Cybersecurity MSc: Applied Cryptography 2024-25

2nd Project: Selective Identity Disclosure

November 26, 2024 Due date: December 22, 2024

Changelog

- v1.0 - Initial version.

1 Introduction

Selective Identity Disclosure (SID) is a way to selectively disclose a group of identity claims, while proving their ownership.

Imagine the following: when you present you Citizen Card (CC) to anyone, you have no way to restrict the set of identity attributes that are disclosed to the person that sees it. Now imagine that you have a similar approach in a digital world, where you have a Digital CC (DCC) with exactly the same identity attributes that are imprinted in the CC. Such a DCC would be signed by an authority with jurisdiction over the attributes present in the DCC. However, it is easy to see that using the DCC without any extra protection releases all of its attributes, probably more than the DCC requester needs to know. Thus, the goal of this work is to create a DCC protection system so that its owner may:

- Disclose only a given set of its attributes;
- Prove the ownership of those attributes;
- Ensure the correctness of those attributes.

2 Homework

The work consists on implementing 4 applications:

- `req_dcc`: an application to request a DCC for a person. It is run solely by the DCC owner.
- `gen_dcc`: an application to generate a DCC for a person, given a request. It is run solely by the DCC issuer.
- `gen_min_dcc`: an application to show a selected set of DCC attributes; it produces a minimalistic DCC (minDCC) from a DCC. It is run solely by the DCC owner.
- `check_dcc`: an application to validate a minDCC. Anyone can use this application.

2.1 Structure of a DCC

A DCC is a structured document, such as a JSON object. It contains the following fields:

- A set of identity attributes. Each attribute has (i) a label (e.g. a JSON name), (ii) a value and (iii) a commitment value. This commitment value is is produced from the attribute value and some secret; thus, releasing the secret and the attribute enables a validator to produce

the commitment value and, therefore, to verify the correctness of the attribute value. The commitment value should be such that it does not reveal any characteristic of its original value (e.g. the length). Therefore, you must use a digest function to produce it.

Each commitment value must result from an operation combining (i) the name of the attribute, (ii) the value of the attribute, (iii) a pseudo-random mask. This pseudo-random mask must be derived from (i) a password and (ii) from the attribute name. Thus, different attributes will be masked with different values.

- The description of the digest function used for all attributes' commitment values.
- A public key of the owner of the attributes. This element should include some description of the nature of the asymmetric cryptosystem of this key.
- A signature of the DCC issuer over some of the previous fields: commitment values and public key. This signature must contain (i) the signature value, (ii) a timestamp, (iii) a description of the asymmetric cryptosystem used for signing and (iv) the issuer certificate.

The issuing of a DCC requires its owner to provide a DCC request, which is a list of attribute names and a list of masks, one for each attribute name, and the identification of the digest function used.

2.2 Structure of a minDCC

A minDCC is a structured document, such as a JSON object. It is produced from a DCC and it contains the following fields:

- An array with the original commitment values present in the DCC. This array will be used to validate the DCC signature.
- The description of the digest function used for producing all commitment attributes.
- A set of identity attributes that are revealed. The name identifies the attribute, the value is a tuple with (i) the attribute value and (ii) the mask that was used to hide it in the DCC. A validator should be able to combine all three items (name, value and mask) to produce one of the commitment values above referred.
- A public key of the owner of the attributes, as in the DCC.
- A signature of the DCC issuer, as in the DCC.
- A signature of the minDCC producer, which should be the DCC owner, over the previous fields. This signature proves that the minDCC was produced by the DCC owner. This signature must contain (i) the signature value, (ii) a timestamp and (iii) a description of the asymmetric cryptosystem used for signing.

A minDCC validator should be able to (i) validate the signature of the minDCC producer and (ii) validate the signature of the DCC issuer.

2.3 Implementation details

For implementing a DCC issuer, consider using an asymmetric key pair that you generate (and store) and a self-certified certificate for including in the DCC.

For implementing the DCC owner, consider using an asymmetric key pair that you generate (and store). If, on top of that, you allow using the asymmetric credentials available in the Citizen Card, you will be awarded a bonus. Using the Citizen Card requires using its PKCS #11 library.

For selecting identity attributes, consider using an arbitrary set of identity properties, including a photo, or extracting all the attributes present in the Citizen Card. This last option has no bonus and requires using the Citizen Card PTEID library.

For students that do not have a Citizen Card we can provide a test specimen. Also, smartcard readers can be borrowed.

3 Evaluation

The project will be evaluated as follows:

- Implementation of the 4 applications: 70%;
- Use of different algorithms in the `gen_dcc` applications: (i) digest function used for attribute hiding, (ii) cryptosystem used for the owner key pair, (iii) cryptosystem used for the issuer key pair, (iv) signature algorithms: 15%;
- Written report, with a complete explanations of the strategies followed and the results achieved: 15%
- Bonus for using key material from a Citizen Card: 10%;

4 Homework delivery

Send your code to the course teachers through Elearning (a submission link will be provided). Include a small report, with no more than 10 pages, describing the decisions and strategies taken and implementation examples (not a complete copy of the code developed!) to illustrate your text.

Every piece of code imported from anywhere must be stated in the report and in the code itself. Failure to do so will be penalised.

5 References

- *Middleware Oficial de Identificação Eletrónica em Portugal - Cartão de Cidadão, da Chave Móvel Digital e Sistema de Certificação de atributos profissionais*, `https://github.com/amagovpt/autenticacao.gov`
- *Manual do SDK – Middleware do Cartão de Cidadão*, `https://amagovpt.github.io/docs.autenticacao.gov/manual_sdk.html`