Beyond the Password

Ricardo Almeida
Instituto Superior Técnico
Lisbon, Portugal

ABSTRACT

Nowadays, most of our devices are used to store our personal information and the conventional security mechanism to securely store this data is password. In this report are explained the problems associated with storing personal information when using passwords as a data security mechanism. The workaround for this problem is to use biometrics, rather than passwords, to protect our data within devices. The purpose of this work is to implement a system, called BioAuth, to perform second-level authentication requests using biometrics. To achieve this objective it is first carried out an investigation of already existing works that use biometrics and then an analysis of these same works, where it is concluded that the biometrics most suitable to implement in mobile devices are fingerprints, facial recognition and speech recognition.

Finally, is explained the system architecture and implementation, how the system tests were performed and what can be added to the system in the future.

Author Keywords
Biometrics; Authentication; FIDO UAF; Security; Cryptography; Xamarin;

ACM Classification Keywords
• **Security and privacy~Biometrics** • **Security and privacy~Web protocol security** • Security and privacy~Digital signatures • Security and privacy~Mobile platform security

INTRODUCTION

Nowadays with the evolution of technology, it is possible access all kind of services such as e-commerce, online banking, e-payment and many others using our personal devices, like smartphones.

Such services require user authentication to allow users to access their accounts or to perform web transactions. Almost all these services use passwords as the authentication and authorization mechanism.

However, problems for the users emerge by using passwords as a security mechanism. As an example, if the user wants a strong protection, he must memorize big passwords, that generally should not contain dictionary words, and are a combination of numbers, special characters, lowercase and uppercase letters [1]. It becomes hard for the user to remember such passwords.

However, even with strong passwords, the information is not completely secure, since over the years, criminals have learned to crack every kind of passwords [2].

Security and memorization are not the only problems associated with big passwords. It is important to notice that nowadays people are using smartphones for everything, so user-friendliness authentication mechanisms are important too [1], since mobile devices are intended for quick and frequent access. Strong passwords are not appropriate for use in mobile devices due to the length of time required for their input, which can lead to password disclosure.

These two problems need to be solved: improve (i) security and (ii) usability of authentication and authorization mechanisms.

To achieve a higher level of security there exists alternative authentication mechanisms apart from knowledge (passwords), such as possession factor and biometrics.

Authentication based on a possession factor, although it is more secure than knowledge factor, it's still not a perfect solution since the user needs to possess and carry an additional object such as a smart card or an USB Pen, in order to authenticate. And it is the opposite of usability.

Biometrics are the answer to these problems. Biometrics are described as automatic recognition of individuals through their unique physiological (fingerprint, face, iris, etc.) or behavioral (voice, gait, signature, etc.) attributes.

Biometrics offer several advantages over knowledge (passwords) since they are present on users all the time, can be inputted as quickly as a glance or a touch and they cannot be forgotten.

With biometrics the two problems mentioned are solved since that even if a smartphone is stolen, the information can only be accessed by the legitimate user, and to access information the legitimate user has no need to remember big and complicated texts, since it is only needed to present to the biometric system one of his biometrics attributes.\\

Most recent smartphones contain many useful sensors, such as GPS, cameras, touchscreens, fingerprint scanners, gyroscopes and microphones. Such sensors allow the use of biometrics as an authentication mechanism.

Additionally, biometrics are the best choice to study and implement, since mobile devices are typically dedicated to a single individual.

Objectives of the Work

The objective of the proposed work is to study the current technology regarding biometric systems and authentication protocols in literature to implement a second-factor-authentication framework as an alternative method for business applications available on the web, such as BankOnBox and eDocLink (both systems belong to Link Consulting company). The framework must use current mobile devices hardware capabilities to perform biometric authentication.

RELATED WORK

In this chapter are studied:

- Biometrics, which biometrics exist and which present the best results in terms of accuracy to recognize the correct person;
- Authentication protocols, to compare different biometric authentication protocols and implement the system using the best protocol.

Biometrics

Biometrics can be divided into sub-categories:

- Hand Region: fingerprint, palmprint, hand geometry, hand vein pattern and finger knuck print;
- Facial region: face, ear shape, teeth, tongue print;
- Ocular region: retina, iris, sclera vasculature;
- Medico-chemical: Body odor, DNA, heart sound, electro cardiogram;
- Behavioral: Keystroke dynamics, voice, signature, gait;

Fig. illustrates the accuracy of each biometric system currently [3]. The accuracy of a biometric system is the ability of the system to correctly recognize the user.

Due to current mobile devices hardware limitations most of these biometrics cannot be used as authentication mechanisms, while other, like gait, are not proper for authentication purposes [4].

Currently, the best biometrics to use on mobile devices for authentication are:

- **Fingerprint**, widely implemented in current smartphones, by a fingerprint sensor.
- **Voice**. Every smartphone comes with a built-in microphone.
- **Face**. Like voice, every smartphone comes with a built-in camera and the image quality of these cameras is constantly increasing.

Biometric modality	Accuracy level
Fingerprint	99.9%
Palmprint	> 95%
Hand geometry	> 95%
Vein pattern	99%
Finger knuckle print	Not available
Face	95%
Ear	> 95%
Tongue print	Not available
Iris	99.9%
Retina	99%
Sclera	Not available
Voice	> 90%
Keystroke dynamics	> 90%
Gait	> 90%
Signature	> 90%

Figure 1 - Biometrics' accuracy

Authentication Protocols

To integrate biometrics in the system to perform remote authentication, the main emerging protocol standards are Fast IDentity Online (FIDO) Alliance and Biometric Open Protocol Standard (BOPS) [5].

The core ideas of FIDO Alliance are (i) ease of use, (ii) security and privacy, and (iii) standardization.

FIDO Universal Authentication Framework (UAF) is a protocol that provides password-less and multifactor security for online services. The users can register their devices in the system by selecting a local authentication mechanism such as fingerprint, voice or face. The UAF protocol allows the service to select which mechanisms are presented to the user. Once registered, the users simply need to repeat the local authentication mechanism, chosen at registration phase, whenever they need to authenticate to the server.

Similar to FIDO, there exists BOPS developed by Hoyos Labs. It was recently published as a standard by IEEE. Both protocols authenticate locally (user device) and the result is then sent to the specific server. The main difference between FIDO and BOPS relies on biometric requirements, since BOPS specify several minimum requirements for the biometric recognition system, such as thresholds for equal error rate (EER), liveness detection, among others requirements, while FIDO is more abstract, in a way that there exist no requirements for the biometric recognition system.

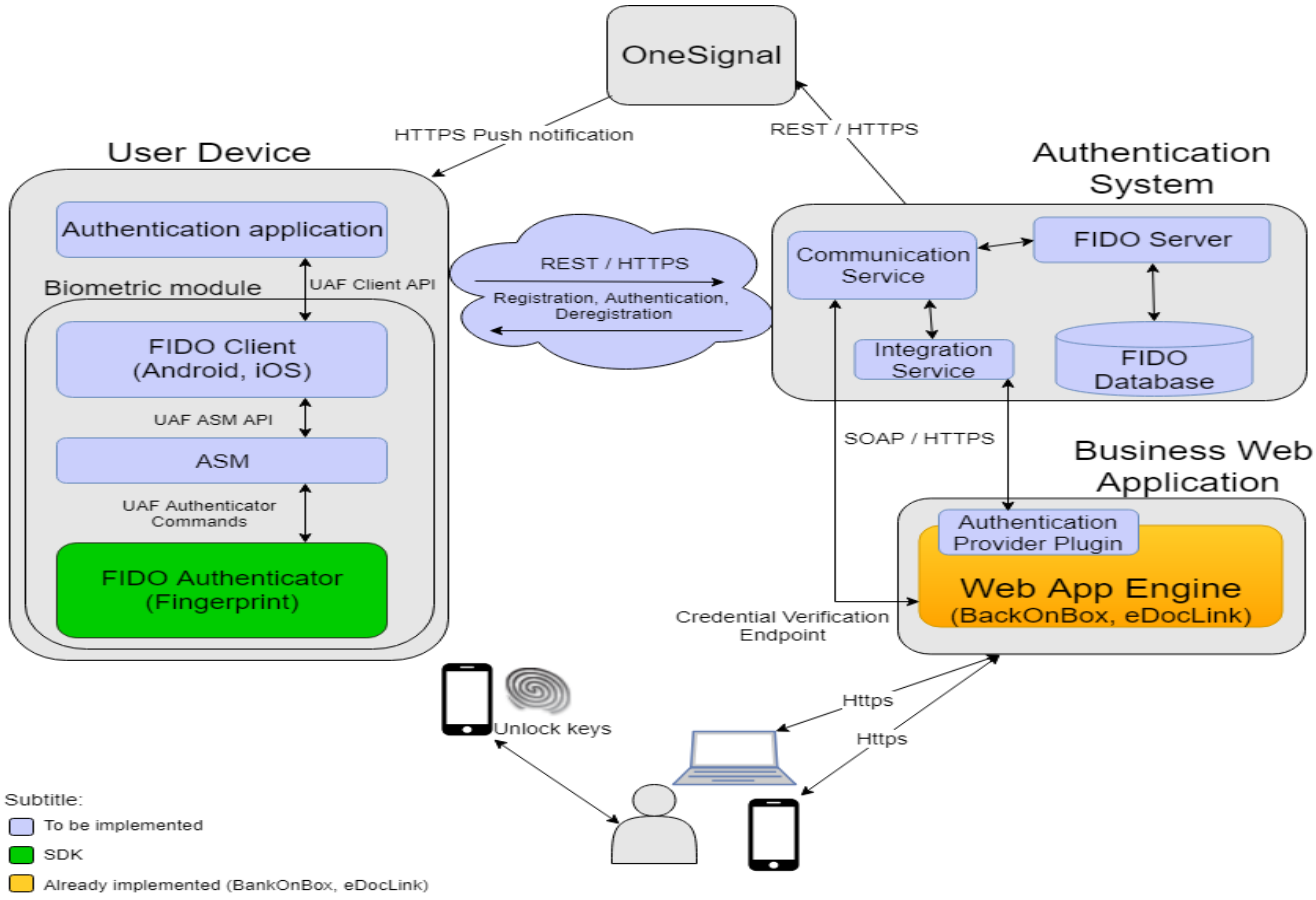

Figure 2 - BioAuth's Software Architecture

FIDO Alliance and BOPS are standard protocols for secure biometric authentication. However there more companies adopting FIDO Universal Authentication Framework (UAF) protocol than BOPS, making FIDO UAF protocol the best option to implement the system.

SOFTWARE ARCHITECTURE

The proposed BioAuth's software architecture (Fig. 1) will be based on FIDO UAF Protocol with fingerprint biometric authentication.

The system is composed by the following components:

1. **User Device** This module represents the user's mobile device (Android and iOS). For a regular user the mobile device is the way to perform the registration, authentication and deregistration in BioAuth system. The protocol is divided into 4 submodules:
   1.1. **Authentication Application** - user interface, where the user interacts with the application.
   1.2. **FIDO Client** - responsible for processing all the information received and information to be sent. It processes the information stored inside the mobile device.
   1.3. The **Authenticator-Specific Module (ASM)** is a software interface on top of FIDO Authenticators. The ASM gives a standardized way to the FIDO Client to detect and access the functionality of UAF authenticators and hides internal communication complexity from clients.
   1.4. **FIDO Authenticators** are the biometrics used for authentication into the system. In the current implementation only fingerprint authenticator is implemented.
2. The **Authentication System** processes registration, authentication and deregistration requests. It contains four submodules:
   2.1. **Communication Service** is the entry point of all communication with the Authentication System. Every message is first analyzed by the Communication Service that then forwards it to the FIDO Server.
   2.2. **Integration Service** is responsible for the integration between the Business Web Application and the Authentication Module. Messages from the Business Web Application are received by the Integration Service and retransmitted to the Communication Service to be analyzed. The responses are sent to the Business Web Application by the Integration Service too.
   2.3. **FIDO Server** is responsible for processing all the requests. It processes the registration,

authentication and deregistration requests and acts in conformity with the operation result. It communicates with the database to store and access user data.

2.4. **FIDO Database** is responsible to store information related to users registered in BioAuth system.

3. **Business Web Application** represents the system which the BioAuth is integrated.
   3.1. **Web App Engine** represents the business system BankOnBox, the online home banking system.
   3.2. **Authentication Provider Plugin** is responsible to handle the communication between the Authentication System and Business Web Application.
   3.3. **Credential Verification Endpoint** used by the Communication Service to verify the user credentials before proceeding to registration.
4. **OneSignal** service is used to send push notifications.

Data Flow

BioAuth's communication protocol is based on FIDO UAF protocol. The communication is divided in three steps: registration, authentication and deregistration.

Registration

Before the registration in the BioAuth system, there exists a pre-registration step that starts at BankOnBox (Business Web Application). It is BankOnBox internal policy to manage the client registrations on BioAuth system. If a user wants to register in BioAuth to perform second-factor-authentication using biometrics he needs to communicate that decision to BankOnBox bank operator. The bank operator will send a pre-registration request to the FIDO Server, meaning that the user is, from that moment, is able to register in BioAuth system.

The user will then access the mobile application and fill a form with his BankOnBox credentials, to verify if the user is valid and that a pre-registration already exists. It is verified by the FIDO Server. If the user is valid, the FIDO Server creates a "registration request" message that contains a random challenge that itself created.

The FIDO Client receives the "registration request" and starts the local user registration process. The user is prompted to use his fingerprint to create a cryptographic RSA key pair. The RSA private key is stored inside the mobile device and is used to create a signature of the challenge received. The FIDO Client sends a "registration response" message containing the signature along with the public key.

When the FIDO Server receives the "registration response" message, it verifies the signature using the client's public key and sends the result of the signature verification to the FIDO Client. After this process, if the verification was valid, the user is registered in BioAuth system and is able to perform second-factor-authentications using the fingerprint.

Authentication

The authentication starts with the user in his BankOnBox account submitting a transaction request.

The BankOnBox, using the Authentication Provider Plugin sends a message to the Authentication System, received by the Integration Service, informing that a transaction needs to be verified and the user needs to be authenticated.

The FIDO Server sends a request to OneSignal to be sent a push notification to the user's device. Inside with that request message goes an "authentication request" message, created by the FIDO Server, containing a random challenge, that the User Device will receive when it receives the push notification.

When the FIDO Client (User Device) receives the push notification, it extracts the challenge from the "authentication request" message. To sign the challenge the mobile device Operating System (OS) needs special user permissions to access the user's RSA private key, created in the registration phase. The permission to access and use the private key is granted by the user authenticating himself to the system using the fingerprint registered in registration phase. After a valid user fingerprint authentication, the FIDO Client signs the challenge and creates an "authentication response" message containing the signature that sends to the Authentication System.

The "authentication response" is verified by the FIDO Server.

After the authentication, the BankOnBox will send a request to the Authentication System to verify if the authentication was valid. If the authentication was valid, the transaction is performed.

Deregistration

Deregistration process is similar to registration. BankOnBox manages the user entries on BioAuth so a pre-deregistration phase must take place before the deregistration. The deregistration request is sent by the bank operator to the Authentication System. After this first step, the deregistration takes place.

Deregistration starts in the user's mobile device with the user pressing the "Deregister" button on the Authentication Application. The FIDO Client sends a message, containing the user identifier, to the Authentication System informing that a deregistration is requested.

The FIDO Server creates a "deregistration request" message, containing a random challenge.

The FIDO Client receives the "deregistration request", extracts the challenge, and signs the challenge. A "deregistration response" message, containing the signature, is sent to the Authentication System.

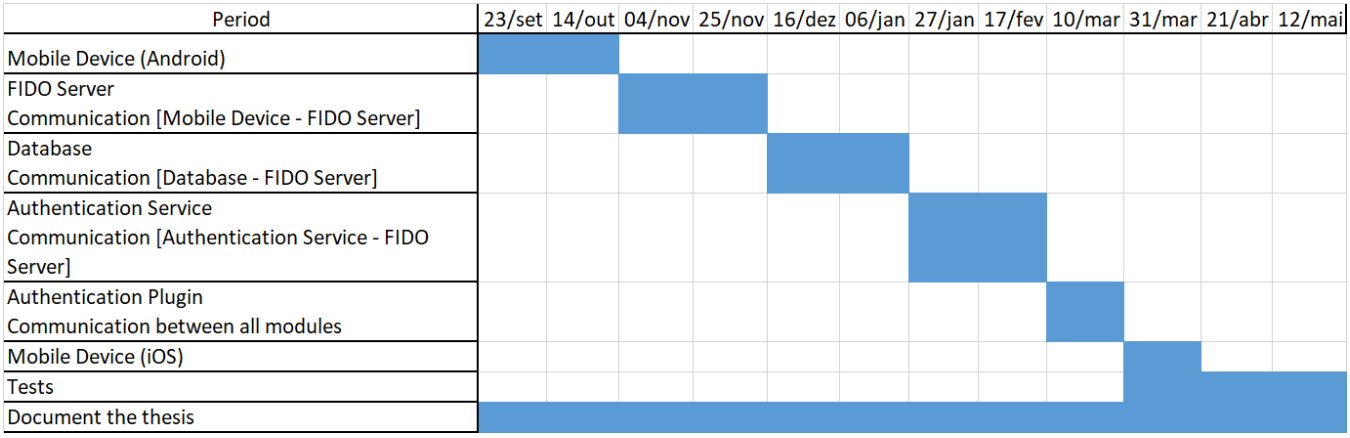

Figure 3 - BioAuth's schedule

The FIDO Server verifies the signature and if the result of verification is successful the user is set as invalid in the system (the user entry is never deleted from database). The verification result is sent to the FIDO Client.

If the verification result was successful, the FIDO Client permanently deletes the user RSA cryptographic key pair.

IMPLEMENTATION

Development Process

The system is implemented following a waterfall model.

Fig.3 shows the schedule for BioAuth system.

Development Environment

The system is implemented using Microsoft technology, for mobile and for server implementation.

Mobile device application is implemented using Xamarin Forms. Xamarin is a cross-platform software from Microsoft that makes programmer's life easier, since a single application can be built into multiple Operating Systems (OSs).

This way there is no need to learn another language, like Swift for iOS. In Xamarin Forms, using C# programming language, an application is written for both OSs, Android and iOS.

The Authentication System, where the FIDO Server is located, were implemented using ASP.NET Web API Technology. ASP.NET Web API is a framework for building web APIs on top of the .NET Framework.

User Device

The user device module is responsible for:

- Communication with the Authentication System;
- The creation of the RSA cryptographic key pair;
- RSA cryptographic key pair storage;
- Use of fingerprint to authorize the use of the private key.

These tasks are performed using APIs for specific OS, Android and iOS.

Microsoft provides wrapper classes from native APIs to use native methods from each OS.

Communication with the Authentication System

The communication between the User Device and the Authentication System uses REST web services. The messages are sent as JSON objects, as specified in FIDO UAF Specification documentation.

The communication on the User Device is handled by the FIDO Client. To implement the communication it is used a RESTful library, called Refit.

With Refit is just needed to specify the communication endpoints in an interface class and call these methods passing the objects to be sent. Refit automatically serializes these objects to JSON objects.

RSA cryptographic key pair creation, key pair storage and fingerprint

These tasks were implemented in the specific OS projects, since are used native APIs.

In Android the RSA key pair is created using the "KeyPairGenerator" class. The key pair attributes are passed as methods.

In iOS is used the "SecKey" class, using the method "CreateRandomKey" that receives a dictionary containing the key pair attributes.

The RSA attributes used on both OS are:

- 2048-bit key length
- SHA-256 message digest
- PKCS#1 padding

The key pair is stored inside secure hardware for both OS. On Android is stored inside Secure Element (SE) or Trusted-Execution-Environment (TEE), depending on the hardware specifications. On iOS is stored inside Secure Enclave (SE), a co-processor.

To manage the key storage are used Keystorage Providers for each OS. On Android is used the Android Keystore, on iOS is used Keychain.

The private keys can only be accessed and used by user authorization. The authorization is given by the user by proving his identity using his fingerprint.

In this project only fingerprint authenticator is implemented since Android and Apple do not provide APIs to face and voice authentication to store and access keys.

In Android fingerprint authentication is implemented using Fingerprint API while in iOS is using TouchID API.

In Android fingerprint authentication is required by the application developer by using the method "CreateConfirmDeviceCredentialIntent()", while on iOS is the OS that automatically asks for the fingerprint each time the OS needs to access the private key.

Authentication System

This module is responsible for:

- Communication with external modules (User Device, Business Web Application and OneSignal)
- Signature verification
- Storage of user's information, like user identifier, and RSA public key.

Communication with external modules

The Authentication System communicates with external modules by two ways, the Communication Service and the Integration Provider.

The Communication Service is implemented using Microsoft ASP .NET Web APIs. It communicates with the User Device via HTTPS using RESTful web services. Each method of the Communication Service class has an attribute specifying the endpoint.

The Integration Service is responsible for communication between the Authentication System and the Business Authentication Application. The communication may use RESTful or SOAP Web Services. In this system the Integration Service communicates with BankOnBOx system using SOAP. The Integration Service is a SOAP Web Service created using Windows Communication Foundation (WCF) framework (a framework to build service-oriented applications) from .NET.

Communication with OneSignal is performed via HTTP using REST web services. The communication is implemented using OneSignal API, and the construction of requests to send to OneSignal is performed using "OneSignal.CSharp", that facilitates the creation of all message required fields.

Signature Verification

Signature verification is performed by the FIDO Server module.

It was used the BouncyCastle library to perform signature verification using the user public key. Since Android and iOS OSs send the user public keys with different encoding the method to reconstruct the public key and verify the signature is different for each signature sent by each OS.

The public key is sent by Android encoded as ASN1-X.509-DER while iOS encode as ASN1.PKCS#1-DER.

To recreate the public key from Android devices is used the class "PublicKeyFactory" and then the class "RSACng" to verify the signature.

With iOS the public key is recreated using the class "PKCS1Asn1SequenceToPublicKey" and the signature is verified using the class "SignerUtilities".

Storage of user's information

To integrate the FIDO Server with a database it is used Entity Framework. Entity Framework is an object-relational mapper that let me work with the database using .NET objects. I followed a Code-First approach where I create and manage tables only using code.

Business Web Application

The Business Authentication Application represents a system like an home banking system or a file management system. BioAuth current implementation is only integrated with BankOnBox system, an online home banking system.

existing system.

Integration Provider Plugin

BankOnBox uses the concept of providers to authenticate with external systems. I developed my own provider to authenticate with BioAuth framework.

Authentication Provider Plugin handles the requests (pre-registration, authentication, authentication verification and pre-deregistration) from Web Business Application to the Authentication System.

The current implementation uses SOAP Web Services, since all BankOnBox communication structure uses WCF with SOAP. It accesses the Integration Service (Authentication System) with SOAP web services (consumes) using the WCF framework.

EVALUATION

The following types of tests were performed to the system:

- Usability tests;
- Interoperability tests;
- Security tests;
- Performance tests;

Usability Tests

Usability tests were performed without the BankOnBox component fully integrated (Business Web Application).

It was missing the webpage from BankOnBox, where the user performs the transactions, and the webpage where the bank operator sends the pre-registration and pre-

deregistration requests. Those messages were sent by me using SoapUI, to simulate BankOnBox behavior.

The tests were carried out by me and tested on eight users (seven using Android devices, one using an iOS device).

To test the system usability, I asked the users to perform the registration, authentication and deregistration tasks by themselves using their mobile devices with the BioAuth application installed.

Three users did not know what to do when the fingerprint authorization screen appeared, while registering, and I explained it was to use the fingerprint on mobile device. After this doubt, the users had no more doubts and when the fingerprint authentication screen appeared at authentication and deregistration process all users successfully used the fingerprint.

At the end of test, I asked the personal opinion of the experience. All users appreciated and found the application behavior intuitive, however all users pointed that the graphical user interface should be improved.

Usability Tests

Since the system was implemented following a waterfall model, it was crucial to perform interoperability tests to ensure that each module would communicate with other modules without any compatibility issues.

The following interoperability tests were performed:

- Verifying that the User Device was sending messages to the Authentication System using RESTful Web Services communication.
- Verifying that the Authentication System and the Business Web Application were successfully sending messages to each other.

The result was that the integration of all modules was easy and no errors occurred.

Security Tests

The security tests, in BioAuth system, could be divided in three parts:

- The security of cryptographic key storage in mobile devices;
- The security of the communication and;
- The security of the FIDO Server;

Cryptographic Key Storage Security

While investigating about ways of testing the security of keys storage for iOS and Android [6] [7], what I found that the only way to retrieve information from key storage providers is by rooting (Android) or jailbreaking (iOS) the mobile device.

The only thing that could be done in BioAuth system is to prevent the application to be installed on a rooted or jaibroken device, by adding extra code to the application. Although new ways of rooting are always being used by malicious users it still is something to add in the code in the future.

Communication Security

The entire system communication is based on HTTPS protocol, so the messages are cyphered.

Although I couldn't do more, to my knowledge after some online investigation, I tried to sniff the message content exchanged between the User Device and the Authentication System using Wireshark tool. Wireshark is a network packet analyzer that captures network packets and displays the packets data as detailed as possible.

I could see the HTTP messages being exchanged, however I could not extract any useful data from the messages since the data was ciphered. The verification of message exchange between the Authentication System and the Business Web Application were performed too and the result was the same, no useful data could be read from human eyes.

However, I just did a security verification, since i just verified that the content of messages was unreadable. I did not attack the message content, to decipher the data, neither performed any automated test with any testing tool.

FIDO Server Security

To test FIDO Server security, I investigated online for an automated tool to perform automated tests. However, I could not find any free automated test tool. Even so, I performed a static analysis of the FIDO Server, by analyzing the code and functionality, based on a STRIDE threat model [8].

- **Spoofing Identity** - since the authentication is performed through biometrics, this threat is less likely to occur than in a system with password, since counterfeit fingerprint is more difficult to perform [4] (depending on the password). FIDO Specification leaves the biometric implementation to the developer, but they also mention this threat as a problem that cannot be avoided but is mitigated by the use of biometrics.
- **Tampering with Data** - communication is performed by HTTPS so information cannot be read by human eyes. But, a malicious user may randomly manipulate the information of the message to create entropy in the system. To mitigate this problem, there is a field in any message sent by the server named "ServerData", specified in FIDO Specification, which is a random value, created by the FIDO Server and that the client can't change and must send it as it was received to the client. This value is used to identify the communication connection and to identify if the data in the message was randomly manipulated, since it is encrypted (HTTPS). If so, the FIDO Server discards that message.

- **Repudiation** - In this system all transactions are logged and all messages contain a signature, signed by the user private key, that only the user has access to, by using his fingerprint to unlock the private key usage. This way non-repudiation is granted in this system.
- **Information Disclosure** - In BioAuth's system all message exchange is performed through HTTPS so all HTTP message data, even the headers, are encrypted. The attacker cannot access any privilege information about the user.
- **Denial of Service** - The FIDO Server verifies the HTTP header of every message received and discard all messages that comes with an incorrect "Content-Type" HTTP header, without even processing the message. This measure is not the most secure and does not prevent completely this threat, but at least prevents some malicious messages to be accepted and slow the system. According to [9] there are not many effective ways to prevent Denial of Service (DoS) attacks. However, some additional methods can be implement in the future, such as deploy a firewall or third-party services to prevent some DoS attacks [10].
- **Elevation of Privilege** - In BioAuth system all users have the same account type, so there is no elevation of privilege threat possible since there are no special permissions for any particular users.

Performance Tests

Performance tests were carried out to check how the Authentication System behaves and performs under a different number of concurrent virtual users performing transactions over a certain period using SoapUI tool.

The following variables were used to simulate different user behaviors:

- Number of virtual users (2, 10, 100 and 1000);
- Test time duration (seconds) (always 60 seconds);
- Time delay between each message is sent (milliseconds) (1000 ms and 100 ms);

The following statistics were collected:

- Total number of messages sent;
- The number of bytes processed;
- Requests per second;
- Error rate (percentage of requests that failed);

The results demonstrate that by increasing the number of users for a constant time delay between messages, the error rate increases too. And for the same number of users, the number of error increases when the time delay between messages is decreased from 1000ms to 100ms.

The system can handle and average of 6.69 requests per second without any errors and 25.53 with an error rate of 1%.

In a real-life scenario the system hardware specification is better than the one used to perform this test which can result in a better performance.

CONCLUSION

BioAuth is a framework developed for second factor authentication, such as authorize online banking transaction. The authentication is performed using biometrics. At the moment only fingerprint biometric is used.

The framework is composed by an application for mobile devices and a server. The mobile application is used to perform second factor authentications using biometrics. It was implemented using Xamarin Forms. The server was implemented using ASP.NET Web API technology.

The communication between the mobile application and the server are based on FIDO UAF Specification.

User authentication is performed by mean of asymmetric cryptographic keys. The communication is performed using REST Web Services, through HTTS. The BioAuth framework is integrated with BankOnBox, an online home banking system developed by Link Consulting company.

The framework communicates with BankOnBox using SOAP Web Services, through HTTS. The framework was evaluated in terms of usability, performance, interoperability and security. The evaluation results were satisfactory, since no negative result was obtained.

The technical contributions of this project were:

- The use FIDO UAF protocol for remote authentication using biometrics;
- Using Xamarin Forms to implement the mobile application for Android and iOS;
- Using native secure APIs, from Xamarin, to implement biometric authentication and cryptography operations on Android and iOS;
- Implementation of a server that securely communicates with outside modules using REST and SOAP web services over HTTPS;
- Implementation of a server that stores the client's cryptographic public key and verifies signatures from the client using that same cryptographic public key with BouncyCastle cryptography library;

This framework comes in an era where mobile devices are being used more than laptops or desktops.

It is a different idea, that allows user to use biometrics, and innovative technology introduced in recent smartphones that provide an high level of security too.

Biometrics makes this system different from other authentication system, since alternative systems authentication is based on passwords or an external authentication device, which are authentication methods easily to be forgotten.

System Limitations and Future Work

Currently, BioAuth is only integrated with BankOnBox. In the future BioAuth can integrated other business systems.

The current framework only uses fingerprint as biometric authentication mechanism. In the future the system can be improved by adding new biometric authentication mechanisms, such as face and voice recognition.

BioAuth is only used as a second factor authentication mechanism on BankOnBox. Another improvement is t use BioAuth as first factor authentication mechanism, to perform logins, for example. Like was mentioned by the tested users in usability tests, the BioAuth's mobile application GUI can be improved too in the future. To improve the security between the User Device and the Authentication Service, the communication protocol can be improved, by adding an extra layer of security, like the one used with BankOnBox (WS-Security using x509 certificates).

REFERENCES

1. L. M. Mayron, "Biometric Authentication on Mobile Devices," IEEE Security & Privacy, vol. 13, no. 3, pp. 70–73, 2015.
2. B. I. Mikhail Gofman, S. Mitra, T.-h. Kevin Cheng, and N. T. Smith, "Fusing information from multiple biometric traits enhances authentication in mobile devices," COMMUNICATIONS OF THE ACM, vol. 59, no. 59, 2016.
3. J. A. Unar, W. C. Seng, and A. Abbasi, "A review of biometric technology along with trends and prospects," Pattern Recognition, 2014.
4. C. Sanchez Avila, J. Guerra Casanova, F. Ballesteros, L. J. Martin Garcia, M. F. Arriaga Gomez, D. de Santos Sierra, and G. Bailador del Pozo, "State of the art of mobile biometrics, liveness and non-coercion detection," 2013.
5. M. Stokkenes, R. Raghavendra, and C. Busch, "Biometric authentication protocols on smartphones: An overview," in Proceedings of the 9th International Conference on Security of Information and Networks, Newark, NJ, USA, July 20-22, 2016, 2016, pp. 136–140.
6. B. Muller and S. Schleier, Mobile Security Testing Guide. OWASP Mobile Team, 2017.
7. T. Cooijmans, J. de Ruiter, and E. Poll, "Analysis of secure key storage solutions on android," in Proceedings of the 4th ACM Workshop on Security and Privacy in Smartphones & Mobile Devices, SPSM@CCS 2014, Scottsdale, AZ, USA, November 03 - 07, 2014, 2014, pp. 11–20.
8. OWASP Team. 2017. Threat risk modeling. (13 July 2017). Retrieved March 10, 2018 from https://www.owasp.org/index.php/Threat Risk Modeling#STRIDE
9. S. Farraposo, L. Gallon, and P. Owezarski, Eds., Network Security and DoS Attacks, 2005.
10. The Windows Club. Denial of Service Attack: What it is and how to prevent it. Retrieved March 10, 2018 from http://www.thewindowsclub.com/dos-denial-of-service-attack