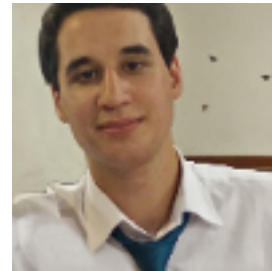Instituto Superior Técnico — Taguspark

Network and Computer Security

Smartphone as a security token (Proposal)

Group 18

ist173891 — David Gonçalves
david.s.goncalves@tecnico.ulisboa.pt

ist194104 — Leonardo Troeira
leonardo.troeira@tecnico.ulisboa.pt

ist194122 — Francisco do Carmo
francisco.baptista.carmo@tecnico.ulisboa.pt

1   Problem

Password-based authentication has the advantage of being simple and convenient to use. However, there are too many attacks that can be performed remotely that render this authentication mechanism not secure enough on its own—phishing, XSS, SQLI, rainbow tables, dictionary attacks, brute force... to name a few.

As such, if we're seeking stronger security mechanisms, one option is to make use of multi-factor authentication, usually involving knowledge and possession. This poses a trade-off between security and usability however, since the user will be needing to carry something with him all the time (e.g. smartphone).

Besides authentication mechanisms, we're also faced with the challenge of data possibly being stolen, either by sniffing network packets or acquiring control over the victim's file system, among other possible attacks. Therefore, assuring that data is stored and transmitted securely (relying on encryption algorithms) is a must. And again, for additional security, we should have another factor of authentication.

1.1   Requirements

1. The web server must be immune to XSS and SQLI.

2. Communication between client and server must be done through a custom secure channel.

3. The system must provide two-factor authentication.

4. User files must be stored encrypted.

5. User files must be decrypted only if the user has their smartphone nearby the client application.

6. The system must provide fault tolerance. If the system crashes after decrypting the files, make sure that the files are encrypted again.

7. Communication between the smartphone and client application must be done using a secure protocol.

1.2   Trust assumptions

Our problem assumes that the network and user's device cannot be trusted. However, we assume that the user's device is not compromised while it is in use, and that the smartphone and server are not compromised either.

2 Proposed solution

For our solution, we shall create a simple website—on a server machine—with a register and login forms, for which we shall obviously sanitize user input and make use of SQL parameters in order to prevent XSS and SQLI respectively. The user's password shall be stored hashed and salted in order to prevent it from being easily obtained by potential attackers. As an additional security layer for authentication, we shall use 2FA with the help of Duo SDK, thus leveraging the smartphone's proximity to the user as part of an increased security solution. The server shall also log any user actions.

The user shall be able to download files, from the server's website, which come encrypted with the user's public key. The user's key pair shall be generated at the registration phase, and the private key shall be stored in the user's smartphone, which will necessarily need to be connected to the user's device in order to be able to receive it, and we shall make a simple Android app for this purpose. The smartphone and user's device shall connect through Wi-Fi, utilizing a secure protocol (e.g. WPA). For extra security, the user's private key should be symmetrically encrypted with a password, which should be different than the user's login password, and may be delivered to the user upon login.

Additionally, any communication between the user and the server shall be done using symmetric encryption, with a random symmetric key which is delivered to the user encrypted with his public key, in order to guarantee confidentiality and integrity. Any requests from the user shall have appended a timestamp and a nonce, and shall also be signed with his private key in order to guarantee authenticity and non-repudiation.

Additionally to the server's website, we shall create a simple client application for viewing these encrypted files that were downloaded from the server. In order for the client application to decrypt these files, the smartphone must be connected to the user's device. The system shall be fault tolerant—as soon as this connection is lost, the client application shall clean its state, thus the user shall no longer be able to view the encrypyed files.

3 Plan

3.1 Versions

- **Basic:** For our basic solution we shall create a website with a register and a login forms, immune to XSS and SQLI attacks, that stores passwords hashed and salted, creates a key-pair for each user upon registration and gives it to them. The server shall also be able to encrypt trivial text files—context doesn't matter—with the user's public key and allow the user to download them. We shall also create a simple client application (typically JavaScript on the website) so that these files can be viewed by the user.

- **Intermediate:** On top of the basic version, we shall implement 2FA using Duo SDK and add another use for our smartphone, which is proximity detection. The private key shall be stored on the smartphone, we'll design a simple Android app to take care of this communication, which shall use a secure Wi-Fi protocol. We shall also and log user actions on the server side.

- **Advanced:** Finally, to complete our solution, we shall guarantee that as soon as the connection to the smartphone is lost or the client application crashes, the encrypted files shall no longer be visible. Additionally, we'll design our own custom secure channel for communication between server and client, as explained in the previous chapter. We'll also encrypt the user's private key with a random key generated by the server that is provided to the user's application upon login.

3.2 Effort commitments

	73891 - David Gonçalves	94104 - Leonardo Troeira	94122 - Francisco Carmo
4 Nov - 10 Nov	Create Server/Client Sides	Create Android App	Create Server/Client sides
11 Nov - 17 Nov	Create secure channel between server and client	Configure 2FA with Duo SDK	Create secure channel between server and client
18 Nov - 24 Nov	Add security to data with cryptography	Add security to data with cryptography	Configure standard secure channel between smartphone and server
25 Nov - 1 Dec	Add fault tolerance to server	Add fault tolerance to server	Add fault tolerance to server
2 Dec - 8 Dec	Test System/Correct bugs	Test System/Correct bugs	Test System/Correct bugs
9 Dec - 11 Dec 17H	Write Report	Write Report	Write Report

4 References

- **Duo SDK:** Duo's trusted access solution is a user-centric zero-trust security platform to protect access to sensitive data at scale for all users, all devices and all applications.

  https://duo.com/product/every-application/supported-applications/apis

  Found: Yes

  Installed: No

  Tested: No