Analysis of a Mobile Banking Application

Fernando Pannullo
Computer Engineering
Aarhus University
Stud. 202102261 auID 703559

Abstract— **This report has the function of analyzing an android app in several security aspects like networking with external environments and the management of local data internally and through other apps, over some attention also for the privacy. The privacy is intended not only for security information like authentication and authorization but focused on data that anyone can consider private.**

A measurement of privacy is more difficult to grade, for the reason that people can vary their perception even for reasoned choice or not. We can establish some metrics, what information leaked or voluntarily went out from the banking service and who can receive the personal information.

The application examined born in a financial-banking area and more specifically with a massive use of technology environment instead of a traditional human personnel for offering the service.

I. INTRODUCTION

The banking app that we consider to analyze refers to a Danish on-line bank. Through what has been said, the service that offer is a financial-banking type, therefore, demands an high level of security, because an underestimation, can cause severe inconvenience; the loss of money of the user in some case, can be catastrophic for both, user and the company, proportionally it have to manage lawsuits and compensation to afford, loss of credibility that can lead to company failure.

The loss of sensible data or decreasing privacy is also a drawback to afford for the same reasons. After a data-breach the company is obliged to notify a supervisory authority competent, under rules of European GDPR laws. A lacking or intentional marketing of the privacy may cause the user changing bank and a potential new customer to consider alternatives if discovered and the customers feels that they have not received the appropriate information. Some or all of this information is usually stated at the disclaimer of every contract we sign. Maybe marketing can offer a customer a comfortable condition of the service, but the company generally has all the power to decide the trade-off, the value of this data for the counterparts.

This security report consists in establishing firstly a *Threat model* to state formally the security environment of the service in a wide-spread view. This is useful to set a boundary where the analysis has to cover, for better focusing on what we are dealing with, to give a direction for all people involved in the engineering security of some entity.

The main fields that these previously hint boundaries cover, is the discovering of the possible adversary, in other words to identify from *who*; that can be a person, a group of persons with the same intention or having clear associations and maybe some hidden associations with each other. The reasons that drive an attack can activate a strong motivation or less, consecutively, the time to spend and budget for accomplishing.

Establishing this, it is interesting to discover another consideration formally named the *Attack Surface*, that is a description of *what* entity, that belongs to the entire structure, it has to consider exposed to threat. With this assumption we create frameworks or single parts, that can be physical or software, to take care when we consider the entire security of the system. There is a section that describes in detail the type of attack in a technical view to ease after the linking from a generalized variety to this environment.

To conclude we can define the *policies*, that describe the properties that the approaching security, have to be included. Listing the most valuable properties helps to develop a more practical and efficient way to build a security evaluation, instead of remaining on generalized recognition of security. Every property needs time, effort and resources to cover the possible threat, so, excluding some features not necessarily essential from others which have a big importance, can be optimized and improve the estimate.

After that the report takes an orientation to focusing on the application itself. Briefly describes the contest of the application and some helpful information around it and having read the necessary, the report deep dive deeply into the security view of the application software and relative connection that can take place with other internal applications, the information and technical data that store locally or derived from the server. Then we consider a network security view that is focused on the linking of the user-interface and internal logic to the user-data that can be outside the app. As network security views outside apps have the meaning of exchanging information with a server via Internet. Finally the attention touches on consideration of privacy in general and specifically, what type of privacy that application can offer. As mentioned before, it is difficult to state a sure judgement in this area.

II. THREAT MODEL AND GENERAL CONSIDERATION

A. General consideration

The application in examination has the name that corresponds to the company-bank name as *Lunar* more specifically a Danish bank. The core business takes place mostly in the mobile-application where a user can use the service, as mentioned, without any physical office. A successful eventual attack, of type *Denial of Service* (DoS) that usually takes minor effort compared to the other, can cause the most jamming of the business.

This not only involves merely the customer satisfaction that can get up to a loss of customers, but have severe impact on other sides. Some users can demonstrate to have suffered a substantial unease and decide to prosecute for obtaining fair compensation. Traditional banks mitigate this risk, having a physical office and deposit, that can manage the basic transitions with minimal use of computer systems.

This premise is important to give a general idea of what we are approaching. In general the banking requires a high level of security, but in this case, the building of the connection through the service and the user is always focused on mobile application development and we can say with reserve, that we have less attack surface to consider.

B. Threat Model

A formal definition of this initial stage for engineering the security aspect is: "Threat modeling works to identify, communicate, and understand threats and mitigations within the context of protecting something of value."[1]. In this way we can start to build the foundations, where we try to identify from who we have to to be afraid and contrast. As mentioned in the introduction, there are some considerations to take, not only from *who* but also *why*, following this last, bring us to consider a correlation to *how* base an attack, like a flow of essential information. The information for responding to this requests, have to be obtained from trusted standards that have the important role as reference, this are: National Institute of Standards and Technology as *NIST*, internet society as *IOC* and The International Telecommunication Union *ITU*.

Is important to generalize before going through this specific case, for the subject in examination. When we think big, it is what is called *Spooks*, as usually a group of people with strong interests at a more sophisticated level. For that they can have a lot of support in terms of power, that is intended as a combination of technical resources, funds and time. From this they can operate hiding and mostly can reach a more sophisticated level as mesmerizing. They can be a government and their operative apparatus for surveillance, state-sponsored operations, espionage, disinformation and psychological operations. Going beyond considering the subjects, there are the *Crooks* where the ability and the knowledge of the single individuals and its stories can leverage the success of their intent, usually are all associated to cyber-crime. Lastly there is a category of *Geeks*, as security researchers (academia and industry), activists, Non-Governmental Organizations, that have the same characteristic in terms of hazards of the *Crooks* but with different motivation.

In the case that we are approaching to consider, the audience that can be interested in breaking the system to obtain something, is quite large, as a bank that manages things that are universally recognized as valuable, money first and other financial assets secondary. As an important structure that manages high valuable assets, it has to show an absolute visible appearance of solidity and safety, requiring a lot of investments. For this reasons it is difficult to enter this type of business, so we can consider a niche with a high potential earnings. This may drive some adversaries to consider spending a lot of resources to have success.

The subjects in this *Threat Model* are identified into customers that can consider that have a little view of the *inside*, in this case the functionality of the application over the services offered and have the possibility to check some trivial feelings of possible imperfections. In the same group are the cyber-criminal that intend to break the system with a "professional" approach. The firsts generally have motivations only for gaining money illegally and the second add the possibility to improve their recognized professionalism from their illegal-business area.

In another group we can insert insiders, like an employee, manager (or ex for both) or someone who have a particular knowledge of the system in one or some area, that in some instances can make the difference for succeeding an attack. In another edge is possible to use their trusted position to elevate their authorization level and hide the traces. Moreover, technology advances, because it can reach some levels or units of the organization from the inside, where the security has a little lossy *wrapping* due to a normal work relationship confidence. In this case generally the motivation is the same as the customer adding some possible addiction to take a revenge for a wrong suffered.

In the last group, specifically a group of groups of people like competitors prone to discredit the opponent and government apparatus that may want to take possession of sensible data in an illegal way and use it, in an uncontrolled smoky way.

C. Attack Surface

This section of analyzing involves more practical-technical aspects and not only the human one. A definition " An attack surface consists of the reachable and exploitable vulnerabilities in a system" [2] can resume all of what we have anticipated in the introduction, these surfaces are synthesized into a possible area of major interest to detect vulnerabilities. Starting from the network one, that includes not only the external communication from the device to the server but also the internal enterprise network. We have the software surface, one that is the most invisible part and can include bugs, wrong implementation of security, old libraries and functionalities considered dangerous to maintain in the software framework. There is a last surface, the human one, where involves the psychological capability to take advantage for the purpose and as we stated before use thrusting for creating a point of defence bypassing.

When we consider this *Attack Surface*, it must consider all of the possible attacks that can take place in this surfaces. In order to create a better sophisticated and effective engineering of security, giving this surfaces we can create an *Attack Trees* for collocate the final objective of potential malicious attacker in the root and the branches and subnodes, that can be reached and breached with various attack type or an combination. On the *leaf* of this structure, we can evaluate the practical attacks that may occur, accordingly to the type of the surface.

The attacks essentially, can be categorized into two main types. One section is something that admit the adversary to hide more effectively, because it consists in "hearing" the transmission without modifying anything, and taking part of the group of *Passive Attacks*. This is what is called eavesdropping and the objective as we said, is to catch in clear the information or messages that can be encrypted or more easily not encrypted. Another practice is to apply a traffic analysis, usually, when the traffic is encrypted and the challenge require some elaborate approach; the opponent is interested to find a "pattern" in the time domain, overhead data, type of format of some messages-packets for obtain more chance of success by combining clues to a sophisticated strategy.

In the other part, as technically opposite practice, we can find the Active Attacks, where the exploit can involve the creation, deletion, modification of the data partially or fully. In certain conditions they are equally dangerous, the important thing is to detect the type in time to act a countermeasure. The *masquerade* is a practice that, as the name advises, modifies the appearances of the attacker, that can be assumed with a combination of some active attacks. The impersonation can be scaled through privileges or types of user, enabling some active features for changing the normal exchange of data or identification-authorization. *Replay* attack types have the ability to use the message, after detecting some lacks of implementation of *nonces* or timestamp for example, to re-transmit having a recognized value in the system like the original one, with a little effort. *Modification* is quite similar to the *Replay*, but has a difference in more dangerous attitude, that is, the content is modified to the attacker's needs. The *Denial of Service* or *DoS* generally require less effort to accomplish. Essentially the main purpose is to destroy or limit the service targeted. Doing this can create several discomforts in terms of performance, for creating fake computational loads to manage or tampering the network.

In this application the *Attach surface* is concentrated basically into the IT environment and his particular parts for that are composed, with the conformation of the system talked before, we don't have any interaction with the public. Considering this, the software takes part of the *surface*, having some potential vulnerability to the web service, as the point of contact for every person and the service offered. Also a point of contact to the clientele is the mail service, arriving to consider some interaction that can be succeeded typically from sector operators as data exchange over *API*.

Firstly the code that exposes the services has to be architectured in order to be *audited* from internal and external technicians, also, in the process of developing it has to be properly tested for every section or framework that composes a service. After that a *penetration test* is a solution for having a global view and a practical feedback of some bugs or leaks when the system is complete.

Having this underneath, is useful to restrict the interface to insert values in controlled manner, with character and word filter to any entries in order to prevent any malicious *injection*.[3] Another important check is over the digital documents, that have to be scanned and removed from some malicious code in the metadata of the file.[4] In another aspect the transfers of sensitive data or secondary personal information, through sector operator, have to be selected in order to avoid passing of unnecessary information to sector operators.

Under a magnifying glass goes the socket, ports, firewall that provide access considering the port and the relative services attached. The libraries, operative system and program that manage this lasts and web service, have to be upgraded when required, in order to avoid unexpected attack, in few words a continue learning in security patches and apply. In this case overall, it is a must to consider to control the data also behind the firewall, where the service is led to facilitate the agility of the service tasks and improve productivity. Some of that can be some group of functions that can be override the internal security for technical reasons and for prosecution with the task. These features have to be authorized by a strict number of high-level employees ,also, the data treatment has to be hierarchically organized in order to select every security level linked to an appropriate authorization level in order to accomplish the *least privileges* fashion. In order to limit a possible *DoS* with some unexpected loads or natural disaster, implementing redundancy on hardware is mandatory in these critical enterprises. This can include the computational power but also the data itself.

Another important aspect is the network *surface*, that is to be considered an enterprise network, a wide area or lastly Internet. An enterprise network has to be secured internally with a monitoring of the device that can be attached to an internal network. Physically via ethernet a solution, is to relay only a static IP address and excluding DHCP easy IP assignment for the network, but it can be quite tedious to ask a configure to every session,so maybe, in some environments, it is sufficient to connect with a valid authentication and authorization. For wireless connection for protecting the enterprise and wide area can be a solution in addition to the classic method mentioned before, with a certificate requirement that require only a one-time installing in the devices and prevent some *brute-force* attack from the external, where the time available for succeed can be huge, over a not controlled area. Encryption is a solution to avoid the tampering of the messages or communication in general, this has to be implemented mostly when the traffic goes outside the internal controlled area. The connections and traffic from the Internet area have to be monitored in order to prevent exploits from protocols that leverage a *DoS*, and that is the reason why only some servers can communicate with this huge area, specifically having a lot in common with software precautions. Some part of the Internet can be made inaccessible for prevent and limit the use of unnecessary enterprise access, at the huge amount of information (as websites or services) that the Internet offers. The traffic can be monitored manually by a competent technician or helped with artificial intelligence, both can combine some suspicious traces to prevent not allowed hiding activities. Finally the wired network and devices connected has to be secured also to avoid tampering, with redundancies in some case like alarms and video-surveillance for offices and majorly in the data-center, hiding the cables and devices in secured places.

Lastly is considered, that human factor that has to be considered as employees or ex-employees, trusted insiders, that can behave intentionally bad and for some unconscious errors and social engineering. Some rule of rotation of the password, or expiring the passwords can mitigate the problems concerning employees combined with a strong transcription of *session-logs*. This avoids the knowledge of somebody else's password and using it over a long time. For ex-employee or changing of level of authorization this procedure has to be immediate and confirmed from one individual or more. A further countermeasure is to periodically change the routine or zone of competence for certain insiders in some critical areas ,doing this, can also limit human error about an evaluation of the security situation. The routines can induce to lower some attention to some new or hiding fall over various fields, from the software security to a physical security surveillance. For what concern the social-engineering the communication with users and partners has to be formal and strictly recognizable, especially for sensitive data in order to avoid *phishing* attacks. Imposing a choice of a strong password also, can help to mitigate a possible uptake combined with a registration of the connection-session data, with this it is possible to have some tracing in case of investigation.

D. Policies

In this section is describing the main corporate security goals and have to consider the specific venture feature in union with service offered. In other words, is where the enterprise tends to allocate their resources, in order to accomplish some milestones, that after is transformed into features to develop and implement in various layers. These aims are resumed into formal words that contain the concept behind.

1) Integrity: That is, as anticipated before, is essential for this type of activity. The enterprise must have and show an indisputable trustworthiness, so that, all the activity has to be guaranteed to be safe in every condition in principle by financial activities. Even if someone made an attempt to create an attack, this has to be very difficult to succeed, an extreme effort behind and even succeed, with a small return.

2) Confidentiality: Is a feature that takes some parts of the *Integrity* because is something that also trustworthy. The financial information has not to be public at all, for the reason that there can be a competitive economical interest from customers and outsiders in general. Privacy also, as personal information has to be protected to avoid some social-engineering and prevent some more severe attack.

3) Authenticity: Something that the user doesn't have a direct perception of, specifying, is the property that some customer or partner has to be assured to have an interaction to the wanted structure. This is not to be impersonated easily from another intermediate unauthorized point. So, there is need a strong link between the user and the service provider. And the credentials have to be in two directions from the user that wants to be authenticated and vice-versa.

4) Availability: Have a role underneath the system, when projected, the entire system has to be guaranteed that it can bear critical conditions after unfortunate events and intentional *DoS*. The financial services don't have minute time constraints, but time without this service can cause some collateral damage in terms of company image and directly for the clients after repetitions and long inefficiency.

5) Non-repudiation: Something that usually is taken for granted, that is a guarantee of the authenticity of the transaction in money (or other financial order), and after that the operation cannot be withdrawn from any of single parties. At maximum there is a possibility of mutually agreeing of both parties for modification or repair.

6) Accountability: Take a little part of *Authenticity* and provide an assurance that every transaction or action has to be accounted to someone that has the authorization to do so. The parts that can be accounted for are all that contribute to a single transaction, and every time it is needed to retrace some steps of one transaction, it is mandatory to have a tracing of the parties involved.

III. Practical Analysis

As mentioned before, it is analyzed in several technical aspects as software security,that goes deep on the software structure of the app, describing all the steps involved to view the code in clear and feature analysing with the demonstrations over the use of dedicated applications. Similarly, for the *Network security* is made an attempt of MITM as *Man In the Middle* attack, describe in what consists and the results including some correlated consideration about the *Authentication*. Finally is considered to have a consideration to the *Privacy* in some aspects, as the possible use of our personal information and ethics for managing the information.

A. Software security

The process of analyzing the software of Lunar app starts from acquiring the packaging of the app, in this case *.apk* package from Android SO. The package is extracted via *Apk Extractor* that save the package into the phone for allow it to transfer easily into the PC. After that is used *jdax-gui 1.2.0* to show the JAVA code inside of the app ordered into a tree sequential numbered packaging, but no original name is applied. Inside packaging we find classes and interfaces. For have more effort on analysis is used *mobSF* program that emphasize potential lacks of security. This program take reference to the indication of software security from CWE *Common Weakness Enumeration* [5], the *Common Vulnerability Scoring System* CVSS "that is an open framework for communicating the characteristics and severity of software vulnerabilities" [6] and OWASP as *The Open Web Application Security Project* [7]. Research in the code is made and globally, there is some security implementation that can be improved, from the choice of encrypting tools and one regarding a security fashion in the logic of the application that is good practice to take care.

The first overlook at app, shows an use of a weak cryptography algorithm in some class, that is one of the most severe issues found. In specific, the app uses in one class the encryption RSA with block cipher mode ECB PKCS# v1.5 (where PKCS# is a way for standard syntax for storing private key information). This algorithms have the demonstrated issue, that can be make clear an encrypted message over an *Padding Oracle Attack* similar to CBC implementation, related to *Bleichenbacher's CCA attack on PKCS# v1.5* [8], having this it can be a strong threat for password recovery in the worst case and privacy leaks in the light severe cases. A padding oracle is a function of an application which decrypts encrypted data provided by the client, e.g. internal session state stored on the client, and leaks the state of the validity of the padding after decryption.[9] Attempt to break the padding of the encrypted messages and analyze the returned error messages from the server, for a reconstruction of the message in clear after iterative attempt. Also there is a problem of *malleability* of the message as by definition a possibility to modify the content of the message without broken the cipher-text, so more clearly, acceptable from the receiver after a successful decryption. A solution can be the use of RSA V2.x that is considered not broken and a strong hash algorithm.

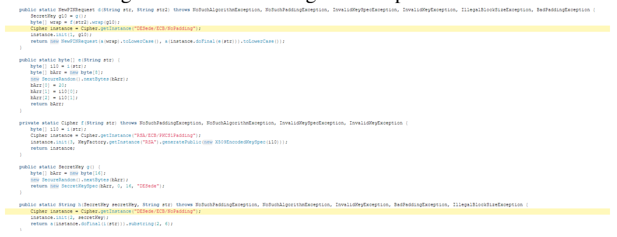


Fig. 1. Class including RSA implementation.

```
public static NewPINRequest d(String str, String str2) throws NoSuchAlgorithmException, NoSuchPaddingException, InvalidKeySpecException, InvalidKeyException, IllegalBlockSizeException, BadPaddingException {
    SecretKey g10 = g();
    byte[] wrap = f(str2).wrap(g10);
    Cipher instance = Cipher.getInstance("DESede/ECB/NoPadding");
    instance.init(1, g10);
    return new NewPINRequest(a(wrap).toLowerCase(), a(instance.doFinal(e(str))).toLowerCase());
}

public static byte[] e(String str) {
    byte[] i10 = i(str);
    byte[] bArr = new byte[8];
    new SecureRandom().nextBytes(bArr);
    bArr[0] = 20;
    bArr[1] = i10[0];
    bArr[2] = i10[1];
    return bArr;
}

private static Cipher f(String str) throws NoSuchPaddingException, NoSuchAlgorithmException, InvalidKeySpecException, InvalidKeyException {
    byte[] i10 = i(str);
    Cipher instance = Cipher.getInstance("RSA/ECB/PKCS1Padding");
    instance.init(3, KeyFactory.getInstance("RSA").generatePublic(new X509EncodedKeySpec(i10)));
    return instance;
}

public static SecretKey g() {
    byte[] bArr = new byte[16];
    new SecureRandom().nextBytes(bArr);
    return new SecretKeySpec(bArr, 0, 16, "DESede");
}

public static String h(SecretKey secretKey, String str) throws NoSuchPaddingException, NoSuchAlgorithmException, InvalidKeyException, BadPaddingException, IllegalBlockSizeException {
    Cipher instance = Cipher.getInstance("DESede/ECB/NoPadding");
    instance.init(2, secretKey);
    return a(instance.doFinal(i(str))).substring(2, 6);
}
```

Another issue, is derived from the use of bad hash algorithms taking some reference to the previous hint, that suffer from possible *collision* as a probability to hash some messages and find one other that hashed to the same value. In two classes is used SHA-1 algorithm produces a 160-bit output of a message with a maximum length of 2^{64} bit for *collision resistance*. [10] In others two classes use the MD5 algorithm, but also compute 128-bit outputs, so that, behave worse. With a *brute-forcing* at our time it takes a relatively affordable amount of time to find a matched pair, due to technological and computational improvements in the years. Of course, for affordable is considered that anyway is needed a high value of computational machine, and time that depends on it. A possible solution is to consider implementing a more secure algorithm with SHA-2 that is of the same family of MD5 and SHA-1, or SHA-3 the latest, that changes the structure over a *Keccak* new generation group.

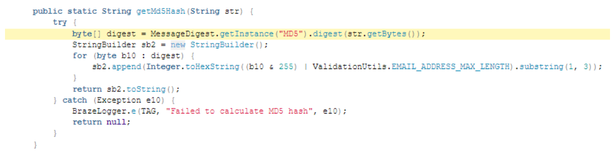


Fig. 2. Class including MD5 implementation.

```
public static String getMd5Hash(String str) {
    try {
        byte[] digest = MessageDigest.getInstance("MD5").digest(str.getBytes());
        StringBuilder sb2 = new StringBuilder();
        for (byte b10 : digest) {
            sb2.append(Integer.toHexString((b10 & 255) | ValidationUtils.EMAIL_ADDRESS_MAX_LENGTH).substring(1, 3));
        }
        return sb2.toString();
    } catch (Exception e10) {
        BrazeLogger.e(TAG, "Failed to calculate MD5 hash", e10);
        return null;
    }
}
```

The choice of the correct random generator is also important, especially if it is used with some weak or broken algorithms. It is recommended for the reason that a random number is used generally for creation of some keys for an cryptographic algorithm or hash function and the condition that someone can reduce the collision resistance or restrict the space of possible keys it can be extremely dangerous. In a sophisticated attack, after snatching some basic information about time of creation of the key, and on the other side a weak random number generator is used, the attacker can obtain a big advantage in the fact of *brute-forcing* attack having a reducing domain. The why stands for in creating a random number with few, single and-or close to predictability *seed* (a number that systematically modified give a key), is the way to create an equal value of key, so that, is a must that it is to be obtained from many sources that can vary from the environment to the computer status. In many classes is used *Java.util.Random* that is not considered suited for an cryptographic implementation, so that instead, it can be a solution to use *java.security.SecureRandom* that produce non-deterministic output [10] and meet the security compliance of RFC 1750: Randomness Recommendations for Security.[11]

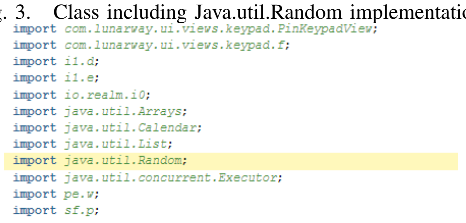


Fig. 3. Class including Java.util.Random implementation.

```
import com.lunarway.ui.views.keypad.PinKeypadView;
import com.lunarway.ui.views.keypad.f;
import i1.d;
import i1.e;
import io.realm.i0;
import java.util.Arrays;
import java.util.Calendar;
import java.util.List;
import java.util.Random;
import java.util.concurrent.Executor;
import pe.w;
import sf.p;
```

The application can use others in the phone for simple task, like photo, video, managing files except for an initial authentication using *Nem-ID* a Danish institutional application that provide a validation of your identity for digital environment, making easier and quicker verify the truthfulness of the future subscriber. The abuse of logging or making "temporarily" this type of information, can lead to a possible tracing of the person, after a stolen device and an accurate research inside the app files.

When we use the definition "on-line bank" in this report, we consider that this bank offers its services totally on-line and most interesting, exclusively via mobile android-iOS application, with no access via internet through browsers.

The bank has no international connotations because of starting requirements, this is because the service is allowed only to the owner of a Danish mobile-telephone number.

The precise customer targeting it can be considered a little vantage in terms of security due to a less management of servers around the world and the adaptation of technical adjustment for every country or continent. In general, the security practice rules are common around the world, but it needs to be fitted to every use case, for example inputs filters and privacy management.

B. *Network security*

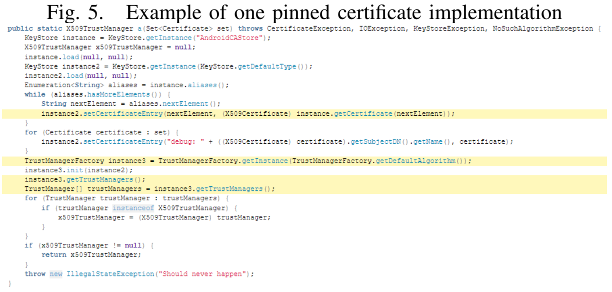
This part of the tests was doing with several tools and together the system is setted for create a MITM Attack as *Man In The Middle Attack*, that modify the communication between the true receiver and the sender, creating an intermediate spot where the attacker can have the control of the communication, typically a work of *eavesdropping* but send, select, delete message it can be wanted. The certificates *pinning* plays an important role, because the server has to verify the provenience or in general the validity of the certificate to accept secured connection over TLS.

Fig. 4. MITM setup

It is created ad hoc in an environment of application that shows and controls operation over this attack. For emulate an Android mobile phone is used a Window version of the program named *Android Studio*, the app Lunar is installed and configured network setting to point to transfer the traffic to the *localhost* path via *proxy* with same operation for the local OS (Operative System) network setting for redirect all the traffic. Here takes place the Mitmproxy that captures and shows the main feature of all the traffic, using a self-signed certificate. Finally is also used *Wireshark*, an application that reveals all the details of the feature of every traffic, that later is described. Practically in this case, the MITM Attack it not was successful because the certificate is *pinned*, so the server has detected an another provenience on the certificate and refused the connection. Certificate pinning is implemented as of associating the application with their expected X.509 certificate, and no other certificate is accepted like a valid certificate signed by others trusted CA *Certificate Authority* .The certificate is *pinned* and hard-coded into the app and retrieved at the time the app first connects to the server. In the app it is not possible to do anything out of a secured connection, so the entire service is passed through a secured TLS connection.

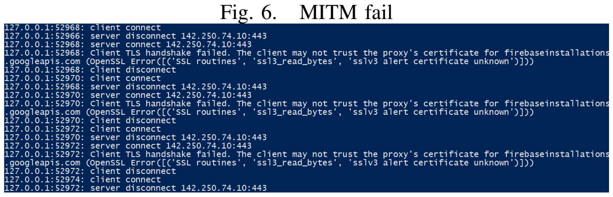
The traffic as mentioned before is passed under magnifying glass of *Wireshark* and with this, the sequence of the *handshake* is cleared and reveal the first approach to the client as our *proxy* and server, with manifesting its preference or availability to the cipher-suite and hash-suite and the firstly more important, the TLS version that is in this case V1.3, considered latest secure and more fast. The protocol consists

Fig. 5. Example of one pinned certificate implementation



```
public static X509TrustManager a(Set<Certificate> set) throws CertificateException, IOException, KeyStoreException, NoSuchAlgorithmException {
    KeyStore instance = KeyStore.getInstance("AndroidCAStore");
    X509TrustManager x509TrustManager = null;
    instance.load(null, null);
    KeyStore instance2 = KeyStore.getInstance(KeyStore.getDefaultType());
    instance2.load(null, null);
    Enumeration<String> aliases = instance.aliases();
    while (aliases.hasMoreElements()) {
        String nextElement = aliases.nextElement();
        instance2.setCertificateEntry(nextElement, (X509Certificate) instance.getCertificate(nextElement));
    }
    for (Certificate certificate : set) {
        instance2.setCertificateEntry("debug: " + ((X509Certificate) certificate).getSubjectDN().getName(), certificate);
    }
    TrustManagerFactory instance3 = TrustManagerFactory.getInstance(TrustManagerFactory.getDefaultAlgorithm());
    instance3.init(instance2);
    instance3.getTrustManagers();
    TrustManager[] trustManagers = instance3.getTrustManagers();
    for (TrustManager trustManager : trustManagers) {
        if (trustManager instanceof X509TrustManager) {
            x509TrustManager = (X509TrustManager) trustManager;
        }
    }
    if (x509TrustManager != null) {
        return x509TrustManager;
    }
    throw new IllegalStateException("Should never happen");
}
```

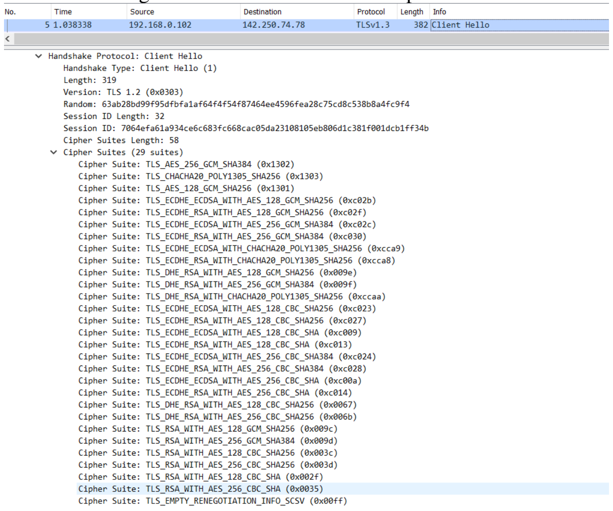
Fig. 6. MITM fail



```
127.0.0.1:52968: client connect
127.0.0.1:52966: server disconnect 142.250.74.10:443
127.0.0.1:52968: server connect 142.250.74.10:443
127.0.0.1:52968: Client TLS handshake failed. The client may not trust the proxy's certificate for firebaseinstallations
.googleapis.com (OpenSSL Error([('SSL routines', 'ssl3_read_bytes', 'sslv3 alert certificate unknown')]))
127.0.0.1:52968: client disconnect
127.0.0.1:52970: client connect
127.0.0.1:52968: server disconnect 142.250.74.10:443
127.0.0.1:52970: server connect 142.250.74.10:443
127.0.0.1:52970: Client TLS handshake failed. The client may not trust the proxy's certificate for firebaseinstallations
.googleapis.com (OpenSSL Error([('SSL routines', 'ssl3_read_bytes', 'sslv3 alert certificate unknown')]))
127.0.0.1:52970: client disconnect
127.0.0.1:52972: client connect
127.0.0.1:52970: server disconnect 142.250.74.10:443
127.0.0.1:52972: server connect 142.250.74.10:443
127.0.0.1:52972: Client TLS handshake failed. The client may not trust the proxy's certificate for firebaseinstallations
.googleapis.com (OpenSSL Error([('SSL routines', 'ssl3_read_bytes', 'sslv3 alert certificate unknown')]))
127.0.0.1:52972: client disconnect
127.0.0.1:52974: client connect
127.0.0.1:52972: server disconnect 142.250.74.10:443
```

of a single message, which is encrypted and compressed under the current Cipher Spec and agrees for selecting the cipher-suite from the list for the next messages.

Fig. 7. Client Hello and the Cipher suite



No.	Time	Source	Destination	Protocol	Length	Info
5	1.038338	192.168.0.102	142.250.74.78	TLSv1.3	382	Client Hello

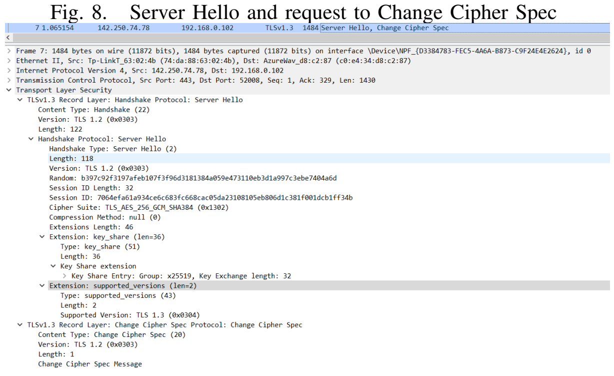
```
Handshake Protocol: Client Hello
    Handshake Type: Client Hello (1)
    Length: 319
    Version: TLS 1.2 (0x0303)
    Random: 63ab28bd99f95dfbfa1af64f4f54f87464ee4596fea28c75cd8c538b8a4fc9f4
    Session ID Length: 32
    Session ID: 7064efa61a934ce6c683fc668cac05da23108105eb806d1c381f001dcb1ff34b
    Cipher Suites Length: 58
    Cipher Suites (29 suites)
        Cipher Suite: TLS_AES_256_GCM_SHA384 (0x1302)
        Cipher Suite: TLS_CHACHA20_POLY1305_SHA256 (0x1303)
        Cipher Suite: TLS_AES_128_GCM_SHA256 (0x1301)
        Cipher Suite: TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256 (0xc02b)
        Cipher Suite: TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 (0xc02f)
        Cipher Suite: TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 (0xc02c)
        Cipher Suite: TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 (0xc030)
        Cipher Suite: TLS_ECDHE_ECDSA_WITH_CHACHA20_POLY1305_SHA256 (0xcca9)
        Cipher Suite: TLS_ECDHE_RSA_WITH_CHACHA20_POLY1305_SHA256 (0xcca8)
        Cipher Suite: TLS_DHE_RSA_WITH_AES_128_GCM_SHA256 (0x009e)
        Cipher Suite: TLS_DHE_RSA_WITH_AES_256_GCM_SHA384 (0x009f)
        Cipher Suite: TLS_DHE_RSA_WITH_CHACHA20_POLY1305_SHA256 (0xccaa)
        Cipher Suite: TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256 (0xc023)
        Cipher Suite: TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256 (0xc027)
        Cipher Suite: TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA (0xc009)
        Cipher Suite: TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA (0xc013)
        Cipher Suite: TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384 (0xc024)
        Cipher Suite: TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384 (0xc028)
        Cipher Suite: TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA (0xc00a)
        Cipher Suite: TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (0xc014)
        Cipher Suite: TLS_DHE_RSA_WITH_AES_128_CBC_SHA256 (0x0067)
        Cipher Suite: TLS_DHE_RSA_WITH_AES_256_CBC_SHA256 (0x006b)
        Cipher Suite: TLS_RSA_WITH_AES_128_GCM_SHA256 (0x009c)
        Cipher Suite: TLS_RSA_WITH_AES_256_GCM_SHA384 (0x009d)
        Cipher Suite: TLS_RSA_WITH_AES_128_CBC_SHA256 (0x003c)
        Cipher Suite: TLS_RSA_WITH_AES_256_CBC_SHA256 (0x003d)
        Cipher Suite: TLS_RSA_WITH_AES_128_CBC_SHA (0x002f)
        Cipher Suite: TLS_RSA_WITH_AES_256_CBC_SHA (0x0035)
        Cipher Suite: TLS_EMPTY_RENEGOTIATION_INFO_SCSV (0x00ff)
```

Some of the server used are analyzed via *Qualys SSL Labs* that shows in the majority a good grade A+ and A. But in some other a B grade due to an old implementation (or acceptance) of TLS v1.1 that use *Pseudo Random Algorithm* as combination of MD5 and SHA-1 based. An upgrade to a TLS V1.2 or more, is advised.

C. *Authentication*

The identification takes place as we mentioned before, firstly with an institutional application *Nem-ID* and after this task, is no longer used. The identification also exchanges documents directly in the application, with no exchanging of documents through e-mail or something not native secured. After that comes the time to have all the instruments for authorization; the very first approach is setted to a pin with four numeric ciphers, which can be considered a basic approach oriented to fast login instead of security view, for

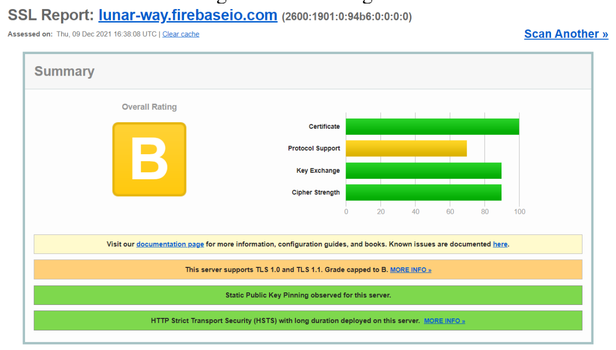
Fig. 8. Server Hello and request to Change Cipher Spec



```
      7 1.065154      142.250.74.78        192.168.0.102        TLSv1.3  1484 Server Hello, Change Cipher Spec

> Frame 7: 1484 bytes on wire (11872 bits), 1484 bytes captured (11872 bits) on interface \Device\NPF_{D3384783-FEC5-4A6A-B873-C9F24E4E2624}, id 0
> Ethernet II, Src: Tp-LinkT_63:02:4b (74:da:88:63:02:4b), Dst: AzureWav_d8:c2:87 (c0:e4:34:d8:c2:87)
> Internet Protocol Version 4, Src: 142.250.74.78, Dst: 192.168.0.102
> Transmission Control Protocol, Src Port: 443, Dst Port: 52008, Seq: 1, Ack: 329, Len: 1430
v Transport Layer Security
  v TLSv1.3 Record Layer: Handshake Protocol: Server Hello
      Content Type: Handshake (22)
      Version: TLS 1.2 (0x0303)
      Length: 122
    v Handshake Protocol: Server Hello
        Handshake Type: Server Hello (2)
        Length: 118
        Version: TLS 1.2 (0x0303)
        Random: b397c92f3197afeb107f3f96d3181384a059e473110eb3d1a997c3ebe7404a6d
        Session ID Length: 32
        Session ID: 7064efa61a934ce6c683fc668cac05da23108105eb806d1c381f001dcb1ff34b
        Cipher Suite: TLS_AES_256_GCM_SHA384 (0x1302)
        Compression Method: null (0)
        Extensions Length: 46
      v Extension: key_share (len=36)
          Type: key_share (51)
          Length: 36
        v Key Share extension
          > Key Share Entry: Group: x25519, Key Exchange length: 32
      v Extension: supported_versions (len=2)
          Type: supported_versions (43)
          Length: 2
          Supported Version: TLS 1.3 (0x0304)
  v TLSv1.3 Record Layer: Change Cipher Spec Protocol: Change Cipher Spec
      Content Type: Change Cipher Spec (20)
      Version: TLS 1.2 (0x0303)
      Length: 1
      Change Cipher Spec Message
```

Fig. 9. Server B graded



SSL Report: lunar-way.firebaseio.com (2600:1901:0:94b6:0:0:0:0)

Assessed on: Thu, 09 Dec 2021 16:38:08 UTC | Clear cache

Scan Another »

Summary

Overall Rating

B

Visit our documentation page for more information, configuration guides, and books. Known issues are documented here.

This server supports TLS 1.0 and TLS 1.1. Grade capped to B. MORE INFO »

Static Public Key Pinning observed for this server.

HTTP Strict Transport Security (HSTS) with long duration deployed on this server. MORE INFO »

TABLE I
GRADE RESUME OF OTHER SERVERS USED

SERVER	GRADE
www.lunar.app	A+
api.prod.lunarway.com	A+
static-assets.prod.lunarway.com	A+
cdn.prod.lunarway.com	A
lunar-way.firebaseio.com	B
lunarway-prod-cdn.s3-eu-west-1.amazonaws.com	B
lunarway.com	B

a only 10.000 combination possible. It gives five attempts to insert the right pin and then it blocks the functionality. To be mentioned is the possibility to change the authentication via bio-metrics method, as only the fingerprint, that change the feeling or personalize the security, but not improve at all the security as we describe later. The choice of the authentication tools principally are divided into four categories: something that the individual *knows*, the classical password, the PIN as Personal Identification Number (only numeric) or a set of personal questions for a password recovery for example. Then we have something that an individual *has*: cryptographic keys into electronic cards or general hardware keys, rotational keys binded to an application. The third is something that individual *is*, known as biometric but in static fashion like fingerprints, face and retina. Lastly we can find dynamic biometrics as something that an individual *does* like voice pattern, handwriting characteristics, movement-walk pattern.

In this application as we mentioned before is allowed only one of these security practices, even the PIN has only five attempts to give the right number before blocking or using fingerprints. Only one-way authentication factor nowadays is to be considered customer-comfortable for a fast reaching of the service, but every one of the types of authentication method can be tricked. A PIN can be discovered from applying social-engineering from the social network for example or having some basic information about the target. The fingerprints can be replicated with some sophisticated techniques or forced physically in some dangerous condition without spending too much time. This is why a two-way *authentication* is preferable, for letting the act of doing something bad, more difficult and time consuming. Also a PIN with six numbers, is advisable for avoid *social-engineering* on only four numbers. Implementing a face-recognition it can be also a valid alternative but unfortunately some users at now may have not this type of technology installed on the smartphone due to less common choice from phones producers.

Every single transaction that provides the transferring of money, is treated as another level, that is, require another biometric or PIN authorization in order to proceed, so potentially the forcing of biometric-authorization can be a little mitigated. Password recovery is made through *Nem-ID*, that use also 4 number PIN and fingerprints, so there is same considerations and security balancing.

Last mention, is a possibility to use an internal chat for some requesting or info where the *Authentication* is taken from the access to the app and a call-center number that require only the CPR number for been authenticated and access to private information, that for clearing is composed from date of birth and four random number similar to the PIN consideration.

D. Privacy

Privacy has taken relevance year after year, coming in the later stages when the personal data have not a specific regulation arriving to this last years where GDPR (*General Data Protection Regulation*) has taken place, with benefits that follows. The data manager is made responsible for the holding of personal data, and introduces sanctions for who have made a fraud management.

An example of opaque use of the data was the case of *Cambridge Analityca* scandal that through a well-known social media Facebook [11], a considerable amount of personal data was collected over a consent of the social media platform to be refined for profiling, without consent people to agree to this practice.

The analysis of the *tracker*, i.e., an application that can withdraw data and can send to a third party collector, is made by *Exodus* an audit privacy platform for Android applications. Having a list of *tracker*, starting from *AppFlyer* that is used, the auditor platform has associated the properties: Fraud prevention, malware protection. Identity attribution services. Fingerprints devices by their IDs and tracks across datasets. Mobile services for targeted advertising. Marketing across devices and channels. Generates reports based on user behavior. Tracks which users install apps; tracks successful re-targeting marketing campaigns. Is specified in the *Privacy Policy* also that *AppFlyer*, doesn't sell any data to third party.

The second *tracker* used is named *Braze* and is indicated for having the characteristic of *Cross-Channel Personalization* (providing personal experience and service to customers across multiple channels): Target customers based on personal interests, location, past purchases and more. In the *Privacy Policy* is stated that *Braze* collects data through the use of Developer apps and data is shared with app Developers in the form of aggregated, anonymous data. In the *Privacy Policy* of the website, where is located the general condition, i.e. all program that they offer is stated, that they disclose information to Braze corporate group affiliates, consultants, third-party vendors and other service providers, in connection with customer or technical support, marketing, recruiting, operations, account management, and general business purposes. [12] This statements are quite suspicious, but not so much in this case, because Braze offer different services for different area and for the record, Lunar as mentioned before, use an product addressed to developer instead marketer.

Lastly *Crashlytics* from Google, similar to *Braze* offers a crash reporting solution for app developers. In this *Privacy Policy*, happen something unusual that, in some way, invites the company to draw up his own *Privacy policy* and show to their user, adding the user-enterprise have to not facilitate the merging of personally-identifiable information, finally, advise to change type of Google service if the user-enterprise have to collect data for the purpose of interest based advertising. [13] Similar to the previous one *Google Firebase Analytics* (is considered as a parallel service for *Crashlytics*) gives the functionality like analytics, databases, messaging and crash reporting. The *Privacy Policy* here is similar but more concise, who uses this application, have to filter the data to not give personal data of this service. So, apparently in both of this services, the intention is to falls the responsibilities in some part to the enterprise that uses this two Google services. [14]

To conclude, reading through *Privacy Policy* statement achieved from the Lunar website [15], as well as technical-financial-legal treatment of the data there is a passage that describes the "Distribution of marketing materials" is directly connected with the customer agreement choice. And is reported the possibility to object to our use of your personal data in connection with direct marketing, including profiling in connection with such a purpose. No citation, to a possible trade of data or untraceable-person data, for selling. Essentially state also, that profiling is used mainly for decision in internal financial circumstances.

IV. CONCLUSIONS

This analysis has the scope to highlight all security and privacy aspects that are noteworthy, and it has emerged that globally the development of the app has a good approach to network security, but consider to upgrade the version of TLS in some servers it can be a reasonable thinking, even they are not related to a public service offer. For the part software security has to be improved as described before, and some possible precaution can be implemented with a low effort and good effectiveness. For what concerns privacy there is no dark side, and it is reasonable to assert that the management of the data is correct and respectful of Privacy, but maybe the *Privacy Policy* may spend some words more to clearing better the aspect of marketing and *third parties*. In the future is desirable, not specifically from this app but form Governments, a better consideration of privacy more near at security, not only a statement that addresses for accomplishing the laws or other law firms, but a more user-friendly *Privacy Policies*, in a check-list style with a common vocabulary that everyone can understand, so that, people can estimate the value of their data and decide consciously.

REFERENCES

[1] OWASP Threat Modeling (Online). Available: https://owasp.org/www-community/Threat_Modeling#.

[2] W. Stallings, "Cryptography and Network Security" , Ed. Pearson, 2017, pp. 37.

[3] OWASP Top Ten Web Application Security Risk (Online). Available: https://owasp.org/www-project-top-ten/.

[4] Wil Allsopp, "Advanced Penetration Testing", Ed. Wiley, 2017.

[5] Common Weakness Enumeration (Online). Available: https://cwe.mitre.org/index.html.

[6] NIST National Vulnerability Database (Online). Available: https://nvd.nist.gov/vuln-metrics/cvss.

[7] The Open Web Application Security Project® (Online). Available: https://owasp.org/.

[8] Daniel Bleichenbacher, "Chosen Ciphertext Attacks Against Protocols Based on the RSA Encryption Standard PKCS 1", Bell Laboratories.

[9] OWASP Testing Padding Oracle (Online). Available: https://owasp.org/www-project-web-security-testing-guide/latest/4-Web_Application_Security_Testing/09-Testing_for_Weak_Cryptography/02-Testing_for_Padding_Oracle

[10] Java™ Platform Standard Ed. 8, Class SecureRandom (Online). Available: https://docs.oracle.com/javase/8/docs/api/java/security/SecureRandom.html.

[11] The Cambridge Analytica scandal changed the world – but it didn't change Facebook (Online). Available: https://www.theguardian.com/technology/2019/mar/17/the-cambridge-analytica-scandal-changed-the-world-but-it-didnt-change-facebook.

[12] Legal - Privacy Policy (Online). Available: https://www.braze.com/company/legal/privacy.

[13] Crashlytics and App Distribution Terms (Online). Available: https://firebase.google.com/terms/crashlytics.

[14] Firebase App Indexing User Data Policy (Online). Available: https://firebase.google.com/policies/app-indexing.

[15] Terms and Compliance (Online). Available: https://www.lunar.app/en/personal/terms-and-conditions.