrstuff, “QR Code Error Correction,” 2011. [Online]. Available: https://blog.qrstuff.com/2011/12/14/qr-code-error-correction. [Accessed: 04-Oct-2017].

[19] Google, “Google Cloud Messaging: Overview,” 2016. [Online]. Available: https://developers.google.com/cloud-messaging/gcm. [Accessed: 03-Oct-2017].

[20] “Android ORM Performance 2016,” 2016. [Online]. Available: http://greenrobot.org/android/android-orm-performa. [Accessed: 02-Oct-2017].

[21] V. Bretas, “CNH passa a ter código QR-Code para evitar fraudes,” *Exame Brasil*, 2016. [Online]. Available: https://exame.abril.com.br/brasil/cnh-passa-a-ter-codigo-qr-code-para-evitar-falsificacoes/. [Accessed: 25-Oct-2017].

[22] E. Conteúdo, “CNH digital começará a valer em fevereiro de 2018,” 2017. [Online]. Available: https://exame.abril.com.br/brasil/cnh-digital-comecara-a-valer-em-fevereiro-de-2018/. [Accessed: 25-Oct-2017].

Appendix A

A.1 Java API Documentation

In this appendix is documented all functionalities available through our API developed in Java. This API library was designed to target API level SDK 23 (Android 6.0). The application is still able to run on older versions. The minimum API level require to run it is SDK 18 (Android 4.3).

Here is described the public classes, method names, descriptions, parameters and returns to support and parse Attribute Certificate with BER-encoded ASN.1 format.

Table A.1 – AttributeCertificateAET Class

Class Name	AttributeCertificateAET()
Description	Main class where the AC is firstly handled. Also used to reach other method's classes

Table A.2 – SetCertificate Method

Method Name	setCertificate()
Type	Void
Description	Giving an AC, it is made a parsing by ac info, signature algorithm and signature value
Parameters	byte[] cert, Context context

Table A.3 – AcInfo Method

Method Name	acInfo()
Type	AttributeCertificateInfo
Description	Pointer to AttributeCertificateInfo class
Returns	ACInfo

Table A.4 – SignatureAlgorithm Melhod

Method Name	signatureAlgorithm()
Type	String
Description	Gets signature algorithm

Returns	SignatureAlgorithm

Table A.5 – SignatureValue Method

Method Name	signatureValue()
Type	String
Description	Gets signature value
Returns	SignatureValue

Table A.6 – GetFile Method

Method Name	getFile()
Type	String
Description	Get AC file encoded in base64
Returns	ACfile_base64

Table A.7 – Oids Method

Method Name	oids()
Type	AttributeCertificateOIDs
Description	Pointer to AttributeCertificateOIDs
Returns	oids

Table A.8 – GetBrandAttributes Method

Method Name	getBrandAttributes()
Type	String
Description	Used to parse the request received from PHP side
Parameters	String str
Returns	response

Table A.9 – GetValuefromOID Method

Method Name	getValuefromOID()
Type	String
Description	For a given OID on AC, its value is returned
Parameters	String oid
Returns	value

Table A.10 – AttributeCertificateInfo Class

Class Name	AttributeCertificateInfo()
Description	Parse acInfo details into objects. Gets also available

Table A.11 – SetACInfo Method

Method Name	setACInfo()
Type	void
Description	Parse all attribute certificate's fields
Parameters	byte[] cert

Table A.12 – GetACInfo Method

Method Name	getACInfo()
Type	LinkedHashMap
Description	Gets all AC details
Returns	ACInfo

Table A.13 – Version Method

Method Name	Version()
Type	int
Description	Gets AC version
Returns	version

Table A.14 – Holder Method

Method Name	Holder()
Type	LinkedHashMap
Description	Gets AC holder details
Returns	holder

Table A.15 – Issuer Method

Method Name	Issuer()
Type	LinkedHashMap
Description	Gets AC issuer details
Returns	issuer

Table A.16 – Signature Method

Method Name	Signature()
Type	String
Description	Gets Signature
Returns	signature

Table A.17 – SerialNumber Method

Method Name	SerialNumber()
Type	BigInteger
Description	Gets serial number
Returns	serialNumber

Table A.18 – IsValid Method

Method Name	isValid()
Type	Boolean
Description	Verifies if AC is valid
Returns	isValid

Table A.19 – NotBeforeTime Method

Method Name	notBeforeTime()
Type	String
Description	Returns not before date
Returns	X509AC.getNotBefore()

Table A.20 – NotAfterTime Method

Method Name	notAfterTime()
Type	String
Description	Returns not after date
Returns	X509AC.getNotAfter()

Table A.21 – Attributes Method

Method Name	Attributes()
Type	LinkedHashMap
Description	Gets AC attributes details
Returns	attributes

Table A.22 – Extensions Method

Method Name	Extensions()
Type	LinkedHashMap
Description	Gets AC extensions details
Returns	extensions

Table A.23 – AttributeCertificateOIDs Class

Class Name	AttributeCertificateOIDs()
Description	Giving a JSON formatted file with recognized OIDs, this class will be used to retrieve a description from OID for better user-experience. Multi-language also supported. English and Portuguese OIDs descriptions lists are also available

Table A.24 – SetOidsfromJSON Method

Method Name	setOidsfromJSON()
Type	void
Description	Reads OID list from JSON formatted file
Parameters	Context context

Table A.25 – GetName Method

Method Name	getName()
Type	String
Description	Gets OID description from a given OID
Parameters	String oid
Returns	oid

Table A.26 – AttributeCertificateValidations Class

Class Name	AttributeCertificateValidations()
Description	Class where main validations will be handled

Table A.27 – Is18Older Method

Method Name	is18Older()
Type	boolean
Description	Verifies the AC's owner is over 18 years old
Parameters	String dateofbirth
Returns	true or false

Table A.28 – IsACSignatureValid Method

Method Name	isACSignatureValid()
Type	String
Description	Verifies if the AC signature is valid
Parameters	AttributeCertificateInfo acinfo, X509AttributeCertificateHolder X509AC, X509Certificate ACIssuerCert, X509Certificate ACIssuerCert2
Returns	ok if valid, error type if invalid

Appendix B

B.1 PHP API Documentation

In this appendix is documented all functionalities available through our API developed in PHP. This API library was designed for PHP version 5.6.30.

Here is described the public classes, method names, descriptions, parameters and returns to support and parse Attribute Certificate with BER-encoded ASN.1 format.

Table B.1 – AttributeCertificateAET Class

Class Name	AttributeCertificateAET()
Description	Main class where the AC is firstly handled. Also used to reach other method's classes

Table B.2 – SetBERCertificate Method

Method Name	setBERCertificate()
Description	Giving an AC, it is made a parsing by ac info, signature algorithm and signature value
Parameters	$AttributeCertificate
Returns	ok if valid, error type if invalid

Table B.3 – GetACbase64 method

Method Name	getACbase64()
Description	Get AC encoded in base64
Returns	base64_encode($_ACpath)

Table B.4 – AcInfo Method

Method Name	acInfo()
Description	Gets attribute certificate info
Returns	$_acinfo

Table B.5 – SignatureAlgorithm Method

Method Name	signatureAlgorithm()
Description	Gets signature algorithm
Returns	$_signatureAlgorithm

Table B.6 – SignatureValue Method

Method Name	signatureValue()
Description	Gets signature value
Returns	$_signatureValue

Table B.7 – Hash Method

Method Name	hash()
Description	Gets AC generated hash to be used in extra validations (if required)
Returns	$_hash

Table B.8 – AttributeCertificateInfo Class

Class Name	AttributeCertificateInfo()
Description	Parse acInfo details into objects. Gets also available

Table B.9 – SetAcinfo Method

Method Name	setAcinfo()
Description	Parse all attribute certificate's fields
Parameters	$acinfo
Returns	ok if valid, error type if invalid

Table B.10 – GetACinfo Method

Method Name	getACinfo()
Description	Gets all AC details
Returns	$acinfo

Table B.11 – Version Method

Method Name	version()
Description	Gets AC version
Returns	$_version

Table B.12 – Holder Method

Method Name	holder()
Description	Gets AC holder details
Returns	$_holder

Table B.13 – Issuer Method

Method Name	issuer()
Description	Gets AC issuer details
Returns	$_issuer

Table B.14 – Signature Method

Method Name	signature()
Description	Gets Signature
Returns	$_signature

Table B.15 – SerialNumber Method

Method Name	serialNumber()
Description	Gets serial number
Returns	$_serialNumber

Table B.16 – NotBeforeTime Method

Method Name	notBeforeTime()
Description	Gets not before date
Returns	$_attrCertValidityPeriod[0])

Table B.17 – NotAfterTime Method

Method Name	notAfterTime()
Description	Gets not after date
Returns	$_attrCertValidityPeriod[1])

Table B.18 – Attributes Method

Method Name	attributes()
Description	Gets AC attributes details
Returns	$_attributes

Table B.19 – Extensions Method

Method Name	extensions()
Description	Gets AC extensions details
Returns	$_extensions

Table B.20 – Oids Class

Class Name	Oids()
Description	Giving a formatted array with recognized OIDs, this class will be used to retrieve a description from OID for better user-experience

Table B.21 – GetName Method

Method Name	getName()
Description	Gets OID description from a given OID
Parameters	$oid
Returns	$value

Table B.22 – SignatureAlgorithmIdentifier Class

Class Name	SignatureAlgorithmIdentifier()
Description	Class with a list with OIDs and its descriptions. Use for a better user-experience

Table B.23 – GetSignatureName Method

Method Name	getSignatureName()
Type	String
Description	Identifies Signature description from a given OID
Parameters	$oid
Returns	$name

Table B.24 – ACValidator Class

Class Name	ACValidator()
Description	Class where all validations will be handled

Table B.25 – Validate Method

Method Name	validate()
Type	String
Description	Validates AC with every private method
Parameters	$validations
Returns	ok if valid, error type if invalid