# SEGURANÇA INFORMÁTICA EM REDES E SISTEMAS

Instituto Superior Técnico – Alameda

Group A27

## Smartphone as a security token


Diogo Nunes - 85184

Clara Pereira - 197070

João Figueiredo - 197071

## I - PROBLEM INTRODUCTION

The world has reached a digital revolution that brought many advantages to our everyday life. Technology has become easily accessible and an important factor to fully integrate modern societies. More specifically, the smartphone is now a *must-have* piece of technology, which allows the common folk to have on their pocket a device with impressive hardware architectures and ever-updating software packages.

With the age of digital technology, comes the age of data, privacy and confidentiality. We protect our devices with complicated passwords, use biometric features to encrypt our files and are more than ever concerned with the privacy of our digital footprint.

The next step in this new privacy awakening is to use the ever-present smartphone device in our pocket as a local authentication token, allowing for seamless authentication protocols and seamless encryption/decryption of data. This would offer the possibility of automatically authenticating a person by the mere presence of that device in the local physical area, which would improve confidentiality, privacy and usability of digital technology.

However, the design of such system must tackle a set of cybersecurity concerns. The main concerns refer to the **authentication** of the smartphone, **the encryption/decryption** of the data files and a possible **key exchange**, and the **high availability** of the service which is listening for new connections with said smartphone (if this service becomes unavailable, the user no longer has legitimate access to his files).

### Example scenario

In order to highlight the advantages of a system that would solve the problem exposed before, we describe an imaginative scenario where such solution existed:

(1) On a first level the user installs both the smartphone and computer applications. On the computer app, the user pairs the smartphone (on the app), giving the device authentication credentials, and defines which files shall be decrypted by the newly defined token.

(2) By default, the computer app maintains the defined files encrypted while there is no authenticated connection with the specified smartphone on the local network. The computer app is constantly listening for new connections.

(3) When there is an authenticated connection with the smartphone, the computer app seamlessly decrypts the user-defined files and makes them available to the user.

(4) When the connection is broken, the files are immediately encrypted. The computer app returns to step 2.

## II - PROPOSAL

In order to solve the exposed problem, we propose the design of a system which consists of a high availability service running on a local machine, which encrypts and decrypts files based on authenticated smartphone connections (for which it is constantly listening on the local network) and a smartphone app (Android) which sends requests for connections with the said service and implements the custom communication protocol.

### Proposal architecture

The main units (presented in Fig. 1) are the Server and the Client and how the common case communication flow executes.

The Server is a Java program which is launched on a computer, encrypts the previously selected files (during the setup phase) and opens up a Bluetooth port, actively listening for connections. When a client (smartphone) requests for a connection, the Server requests the Client to authenticate itself – in case it fails to do so, the connection is dropped and the Server returns to the first stage. In case the Client successfully authenticates itself, while the connection is maintained alive, the Server decrypts the files. When this connection is broken, the Server encrypts the files and returns to the first stage.

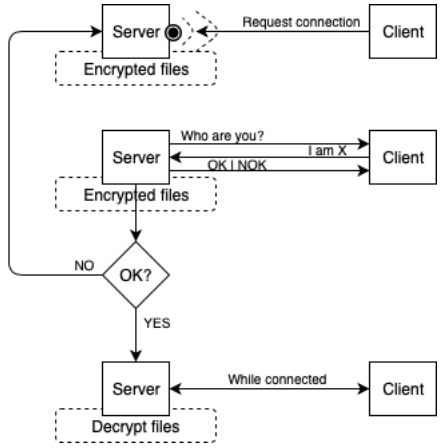


Fig.1 – High-level architecture diagram and usage flow

## Security Risks

In this section, we identify the security risks that the presented proposal attempts to address and present them in the following Misuse Cases Diagram (Fig. 2):

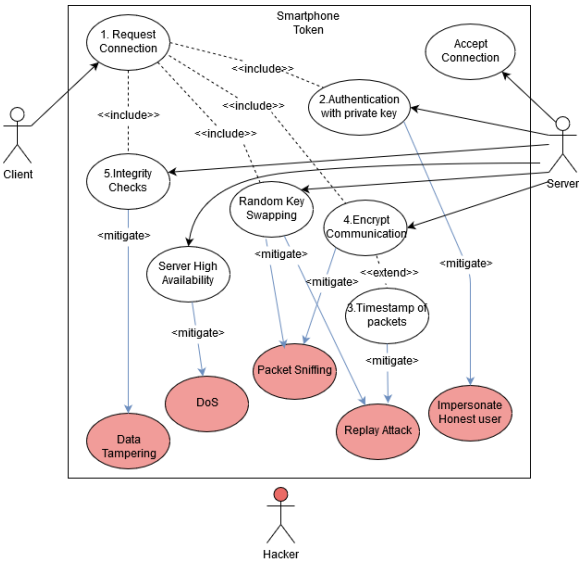


Fig.2 – Misuse diagram of our system

## Assumptions and Trusted Entities

In our proposal we assume that:

- The phone pairing (initial setup) is done by the real user.
- Encrypted files are not deleted
- All devices have Bluetooth

In our proposal, the following trusting levels apply (by default, we trust no one):

- Fully trust the Kerberos Protocol
- Semi-trust the user of the authenticated phone (might be a robber)

## Requirements and Proposed Solutions

| **Code** | **Proposed solution** |
|---|---|
| R01<br><br>*Availability* | Fault-tolerant server with replication and exchange of I'M ALIVE messages + whitelisting of devices |
| R02<br>*Authenticity* | Kerberos authentication protocol |
| R03<br>*Confidentiality* | Public-Private key encryption |
| R04<br>*Integrity* | Digest functions |
| R05<br>*Non-repudiation* | Digital signature and access/authorization logs |
| R06<br><br>*Freshness* | Attach a timestamp to each exchanged packet and use it as a validity check |

| **Code** | **Proposed solution** |
|---|---|
| R01<br><br>*Availability* | Fault-tolerant server with replication and exchange of I'M ALIVE messages + whitelisting of devices |
| R02<br>*Authenticity* | Kerberos authentication protocol |
| R03<br>*Confidentiality* | Public-Private key encryption |
| R04<br>*Integrity* | Digest functions |
| R05<br>*Non-repudiation* | Digital signature and access/authorization logs |
| R06<br><br>*Freshness* | Attach a timestamp to each exchanged packet and use it as a validity check |

## References

We plan to develop the Server in Java and the Client in Android. Both will use the network interface (Bluetooth).

We will also use de Kerberos Authentication Protocol and a set of encryption techniques (not symmetric).

## III - DESIGN VERSIONS

In this section we present the different iterative versions of our work in order to develop the proposed solution.

### Basic (B)

The basic version includes only the core (security) requirements and usage flow steps (on top of which later iterations will be built).

Usage flow steps:

- Server local network listening and Client connection request
- Server-Client custom handshake protocol
- Server-Client encrypting key exchange
- Server encryption/decryption of the selected files

Requirements:

- All communication requirements: R02, R03, R04 and R06.
- Data files confidentiality and integrity: R03 and R04.

### Intermediate (I)

The intermediate version will include the implementation of two important usage flow steps and respective requirements, which have to do with server high availability and improved security in the encrypting key exchange.

Usage flow steps:

- Server replication against system faults
- Client whitelisting
- At every new connection with a given client, the encryption keys are changed in order to minimize leakage risks.

Requirements:

- Server high availability: R01
- Client authenticity: R02

### Advanced (A)

We have decided to include in the advanced version additional usage flow steps, which considerably extend the usability of the system. This extended usability follows exactly the same set of security requirements as previous versions, so we will omit them.

Usage flow steps:

- Allow for the registration of multiple devices so that each can encrypt, and decrypt different set of files present in the same machine
- Implementation of access-authorization-change logs

## IV - WORKPLAN

| Week | DN | CP | JF | Topic |
|---|---|---|---|---|
| **4th Nov** | Server | Server | Client | B |
| **11th Nov** | Server Replication | Client whitelisting | Key swapping | I |
| **18th Nov** | Server Replication | DoS Prevention<br>Multiple devices | Key swapping<br>Multiple devices | I<br>A |
| **25th Nov** | Access logs<br>R | Multiple devices<br>R | Multiple devices<br>R | A<br>Report |
| **2nd Dec** | R + P | R + P | R + P | Report + Presentation |
| **11th Dec** | DELIVERY = CODE + REPORT | | | |

In the table above, the codes have the following meanings:

DN – Diogo Nunes

CP – Clara Pereira

JF – João Figueiredo

R – Report

P – Presentation

B, I, A – Each of the versions

It is important to note that each member is responsible for designing component tests for each of his/her implemented component. Equally relevant, when each version is complete, integration tests are implemented – the development of the next version will only start after the previous has been fully tested.