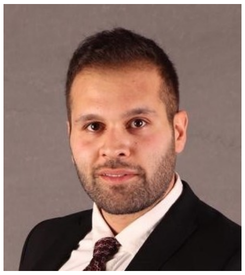
Smartphone as a Security Token

Seamless Multi Factor Authentication

Network and Computer Security
Alameda
Group 28

Afonso Gonçalves
89399

Emil Njor
98073



Farzad Terhanian
98425

January 13, 2022

1 Problem

When multiple people require access to the same door, e.g. to get into the office building of a company, it is often desirable to not distribute physical keys to everyone. The standard solution for this problem today is to use a shared key/PIN which is entered at a keypad near the door to get access. This system has one major flaw: It is very easy to find the PIN and gain access to the building.

A more secure alternative is to use biometric data to control access. These alternatives are however more expensive than regular keypads. Furthermore, biometric data cannot be fully relied on since it can get compromised and it is expensive to replace due to its scarcity. Fingerprint scanners have the additional drawback that everyone is touching the same surface, which can spread diseases.

Multi-factor authentication improves security by adding extra authentication layers. As such we do not have to rely on just one method of authentication. However, increasing the authentication steps often makes life harder for the users. This results in users slacking on the security, which presents new issues. Therefore, it is important to consider the amount of interaction required when implementing new security systems.

The core of the problem is to just allow some people to pass through a door. We can do this by using physical keys, PINs, biometric data, etc, but they all have security flaws when used alone.

1.1 Requirements

In order to implement this secure system, the following requirements should be considered:

Security Requirements:

- The user should be authenticated using more than one factor of authentication;
- The channels used to authenticate a user must ensure integrity, authenticity and freshness.

Non-Security Requirements:

- The system should not require more interaction from the user than e.g. using a physical key.
- The system should be cheaper to implement than having biometric sensors installed at doors.

- The system should not have a shared surface where diseases can be spread.

2 Proposed solution

2.1 Overview

The core of our solution is to use asymmetric keys to establish authentication. The private key would be stored in the user's smartphone and unlocked with the phone's fingerprint scanner. This allows us to both save costs on fingerprint scanners, and to have no shared surfaces or keys. The communication will rely on Bluetooth technology, ensuring that the user is near the door. The only interaction required is when unlocking the private key with the user's fingerprint.

The resulting security depends on three authentication factors [AZE09]:

- *"Something you have"*: The solution requires the users to carry their smartphones to gain access through the door.
- *"Something you are"*: The user's fingerprint is unique and very difficult to replicate.
- *"Somewhere you are"*: The Bluetooth's low range ensures that the user is near the gate while authenticating.

The gate is opened after the client encrypts a random nonce with his private key, as shown in Figure 1. Since the gate only opens when all these factors are met, this solution provides Multi-Factor Authentication.

2.2 Trust assumptions

This solution fully trusts the Gate and assumes that its Bluetooth interfaces correctly read incoming messages and that the gate only opens itself for authenticated users. Furthermore, it assumes that the mobile application is correctly implemented and that the private keys are unique and kept private. It also trusts the smartphone fingerprint scanner, and the fact that the clients' public keys were securely registered at the door. It is also assuming that the generated Nonces are secure random numbers.

However, it is not fully guaranteed that the authentication factors are met, therefore these factors are only partially trusted. The Bluetooth communication channel is not trusted, and we assume that an attacker can do whatever he wants with messages sent through the channel.

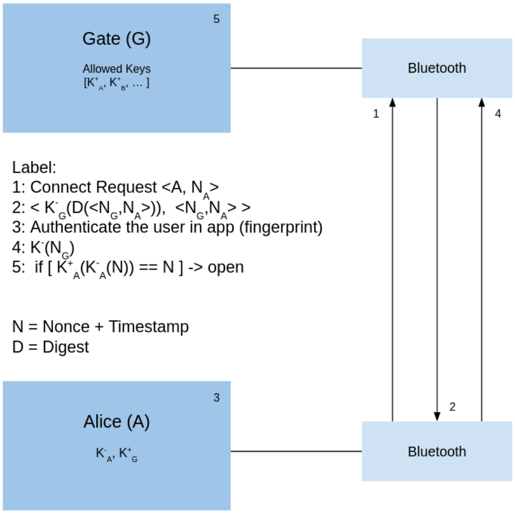

5
Gate (G)
Allowed Keys
[K^+_A, K^+_B, ...]
Bluetooth
1
4
Label:
1: Connect Request <A, N_A>
2: < K^-_G(D(<N_G,N_A>)), <N_G,N_A> >
3: Authenticate the user in app (fingerprint)
4: K^-(N_G)
5: if [K^+_A(K^-_A(N)) == N] -> open
N = Nonce + Timestamp
D = Digest
3
Alice (A)
K^-_A, K^+_G
2
Bluetooth

Figure 1: Interaction overview

2.3 Deployment

The solution will consist of a mobile application, developed in Java for the android platform, that connects to the gate over Bluetooth. The gate is controlled by a *Raspberry Pi*, which is listening for incoming requests. There will also be a privileged application that is allowed to add new users to the gate's white-list.

2.4 Secure channel(s) to configure

Since our project focuses on developing a secure channel over Bluetooth, its configuration is also explained in section 2.5.

2.5 Secure protocol(s) to develop

Authentication is the key security property of this project: Without it, the gate loses its purpose. Consequently, **Integrity** and **Freshness** are required. Confidentiality will not be focused on since there is no sensitive information in the communications.

The protocol to open the gate is described as follows and as depicted in Figure 1:

1. The user (A) gets close to the door and his smartphone sends an

opening request to it. This request includes his owner's ID and a Nonce (N_A) composed by a unique secure random number and a timestamp.

2. The gate (G) checks that it has never received N_A before.
3. G sends back to A a new Nonce (N_G) and the received N_A. This message must be digitally signed by G
4. A verifies that the signature is valid and that the received N_A equals the sent one.
5. A ciphers N_G with its private key ($K_A^-(N_G)$) and sends it back to G.
6. G deciphers the received message ($K_A^+(K_A^-(N_G))$) and checks if it is equal to N_G. If if is, the gate will open

The main protocol depends on a correct public key distribution. The proposed solution assumes that there are administrator accounts that are allowed to add public keys to the gate's white list. It is assumed that the public keys are correctly delivered to the administrator.

3 Plan

3.1 Versions

During the next few weeks we will be developing a **base**, an **intermediate** and an **advanced** version of the application described. The Base version will implement the process of opening the gate, and will expect the public keys of authorized individuals to be previously registered in the system. The base version is vulnerable to replay attacks and to an attacker faking a nonce from the server. In the Intermediate version we will be adding integrity and authenticity to the message 2 sent to protect against these attacks. Finally, the advanced version will allow key distribution at run-time.

3.2 Effort commitments

We will have four weeks to develop this project, during the period from the 16th November to the 11th of December. We expect to spend the first two weeks implementing the base version, the 3rd week implementing the intermediate version, and the 4th week implementing the advanced version.

Week\Person	Afonso Gonçalves	Emil Njor	Farzad Terhanian
W0 (09/11-13/11)	RPi development	Bluetooth communication	Android development.
W1 (16/11-20/11)	Android ↔ RPi	Local fingerprint authentication	Private/Public key encryption/decryption
W2 (23/11-27/11)	Configure RPi as gate	Secure nonce generation	Write Report
W3 (30/11-04/12)	Message Integrity	Update Report	Message Integrity
W4 (07/12-11/12)	Update Report	Change door permission @runtime	Change door permission @runtime

References

[AZE09] F. Aloul, S. Zahidi, and W. El-Hajj. “Two factor authentication using mobile phones”. In: *2009 IEEE/ACS International Conference on Computer Systems and Applications*. 2009, pp. 641–644. DOI: 10.1109/AICCSA.2009.5069395.