Risk and Threat Analysis in a Mobile Environment

Introduction

Unit 1

Index

Extension of Functions and Mobile Devices Usage in a Business Context

Extension of Functions

Smartphones, Enterprises and Security

Smartphones and the constant connectivity to the Internet have supposed a revolution in organizations' way of working.

Smartphones have become an essential tool in order to perform tasks within an organization.

Whenever there is a smartphone with Internet connection, the employee has a mobile office to perform multiple tasks.

The use of smartphones also modifies some aspects related to security within an organization both in positive and negative ways.

Writing emails

Using internal messaging systems

Using work tools from the mobile device:

- Web browser
- Equipment management
- Documents editing and sending

Extension of Functions

Some Advantages

1 It enables a simple integration of some existing platforms within the organization. This way, it is not necessary to perform actions that may pose risks for the organization when carrying out tasks from the mobile device:

- For example: Microsoft Office Mobile enables the synchronisation of documents of the organization by using the same platform utilized by the workstations (OneDrive). This way, it is not necessary to extract sensitive data of the organization (via USB or personal email).

2 Mobile devices already include multiple security features by default that, in the case of personal computers, have to be implemented via third parties' tools:

- Disk encryption mechanisms by default.

3 The operative system is designed to isolate some applications from the rest. This way, in the event of having a security problem in a given application, it generally will not affect the rest.

Extension of Functions

Some Disadvantages

1 Smartphones are a new attack vector that can be used to obtain access to a given organization:
For example: since the device is a personal tool, it sometimes stores access credentials to the organization. That is the reason why it is essential to properly protect them.

2 There is a duality between devices belonging to the organization and employees' personal devices:
Bring Your Own Device policies are aimed at solving this problem.

3 In the event of allowing the use of personal devices within the organization, the fact of trusted individuals using them may pose a risk for the confidentiality of the organization's information:
Relatives that use the device sporadically sending sensitive information by mistake via corporate email accounts.

Extension of Functions

Bring Your Own Device

Set of policies that allow employees to use their personal devices when performing work tasks.

The adoption of this policy is mainly due to:

The difficulty of avoiding employees to use their personal devices at work.

Costs reduction by the organization.

Generally, it is difficult to implement an effective BYOD policy in an organization, since it is quite hard to define limits between personal and labour aspects when using a device.

In the last few years, providers have made a great effort to offer efficient solutions within this context.

Extension of Functions

Bring Your Own Device

Samsung Knox:

- It creates two application containers in the same device.
- Applications from a container cannot communicate with the ones from the other.
- It is too restrictive sometimes.

Android for Work:

- It enables the installation of a set of apps exclusively used within the labour environment and are isolated from the rest of applications.
- Limited to Google applications.

Mobile Device Management:

- Device management systems within the organizations.
- It limits actions and enables the automatic installation of configurations.
- If the employee uses a personal device, it is important to allow the organization to have a huge control over it.

Bring Your Own Device and the Cloud

- The use of smartphones and associated applications very often requires the organization to outsource certain services:
  - Productivity software in the cloud.
  - Accounting in the cloud.
  - Files storage.
- On many occasions, these services handle corporate sensitive data that has to be properly processed.
- Generally, service providers have specific accounts for organizations with greater protection measures, different from accounts offered to usual clients:
  - Some of them include Service Level Agreements.
  - Encrypted data storage.
  - Management of employee's accounts and remote deletion of data.

Examples of Recent Incidents

Examples of Recent Incidents

Introduction

Even if, on many occasions, security breaches resulting from the misuse of smartphones are not public, some cases have been published.

Examples of Recent Incidents

Loss of Sensitive Information

Eventbrite

An employee from the events organization company lost two iPads while in transit. Lost data included:

- Emails.
- Full credit card numbers of 28 people.

http://www.eventbrite.com/blog/our-commitment

Examples of Recent Incidents

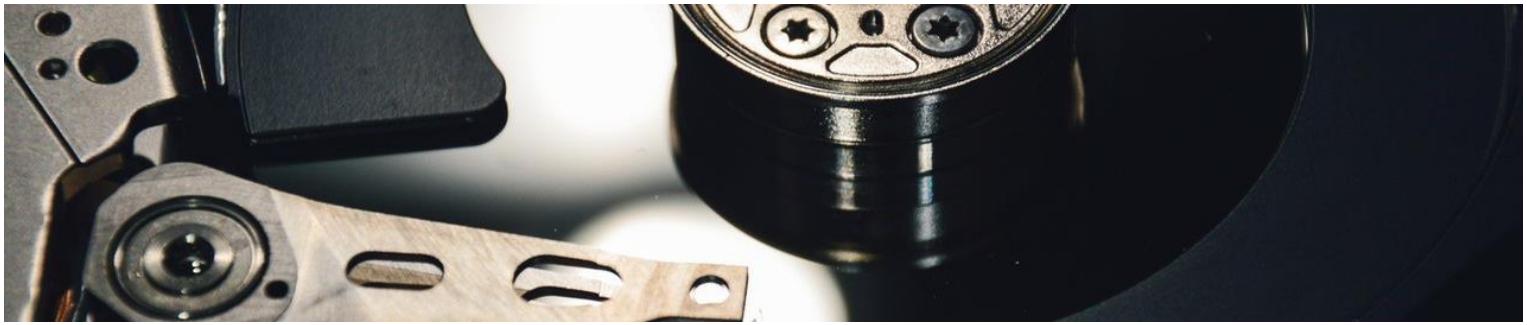
Security Failure

Moonpig

Mobile application of a seller of greeting cards. The API used for the mobile version did not verify credentials to access personal information of the users.

- The failure was reported in 2013, but it was not solved until 2015.
- The company did not inform users about the security breach.

http://www.techworld.com/news/security/moonpig-android-app-flaw-puts-three-million-accounts-at-risk-3592812/



Examples of Recent Incidents

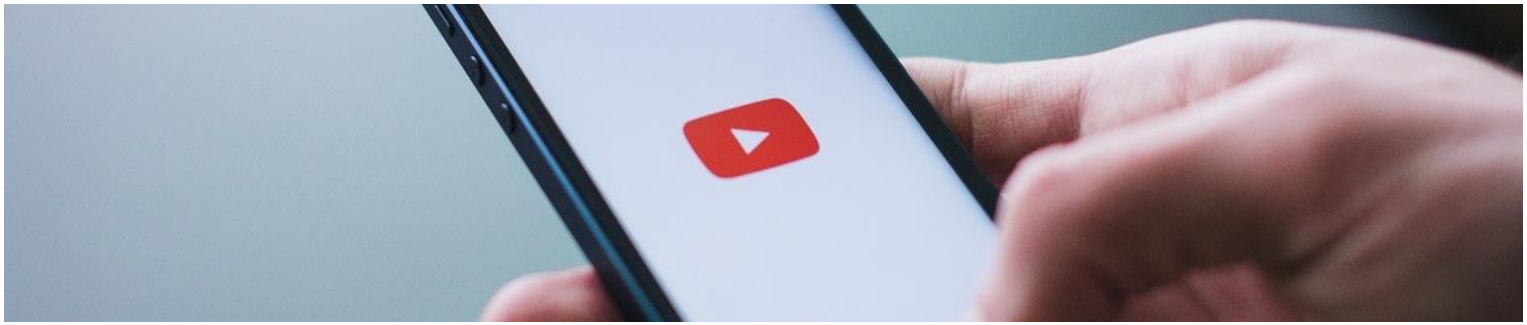
Applications Vulnerabilities

Vulnerability in the UPnP library

UPnP libraries are used for video streaming between devices. There are more than 6 million devices that use such libraries in their devices: TV, smartphones, etc.

The vulnerability enabled the execution of random code in a device that had one of those applications installed.

http://blog.trendmicro.com/trendlabs-security-intelligence/high-profile-mobile-apps-at-risk-due-to-three-year-old-vulnerability



Examples of Recent Incidents

Applications Vulnerabilities

Car Parking applications

An audit firm detected vulnerabilities in multiple applications for the car parking payment in United Kingdom. It allowed attackers to know some users' location and credentials.

http://www.theregister.co.uk/2015/12/11/mobile_parking_apps_audit

Examples of Recent Incidents

Financial Sector

Banking applications

Analysis of 40 iOS banking applications available at:
http://blog.ioactive.com/2014/01/personal-banking-apps-leak-info-through.html

Noteworthy results:

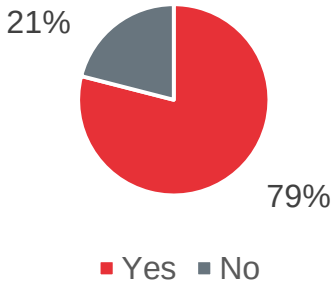
- The 50 % of the analysed applications were vulnerable to *Cross Site Scripting* on the clients side.
- The 30 % had credentials written directly in the code.
- The 40 % leaked information sensitive to logs.
- The 40 % did not properly validate SSL certificates.
- The 20 % sent non-encrypted sensitive information through the network.

Risk Analysis in the Mobile Environment

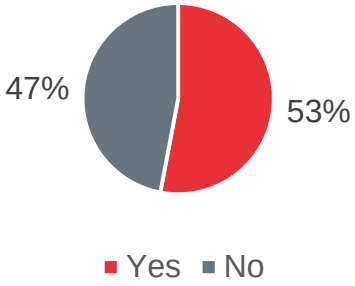
Introduction: Business Environment

- Survey of more than 700 professionals of the sector, conducted by Dimensional Research in June 2013 (https://www.checkpoint.com/downloads/products/check-point-mobile-security-survey-report.pdf).

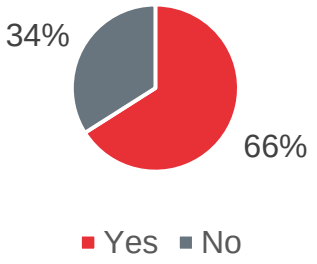
Had to face security incidents related to mobile devices.



Mobile devices including information related to clients.



Consider that careless employees suppose a greater risk than cybercriminals.



Introduction: Business Environment

The mobile device improves the employee's productivity.

The same as all devices that are part of an organization, mobile environments create great number of threats for the organization.

In some cases, threats created by these devices are similar to the ones from the classic business environment. However, specific characteristics of the mobile devices make them susceptible to a quite varied group of threats.

If users are not aware of such threats, they may pose a risk for data, business processes and assets. It may even endanger the continuity of the organization.

The main risks created by mobile devices within an organization are analysed below.

Loss or Theft

- Due to the size, mobility and price, mobile phones are lost or stolen quite often.
- It is the most common threat that organizations have to face.

Two risks in one:

- If there is no backup, data will be lost.
- Unauthorised people may access data.

- It is necessary to assume that both risks may occur since it is not possible to know if any of them would actually become a threat.
- Depending on lost data and regulations, it may be necessary to notify the relevant authorities.
- The employee sometimes notifies it late or does not even inform about it.
- Depending on whether the device was under the organization's control or was not, actions to mitigate the threat may be less effective.

Unauthorised Access

- The attacker may access the following data depending on the protection level that the device provides, applications installed and the attacker's technical level:

Applications data and files:

- Emails and files directly stored in the device or its removable media (in case that there is any) containing confidential information.

Applications credentials:

- It is possible to access users, passwords and tokens for direct access to applications through forensic analysis.

Access to already authenticated resources:

- In some cases, if the phone is unlocked, the intruder is be able to access some services whose apps are already pre-authenticated (social networks, messaging systems, etc.).

Access to data stored in applications' databases:

- Conversations, photographs and any further information stored in local database files.

Attacks

If the attacker finds the device active and unblocked, the attacker only has to maintain it unblocked as long as possible until the opportunity of extracting data arises.

In case that the device is blocked, there are different possible attacks aimed at accessing information.

Attacks may be used to attempt to access the whole device or the information of a specific application.

Attackers often need additional resources such as a computer to execute algorithms for the testing or extraction of passwords.

Some examples are described in the following slides.

Android Cool Boot Attack

- RAM memory of a device needs electricity to work. When a device is turned off, the electric current stops flowing through the circuit and data will be lost little by little. The temperature of the device has a great effect on the RAM deleting speed. The colder it is, the longer it takes to delete data.
- Applied to Android:

The blocked telephone is placed in the freezer for 1 hour.

Then, it has to be connected to a Linux-based computer through a USB connection and it is restarted.

Before charging the main operative system, via the USB connection a start module is loaded that reads contents of the RAM memory that enables the extraction of passwords and any other data stored in the memory.

- Further information available at:
  - https://www1.informatik.uni-erlangen.de/frost

Access to WhatsApp's Database

- WhatsApp messaging program saves all the conversations in an encrypted database on the device.
- The encryption key is created during the first execution of the application and is saved as a parameter of the program.
- There are programs (for Android) that enable the extraction of such information:

- Further information available at: https://github.com/AbinashBishoyi/WhatsApp-Key-DB-Extractor-UnOfficial

iPhone Passcode Bypass

- There are solutions to conduct brute-force attacks on iOS devices.

Hardware
(http://blog.mdsec.co.uk/2015/03/bruteforcing-ios-screenlock.html):

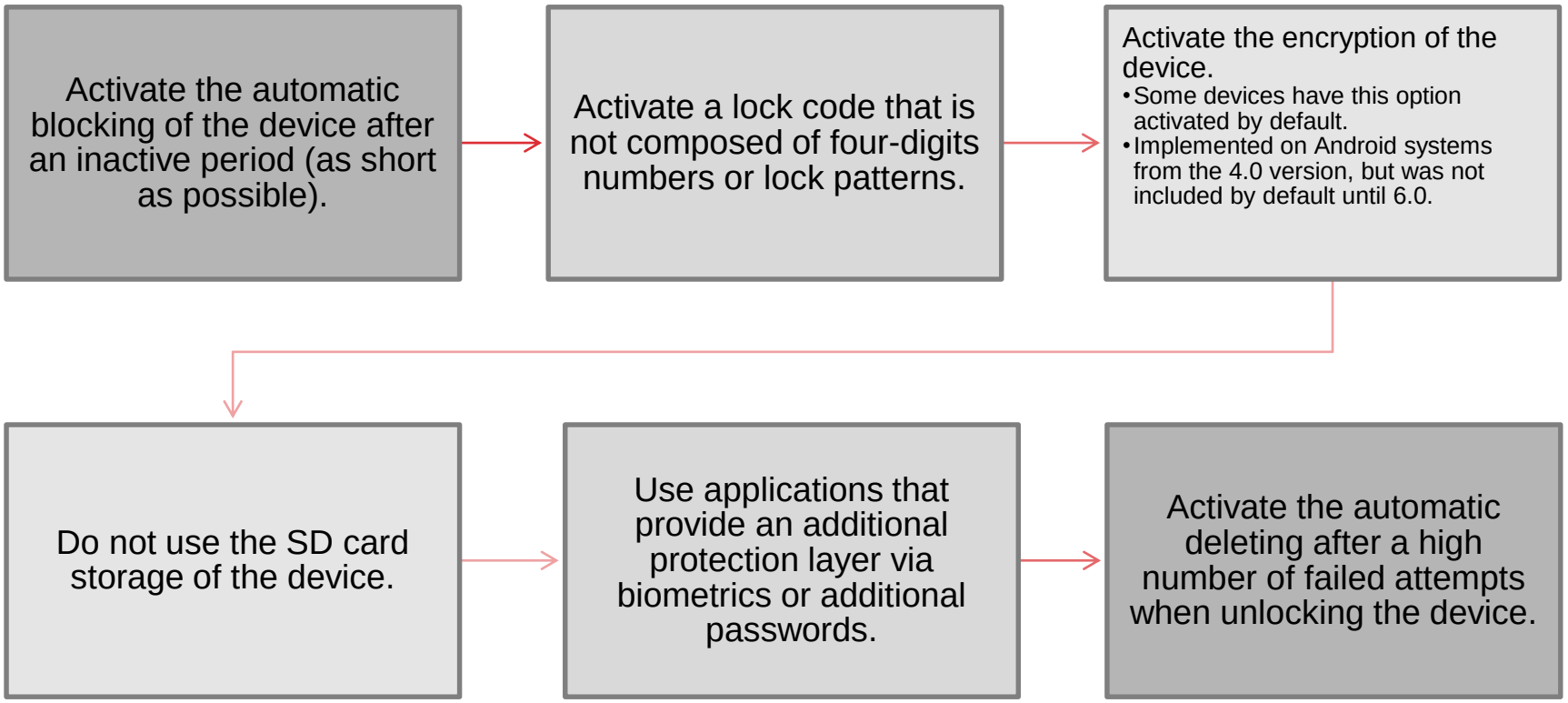
- All 4-digits possible keys are tried via USB connection.
- A light sensor detects whether the introduced key is correct or is not.
- If it is not, the device is turned off very quickly so that data is not automatically deleted after the maximum number of attempts.
- 40 seconds per key, up to 117 hours to try all the keys.

Software
(http://www.iphonehacks.com/2015/03/iphone-passcode-bypassed-bruteforce-tool.html):

- Similar solution but it only works with Jailbreak devices.
- In this case, only 5 seconds per key are required.

Countermeasures

- Even if it is quite difficult to implement measures to avoid the theft or loss of a mobile device, it is possible to implement countermeasures in order to reduce the risk for the organization in case of such events occurring:

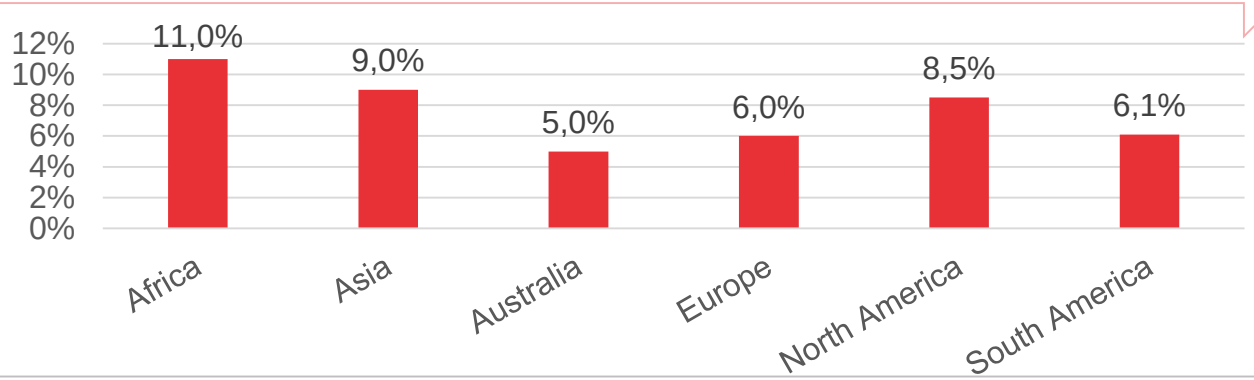


Malware

- Any application that performs unauthorised operations with the aim of damaging a device, a user or an organization is considered a malware.
- *The* amount of malware existing for such devices has grown exponentially in recent years. (http://www.mcafee.com/us/resources/reports/rp-quarterly-threat-q1-2015.pdf):

During the first quarter of 2015, McAfee detected more than a million evidences of new malware. In total, more tan 6 million evidences have been found.

Android is the operative system that malware affected most (more than 90 %).



Malware – Infection Strategy

- Generally, almost all the applications pretend to be a different one for the user to install them:
  - Payment applications offered as free apps in alternative markets.
  - Payment applications installers.
  - Unreliable applications such as “WhatsApp Spy”, etc.

Generally, the malware does not include the mentioned application and is executed right when it is open. In some cases, the malicious code is added to the real application to make it even more realistic (repackaging).

- Android’s permit system is not very efficient, since few users thoroughly review permits required by an application.

Malware – Main Categories

- **Premium SMS**: users automatically subscribe to additional pricing services receiving SMS at extra cost.
  - https://blog.kaspersky.co.uk/fakeinst-targets-us-users/3004/
- ***Ransomware***: it encrypts as many files as possible and demands a ransom that should be transferred via untraceable transfers or bitcoin.
  - http://www.darkreading.com/endpoint/ransomware-ranked-number-one-mobile-malware-threat/d/d-id/1322886
- ***Botnets***: they allow the remote execution of commands through a command and control server. Botnets also enable access to all the existing information.

Malware – Main Categories

Malware: **Anti Botnets Services**

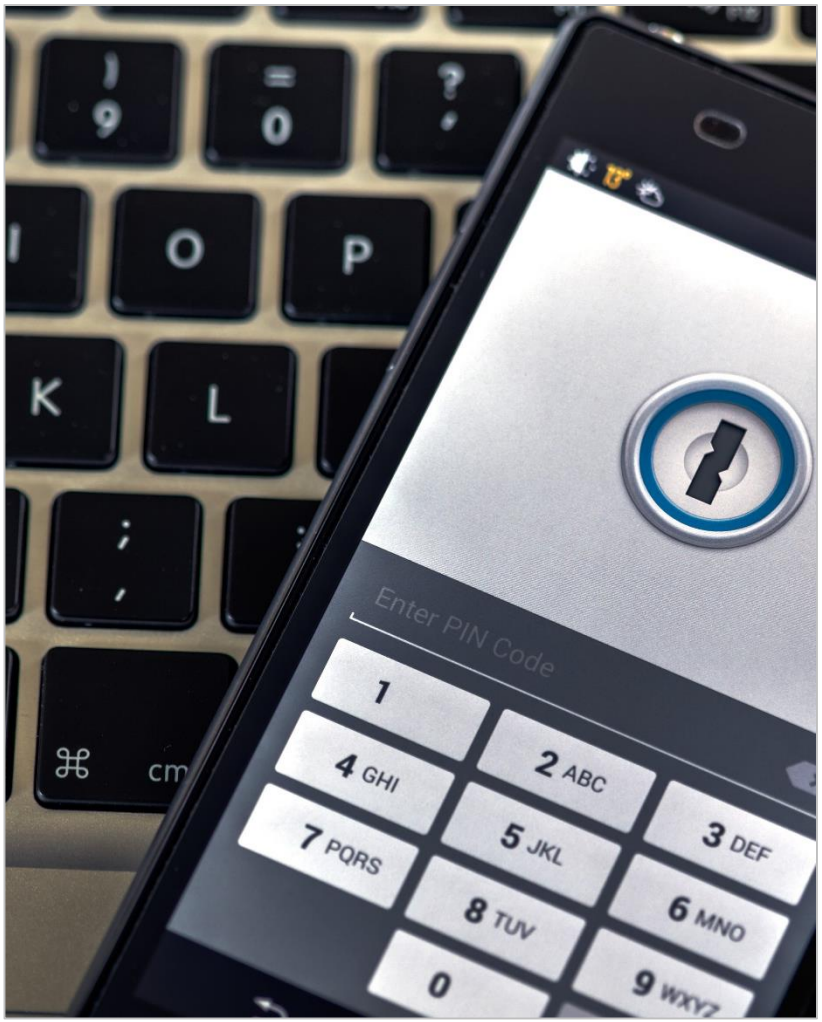
Incibe provides an anti botnets service for end users on their website. Such service allows users to check if any computer or device connected to their network connection or mobile device is infected by a botnet.

It is based on a combination of IP addresses reputation lists together with the local analysis of devices aimed at looking for known malware.

It is available at https://www.osi.es/es/servicio-antibotnet.html

Malware – Main Categories

- **Information theft**: malware that extracts information such as location, contact list and visited webs. It can be used to manage advertisements, create customized malware campaigns or send SPAM among others.
  - http://www.businessinsider.com/up-to-120000-android-phones-have-been-infected-with-malware-2011-5
- **Credential theft**: subtype of the one above. It searches sensitive applications (banking, social networks, etc.) in the device's SD card and try to steal credentials.
  - https://blog.kaspersky.es/android-banking-trojans/6939/

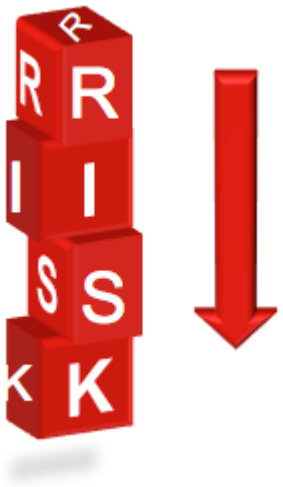


Risk Analysis in the Mobile Environment

Misuse

- Sometimes, users increase risks produced by the devices.

Some risk behaviours:



- Deactivate the lock code of the device.
- Use the browser to perform activities that can be directly carried out via apps.
- Activate the “Stay connected” option in sensitive applications.
- Activate the automatic connection to wireless networks option.
- Connect to open networks in publish places.
- Not to carry out a secure erase of the device when it is not used anymore.
- Download uncertain-origin applications.
- Not to regularly clear the browsing history.
- Store sensitive data in the device without the proper protection.
- Not to activate the remote erase system and lockout.
- Activate the unknown-origin option.

Misuse - Countermeasures

- There are two possible actions to take in order to avoid a misuse of mobile devices:

Restrictions regarding the device's features:

- Imposed by the operative system itself (for example: apple does not allow the installation of applications that are not signed by them) or use a mobile devices management solution that restricts user's actions.

Education:

- Through training courses, seminars and the real visualization of consequences that the possible misuse of a mobile device may lead to.

Misuse - Countermeasures

- On many occasions, users are not aware of when applications are using permissions.
- In the case of location information, its disclosure may go unnoticed even if the device is not being used with a fraudulent purpose:
  - Many social networks publish the location of the device when updates are published. Depending on the social network and its privacy settings, that information may be available for everyone.
  - Some other applications may access the location information even if they are not being used and without notifying the user.
  - In order to use some applications, an authorisation to access other application's information (such as Facebook) is required.

Corporate Information Disclosure

- Once a mobile device is used to perform tasks within an organization, there are multiple ways experiencing the disclosure of corporate information:

Email:	• The coexistence of various personal and work accounts may lead to a confusion and the user may send emails containing corporate information to erroneous recipients or may send sensitive information from the personal account.
Use of services unauthorized by the organization:	• It is probable that, in order to make work easier, the employee uses third party's resources (such as messaging, cloud storage, documents edition) that are unauthorised by the organisation.
Load of confidential files:	• It is also probable that the employee loads confidential files to the personal device in order to telework. Such types of copies may be made via physical connection or accessing the organisation by VNP.

Phone Cloning

- Generally, phone cloning makes reference to the fact of copying the identity of a given (SIM) device within the telephone network.
- It depends on the type of SIM card and the telephone network.
- With physical access to the device (SIM card):

  Two data are required: the IMSI and the encryption key.
  The IMSI is public, but the key is protected and can only be read in very old versions.

- With no physical access to the device (https://www.blackhat.com/us-13/briefings.html#Ritter):

  Viable attack in CDMA 3G networks.
  A femtocell is enough to conduct the attack; i.e.: a repeater that uses the Internet to connect to the telephone network.
  It enables the interception of calls, messages, visited websites and even the cloning of a SIM card.

Jailbreak

- Jailbreak (on iOS devices) or rooting (on Android systems) refers to the elimination of restrictions included in the operative system in order to avoid the execution of unsigned code.
- It is possible to execute applications with administrator privileges in a jailbroken or rooted device; it enables the modification of the system:

Installation of apps that use restricted functions.
Device customization.
Execution of pirated applications.

- They generally take advantage of system's vulnerabilities for the release. Such vulnerability could be used with malicious purposes.
- The execution of unsigned code poses a risk of uncontrolled code in the device.
- The first worm in iOS was spread due to an insecure configuration of the SSH server, after having jailbroken the device.

Wiretapping

- It is possible to install apps aimed at recording calls in some operative systems:

On Android, it requires the microphone permission.

On iOS and other systems, it requires a jailbreak or hiring a service able to redirect calls in order to record them.

- The installation of this kind of applications may be very easily carried out by accessing the device for a short period of time:

Automatic Call Recorder
https://play.google.com/store/apps/details?id=com.appstar.callrecorder&hl=en

- It is also possible to listen phone calls through the telephone network with phone cloning (mentioned before):
  - https://www.youtube.com/watch?v=1yR3F9hnwGU

Instructions for the Forum

- It is the moment to demonstrate knowledges relating to the risks that the use of mobile devices may involve.
- Access the section corresponding to this unit within the forum of the subject.
- The proposed exercise consists of analysing the functioning of any of the mentioned threats (or you can propose a new one as well):
  - Frauds with special pricing numbers (SMS or phone calls).
  - Theft of keys (of any kind) used in online services.
  - Installation of applications with aggressive advertisement (Adware).
  - Installation of applications for bitcoins mining.
- Remember to include the objective of the attackers, vulnerabilities that they take advantage of and the process followed to achieve the objectives.


OWASP Top Ten Mobile Risks

OWASP

- OWASP is an organisation aimed at enabling the creation, development, acquisition, operation and maintenance of applications that may be reliable.
- It was created in 2001 and it was registered as non-governmental organization in 2004.
- All the tools, documents and forums created by OWASP are accessible for people interested in improving applications security.
- Some of the most important OWASP's contributions are mentioned below:

OWASP Top Ten: summary of the 10 most critical vulnerabilities of software development.

OWASP *Testing Guide*: best practices guide for intrusion tests

OWASP *Development Guide*: secure programming guide.

Top Ten Mobile Risks

- OWASP Top Ten Mobile Risks is a specialised version of the OWASP Top Ten. It began in 2013 with a survey to experts.

The list includes the following vulnerabilities:

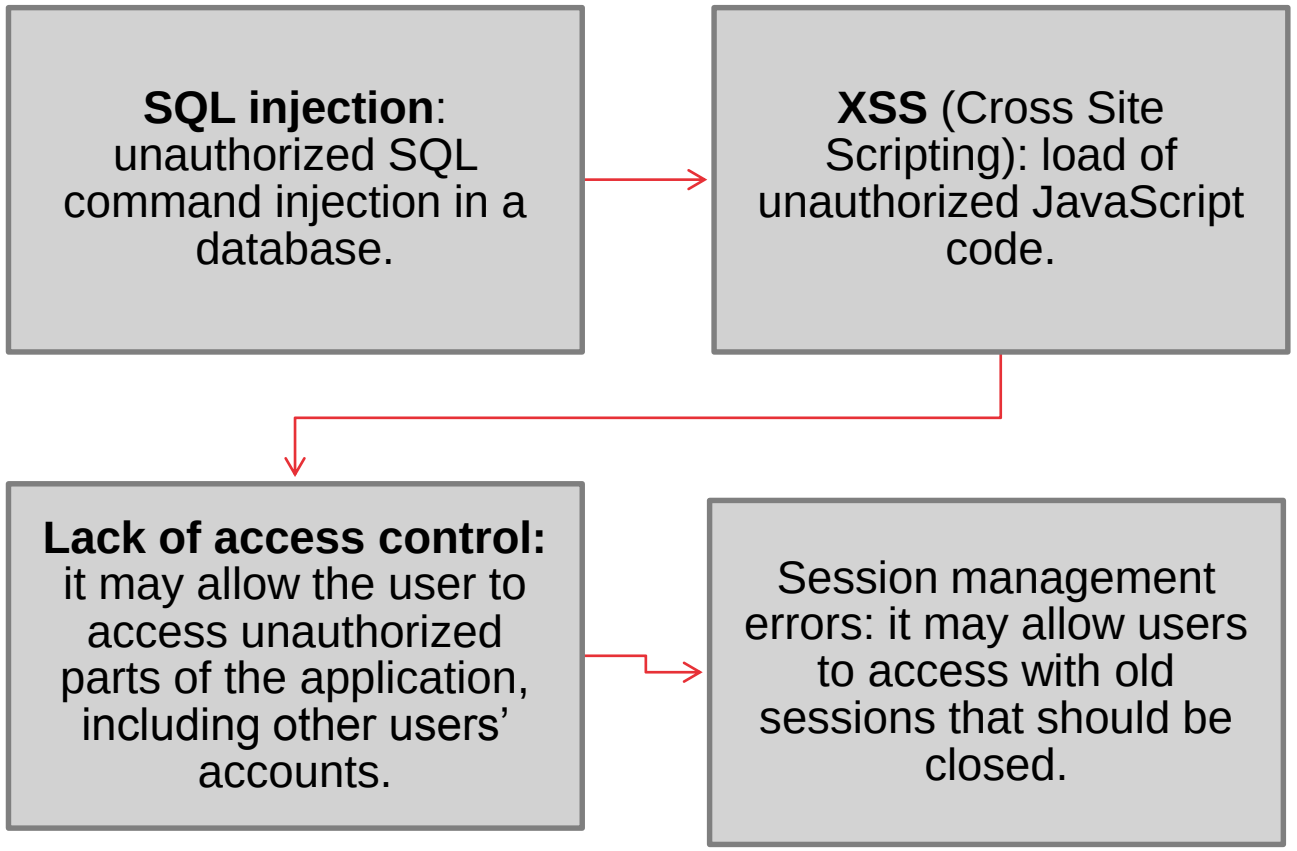
Weak Server Side Controls.
Insecure Data Storage.
Insufficient Transport Layer Protection.
Unintended Data Leakage.
Poor Authorization and Authentication.
Broken Cryptography.
Client Side Injection.
Security Decisions Via Untrusted Inputs.
Improper Session Handling.
Lack of Binary Protections.

Weak Server Side Controls

Attackers	Attack Vectors	Security Weakness		Technical Impacts
Undetermined	Exploitability EASY	Prevalence COMMON	Detectability AVERAGE	Impact SEVERE
It includes any agent able to create unreliable input data for the application: a user, a malware, a vulnerable application, etc.	Services that are remotely accessible and, in most occasions, they only need a previous registration. Data used in the registration may be fictitious.	The mobile application must access an API through a web service vulnerable to any weakness of the server (OWASP Top Ten). SQL injection or XSS are included among them.		Impact of the attacked server vulnerability. In the worst case, the impact of a given vulnerability in the server is severe. An SQL injection vulnerability could even lead to a disclosure of all users' access data and the administration of the whole site.

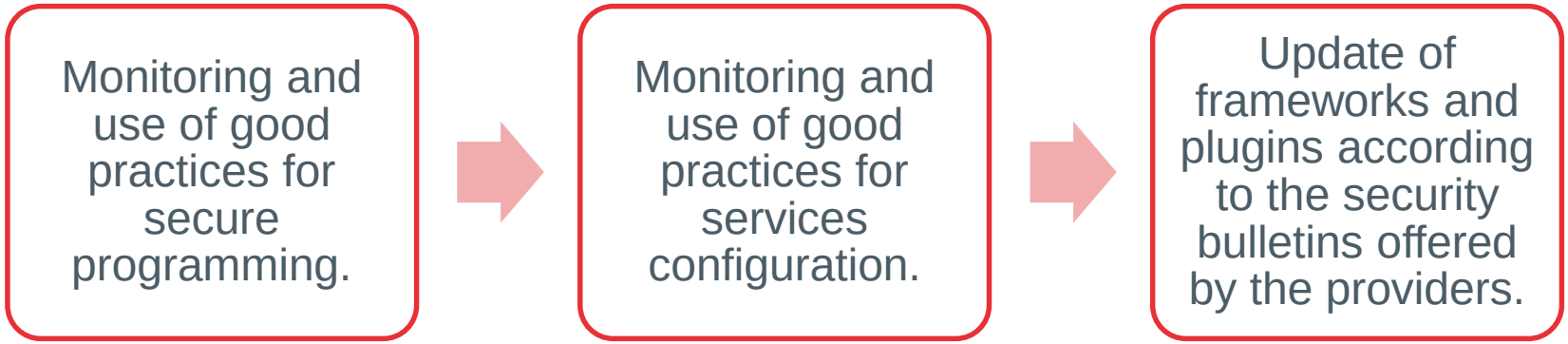
Weak Server Side Controls

- Examples of server vulnerabilities that can be exploited:



Weak Server Side Controls

- Attacks to this vulnerability may be avoided by:



Insecure Data Storage

Attackers	Attack Vectors	Security Weakness		Technical Impacts
Undetermined	Exploitability EASY	Prevalence COMMON	Detectability AVERAGE	Impact SEVERE
It includes any agent able to access the application: an attacker that has physical access, a malware, etc.	Having physical access to the device, the attacker can connect it to a computer and use multiple applications to access the data that it contains. Regarding malicious applications, they are able to access contents shared in the device.	It is quite easy for attackers or malicious applications to access file systems. If developers do not take it into consideration and consider storage unassailable, they will not implement the required mechanisms to protect such systems.		Such types of vulnerabilities may enable access to multiple sensitive data: - Users - Authentication Tokens - Cookies: - Location - Logs - Messages - …

Insecure Data Storage

- Some examples of insecure data storage vulnerabilities are illustrated below:

Non-encrypted credential and password storage in configuration files.

Password static encryption in the application code.

Not-deleted unnecessary data for the application.

Use of weak cryptographic libraries

Insecure Data Storage

- Attacks to this vulnerability may be avoided by:

Not storing data in shared locations of the system. In case that it is necessary, use a strong encryption scheme.

Disabling the option to copy the application to the device's SD card.

Not creating files within the sandbox with read permissions for other applications.

Using the API provided by the operative systems in order to add an additional encryption layer for files, as often as necessary.

Insufficient Transport Layer Protection

Attackers	Attack Vectors	Security Weakness		Technical Impacts
Undetermined	Exploitability DIFFICULT	Prevalence COMMON	Detectability EASY	Impact MODERATE
It includes any entity between the application and the server that is recipient of the information: telephone network, wireless devices in the same network, router, network intermediate points, etc.	It is necessary to have the possibility to capture traffic created by the victim. It is a simple task in some types of networks, but it is not in most of them, since it requires physical access to the infrastructure.	The user cannot verify whether a mobile application is using SSL or TLS connections. In many cases, this connection is only used for the authentication, but not for the rest of traffic. In order to detect it, it is enough to inspect the network traffic used by the device.		It may allow the attacker to access credentials or private information of a user of the service. This information may lead to identity theft attacks.

Insufficient Transport Layer Protection

- Some examples of insufficient transport layer protection that may be exploited are described below:

Absence of certificate verification:

- In this case, the application uses an SSL/TLS connection, but it does not properly verify values included in the certificate (validity, signature). Thus, the server identity is not completely verified and it may lead to “man in the middle” attacks through the use of SSL proxies.

Weak parameters negotiation:

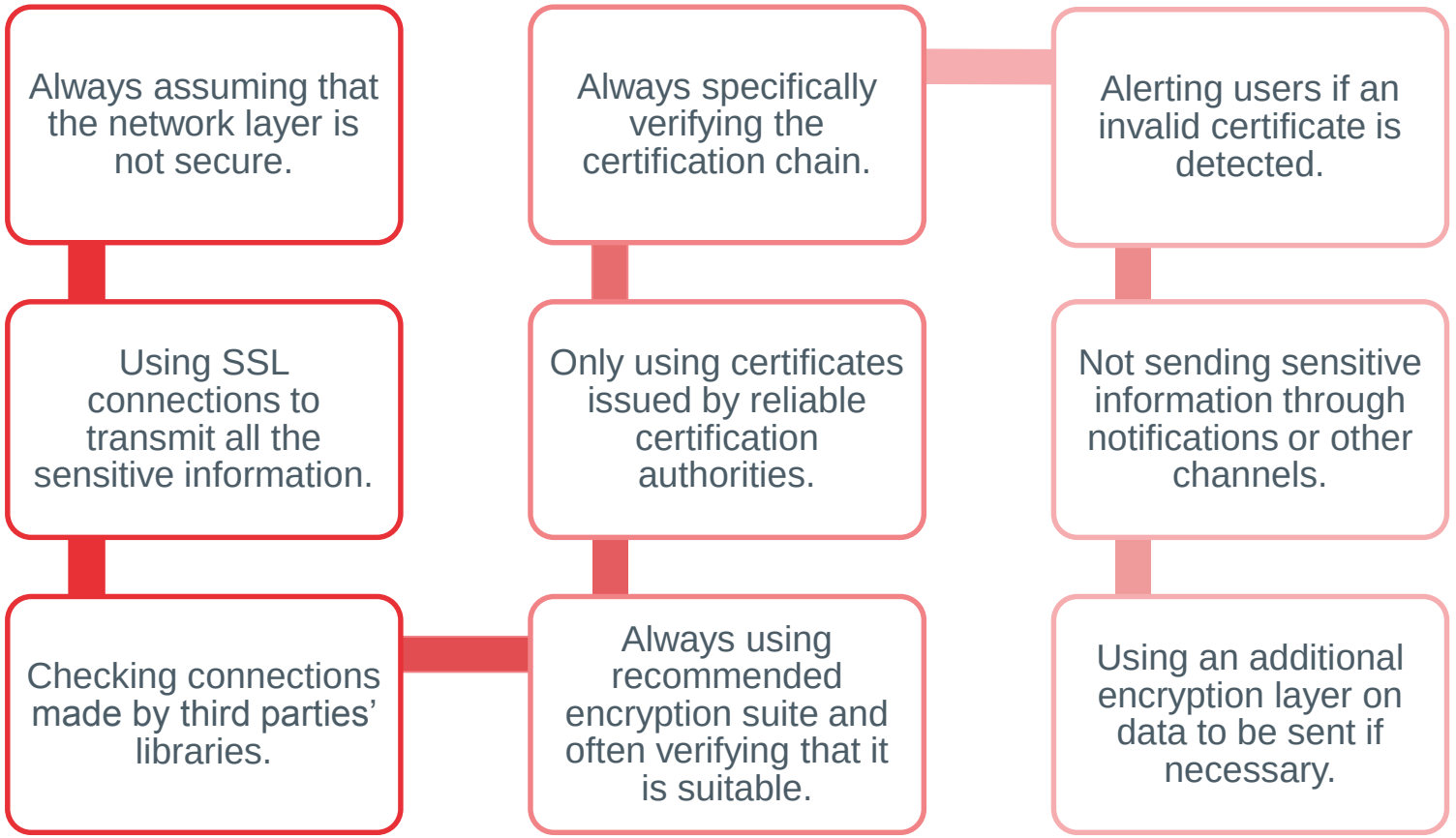
- The mobile application uses an SSL/TLS connection, but it has weak encryption parameters. It allows third parties to access the transmitted information.

Use of decrypted channels:

- The application uses non-encrypted protocols to communicate with the server. In some cases, they can be implemented by third parties’ libraries.

Insufficient Transport Layer Protection

- Attacks to this vulnerability may be avoided by:



Unintended Data Leakage

Attackers	Attack Vectors	Security Weakness		Technical Impacts
Undetermined	Exploitability EASY	Prevalence COMMON	Detectability AVERAGE	Impact SEVERE
It includes any agent able to access the application: an attacker that has physical access, a malware, etc.	Having physical access to the device, the attacker can connect it to a computer and use multiple applications to access the data that it contains. Regarding malicious applications, they are able to access contents shared in the device.	It takes place when a developer uses a library or API that creates insufficiently protected files, containing confidential information. Generally, if there is not a deep knowledge of the internal functioning of libraries, the developer is not aware of this fact.		Extraction of sensitive information by the attacker or other application that has access to data.

Unintended Data Leakage

Example of unintended data leakage:

- URL Caching (request and response).
- Words in keyboards' dictionaries and the keyboard's cache.
- Clipboard of the device.
- Images created for background applications.
- Logs.
- HTML5 data storage.
- Cookies.
- Analysis data collected by third parties' libraries.

Unintended Data Leakage

Attacks to this vulnerability may be avoided by:

It is necessary to carry out a risk analysis of the operative system itself and libraries used by the application to be developed.

Verify actions that are performed by default regarding data mentioned before.

Implement countermeasures when one of the previously mentioned data is sensitive within the application.

In the case of banking applications, for example, when the application is moved to the background, overlay an image so that the screenshot made by the operative system does not include confidential information.

Poor Authorization and Authentication

Attackers	Attack Vectors	Security Weakness		Technical Impacts
Undetermined	Exploitability EASY	Prevalence COMMON	Detectability AVERAGE	Impact SEVERE
It includes agents that have access to the device and are able to use tools for the automatization of start-up tasks.	Once the vulnerability of the authentication mechanism is discovered, the attacker can use tools to avoid this protection measure and easily obtain access.	Such type of attack allows the attacker to access the victim's credentials and execute actions anonymously and hijacking the user's identity. In some cases, the system may be vulnerable due to the weakness of the scheme implemented in the system (see M1).		When the application suffers such attack, it is not possible to know whether the actions that it executes were requested by the real user or not. It is difficult to detect actions that are executed by a properly authenticated user.

Poor Authorization and Authentication

- Examples of poor authorization:

APIs accessed from the device once the authentication has been made, are not properly authenticated by the server. The developer assumes that, since the API has been executed after the user has been authenticated within the application, authorisation tokens are not necessary to execute them. An attacker can take advantage of this event to use such API in a fraudulent way.

The application allows the utilization of weak passwords. A 4-digits numeric password is quite easy to obtain via brute force attacks; even if it has been stored after executing an overview function on it. This information can be obtained if the attacker is able to access the server's confidential information or if the password is locally stored by the application.

Poor Authorization and Authentication

- Attacks to this vulnerability may be avoided by:
- It should be assumed that no authentication or authorisation control carried out in the device is reliable.
- Whenever possible, authorisation and authentication controls should be performed by the server.
- In case that the application has no access to the Internet or a server:

Encrypt the information required by the application via libraries existing in the operative system itself that use biometric information or the lock code.

Security controls that can be avoided with the modification of the application's code should not be implemented. Make the application progress exclusively depend on information encrypted through the mentioned methods.

Add integrity controls to the application itself so that it is not executed if any code modification is detected.

Broken Cryptography

Attackers	Attack Vectors	Security Weakness		Technical Impacts
Undetermined	Exploitability EASY	Prevalence COMMON	Detectability EASY	Impact SEVERE
It includes any agent able to access the application: an attacker that has physical access, a malware, etc.	Data decryption may be carried out via physical access to the device or by monitoring its network connections.	Once the encrypted data has been obtained, the attacker takes advantage of weak points in the encryption to obtain the decrypted version of data.		This vulnerability may lead to a leakage of multiple sensitive data of a mobile device.

Broken Cryptography - Examples

- Examples of broken cryptography:
- Since 2015, the Internet Engineering Task Force (IETF) forbade the use of RC4 encryption in any TLS connection (see RFC 7465):

This was also recommended by Microsoft and Mozilla.
RC4 was also used in WEP (wireless networks encryption standard that became insecure due to a problem with initialization vectors).

- The MD5 algorithm is considered broken by the community since 1996:

It is possible to create two groups of different data that obtain the same summary value.
It allows the creation of fraudulent signatures, among other options
In 2015, this algorithm was still widely used in the network.

Broken Cryptography

- Attacks to this vulnerability may be avoided by:
  - Selecting robust and complex encryption keys.
  - Not storing encryption keys in the code or in files within the application.
  - Check regularly recommendations on the use of cryptographic algorithms (threat Intelligence services and security bulletins).

Client Side Injection

Attackers	Attack Vectors	Security Weakness		Technical Impacts
Undetermined	Exploitability EASY	Prevalence COMMON	Detectability AVERAGE	Impact SEVERE
It includes any user that can send data to different entry points of the application: both internal and external users, other applications, etc.	The attacker sends malicious entry data through the entry interfaces that the application generally provides to users. Any source of data may be useful for the code injection.	The vulnerable application executes malicious code loaded via one of the attack vectors. Unless the attack is a privilege escalation exploit, its surface is limited to the application itself and the data included.		The impact may vary depending on the type of application and the vector attack used. An SQL injection attack may provide credentials and private information. Other attacks able to obtain privilege escalation, may access the whole device.

Client Side Injection

- Some of the examples of client side injection vulnerabilities are described below:

SQL injection: the application receives modified SQL data from the server. When executing the received request, entries with false information are injected in the application's database.

Cross Application Scripting Attacks: a malicious application sends a modified message to another application to cause a failure and execute arbitrary code on its data.

XSS: an HTML-based application may be modified in order to redirect the user to a malicious website that collects one's personal data.

Client Side Injection

- Attacks to this vulnerability may be avoided by:
- Generally, it is necessary to list all the points used by an application to receive data and to ensure that incoming data is always validated.
- The most common ones are described below:

SQL injection:	Use API provided by the operative systems (parametrized queries, Core Data or Content Providers).
XSS:	Ensure that the web version of all the applications validate the received data.
File Inclusion:	Use file validation libraries.
String of characters and XML:	Use XML data validation libraries and secure functions to handle character strings.
Communication between apps:	Check that all acceptable data are processed by the application and remove the rest.

Security Decision Via Untrusted Inputs

Attackers	Attack Vectors	Security Weakness		Technical Impacts
Undetermined	Exploitability EASY	Prevalence COMMON	Detectability AVERAGE	Impact SEVERE
Any application of the operative system, able to send messages to other applications existing in the system.	Any entry point for the reception of data from other applications.	The attacker may perform reverse engineering of the parameters received by the attacked application and make it perform actions that were not initially planned to be executed.		The attacker's application obtains permissions since it is able to perform some actions that the victim could perform before. For example: access to confidential information or use of protected resources.

Security Decision Via Untrusted Inputs

- The following examples illustrate possible security decision via untrusted inputs:

Use of services on Android: a conversation recorder application uses a service to record conversations. The use of the microphone service has been authorised. Such service is not protected by permissions and is exported, so it enables access to other applications. A malicious application uses the service to obtain confidential information.

→

Activity Hijacking: a malicious application may login to respond to intents, initially aimed at other applications. In such case, instead of displaying the expected application's interface, the one belonging to attacker's application is displayed being the user unaware.

Security Decision Via Untrusted Inputs

- Depending on the type of resource that accesses the application and the communication mechanisms that it has, different prevention measures may be implemented:

In case that a sensitive resource is exposed between applications, it may be necessary to request a specific permission for applications that are going to communicate with ours.

Check the entry parameters that the application receives form other applications.

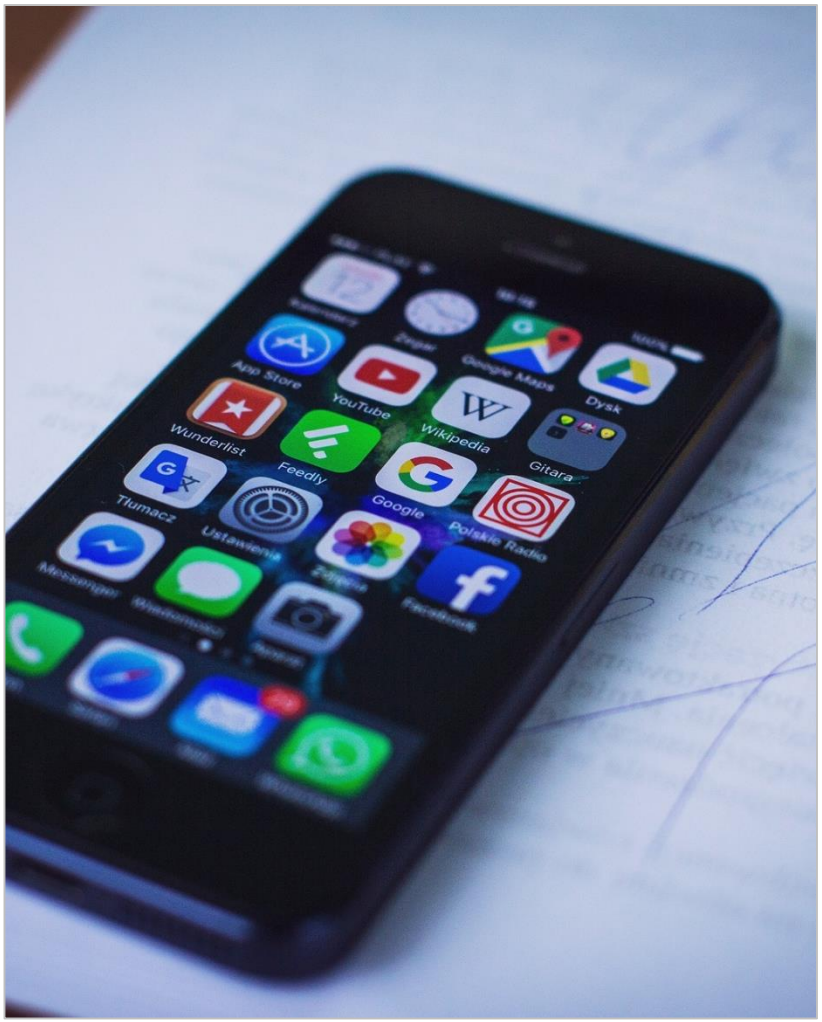
Use explicit communications between the components of the application so that other applications cannot login to receive messages destined to our application.

Improper Session Handling

Attackers	Attack Vectors	Security Weakness		Technical Impacts
Undetermined	Exploitability EASY	Prevalence COMMON	Detectability AVERAGE	Impact SEVERE
Any user with access to the HTTP/S communications of the application.	Any user with physical access to the device or the network used to access the services.	In order to match all the requests from a client to a server, a series or cookies or tokens are created during the first connection to that server. Such tokens can be used in future messages to match messages sent to the user. If they are not well-managed, and are accessed by attackers, requests may be sent hijacking the victim's identity.		The attacker with access to session tokens may perform actions as if it was the victim itself. The limitation of the attack will be the same as the ones established by the specific application for the victim.

Improper Session Handling

- Some examples of improper session handling are described below:
  - Banking application whose sessions do not expire due to inactivity: the vulnerable application does not log off after a downtime. In case that the mobile device is stolen or lost, the attacker may access the banking account of the victim.
  - Application that does not use an SSL protocol in communications and does not invalidate session tokens in the server: in this case, a third party with access to the network may capture tokens sent during the communication process with the server and use them at any time in the future, even if the user has logged out in the mobile device.



Improper Session Handling

- The following prevention measures can be implemented for this type of vulnerability, among others:

Establish waiting times to log off according to the type of security required:	Update session tokens after every change of the session status:	Obtain tokens using random number generators. Such tokens should be long enough and have a high quality.
• 5 minutes for applications that require a high level of security. • 30 minutes for applications that require a medium level of security. • 1 hour for the rest of applications.	• Changes from anonymous to registered, from a user to a different one, etc. • This way, it is possible to detect the reutilisation of old sessions.	

Lack of Binary Protections

Attackers	Attack Vectors	Security Weakness		Technical Impacts
Undetermined	Exploitability MEDIUM	Prevalence COMMON	Detectability AVERAGE	Impact SEVERE
A person that has access to the device and uses debugger or reverse engineering tools.	The attacker may use an automatic tool in order to obtain the original code of the application. There are automatic tools as well, to turn benign applications into pieces of malware.	Even if binary protection measures are applied, a persistent attacker with enough technical knowledge would be able to perform a reverse engineering attack on the code of the application. In order to avoid such attacks, it is possible to add verification code that can be removed by the tools themselves.		If an attacker obtains source code of the application, intellectual property of the company may be revealed. The binary modification during its execution may be used to reduce security controls existing within the application.

Lack of Binary Protections

- Lack of Binary Protections:
  - Any static or dynamic analysis of an application tries to take advantage of this vulnerability in order to gain knowledge regarding the application.
  - In some cases, it is not performed to illegally obtain intellectual property, but to analyse actions performed by a malicious application.
  - The security analysis of an internal application requires the performing of this type of attacks.
  - Some common tools to perform static and dynamic analysis are mentioned below:

Androguard for Android.

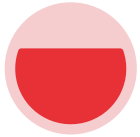
IDA Pro and Hooper to perform reverse engineering and code debugging.

Lack of Binary Protections

- It is possible to implement certain controls, but in some cases, the tools themselves (code debuggers or repacking) may be used to avoid them:

Jailbreak or rooting detection:

It detects whether the device can execute unsigned code. Devices with jailbreak, can execute code debuggers with no restrictions.



Integrity controls:

Check that class files have not been modified before executing certain parts of the application.

Code debugger detection:

The application may detect whether there is a code debugger inspecting its code and act accordingly.


Detailed Study of Attacks Threats,
Motivations and Impact

Real Cases

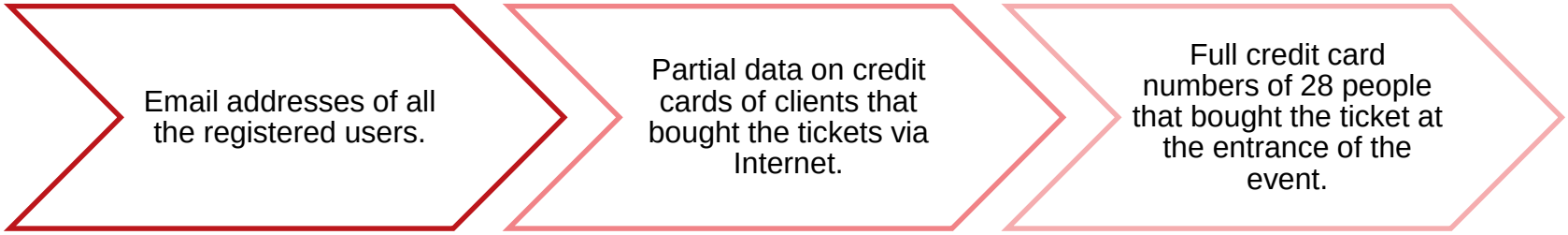
- Once the 10 main vulnerability that may affect mobile applications have been described, the 5 real cases mentioned at the beginning of the chapter will be more in-depth analysed:
  - Evenbrite
  - Moonpig
  - UPnP
  - Car Parking
  - Banking
- The following characteristics will be described in each real case:
  - Top 10 vulnerabilities involved.
  - How attackers have perform the attack or could have taken advantage of vulnerabilities.
  - Technical and business consequences that the attack had.
  - Measures that could have been taken to avoid the attack.


Eventbrite

Eventbrite

Overview

- Information available at:
- http://www.eventbrite.com/blog/our-commitment/
- Eventbrite is an event management company.
- An employee lost two iPads when leaving an event.
- Lost data included:



Top Ten OWASP Mobile

- The main vulnerability involved within this case is: **Insecure Data Storage**
  - The loss of the device is not a vulnerability itself.
  - Data were stored in an internal application developed by Eventbrite, called “Eventbrite at Door”.
  - Data stored within the application were not properly protected and were stored non-encrypted in the data of the application itself.
  - The product was at the earliest stages of development and not all the possible security measures were implemented yet.

Vulnerability exploitation

- An attacker with access to the stolen iPads could have accessed data by using the procedure described below.
- In case that the iPad was protected **with a password:**

1 Perform jailbreak or try a brute force tool to discover the code.

2 Install a tool to access the file system of the device, such as iFunBox or an SSH server.

- In case that the iPad was **not** protected **with a password:**
  - Download a tool such as iFunBox and access the application data.
- It is important to mention that the attacker could also have accessed further data stored in other vulnerable applications:
  - Access credentials to services.
  - Emails, etc.

Consequences

Technical consequences:

- Third parties' banking data exposed.
- Other applications' data and credentials exposed.

Consequences for the business:

- Loss of reputation.
- Possible fines related to the improper personal data processing.

Incident response by Evenbrite:

- Report the theft to all the affected users.
- Report the situation to the police.
- Perform the remote block and wipe of the affected devices.
- Implement measures to avoid this event being repeated:
  - Improve the application's security.
  - Remove sensitive information when the iPads leave the organization.

Specific Protection Measures

Such measures include the following events:

Configuring all the iPads with a strong access password.

All files containing sensitive information and created by the application should be encrypted with keys derived from the access password to the iPad.

Remove all data included in the application when the iPad is going to be physically moved:

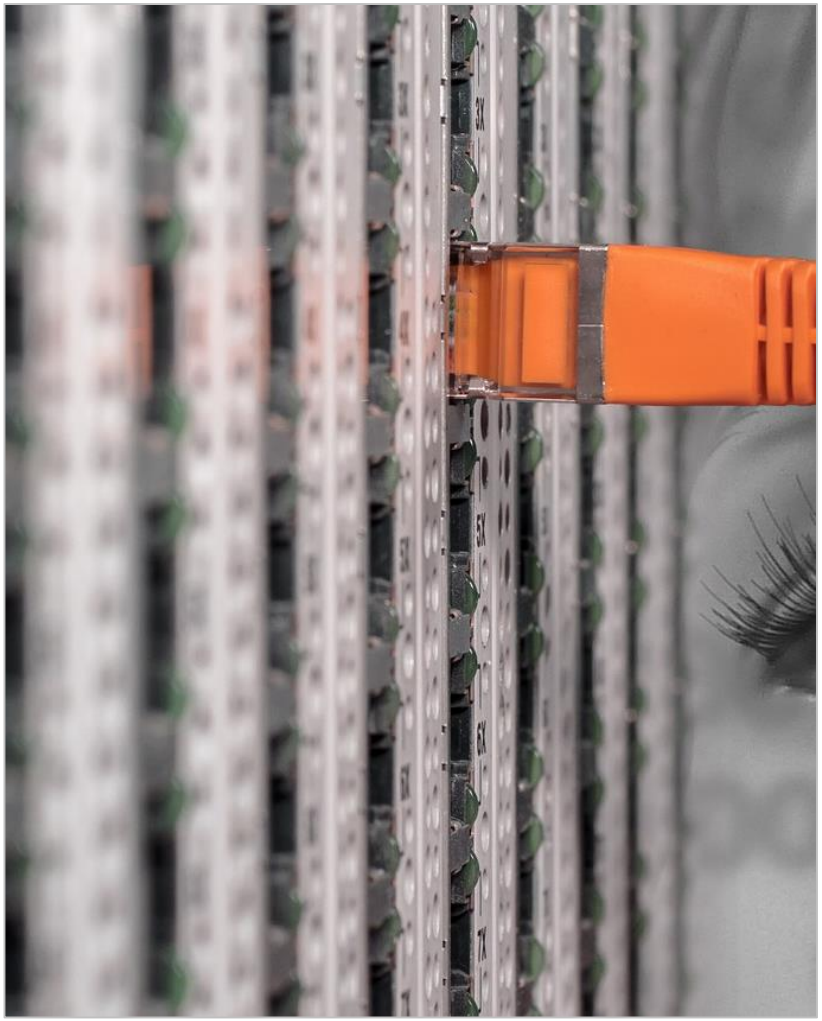
The security of the transmission of data to the servers should be verified.

Check the existence of cache files.


Moonpig

Top Ten OWASP Mobile

- The main vulnerability involved within this case is: **Weak Server Side Controls.**
  - The back-end of the mobile application had no access control mechanisms in order to access the users' personal information.
  - Even if the failure was discovered by analysing the mobile application, all the users of the website were affected by this problem.
  - It is essential to provide access control for all the entry points to the application, regardless the access platform.
  - It is unknown whether there was any user affected by the detected vulnerability.



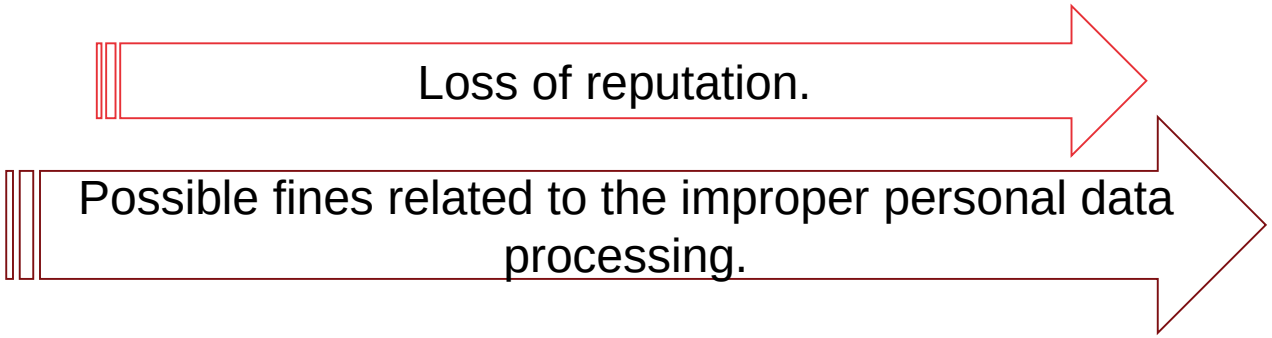
Vulnerability Exploitation

- The attacker analyses the web server using the mobile application as information source, to obtain the structure of calls to the back-end's API.
- Then, identifies that requests to access personal information do not include any credentials.
- The attacker uses a proxy or a HTTP request generator to create a group of random requests.
- And then, executes a script to obtain data of the clients of the service. Such data include:

1. User name and email.
2. Last digits of the credit card.
3. Addresses and recent mailings.

Consequences

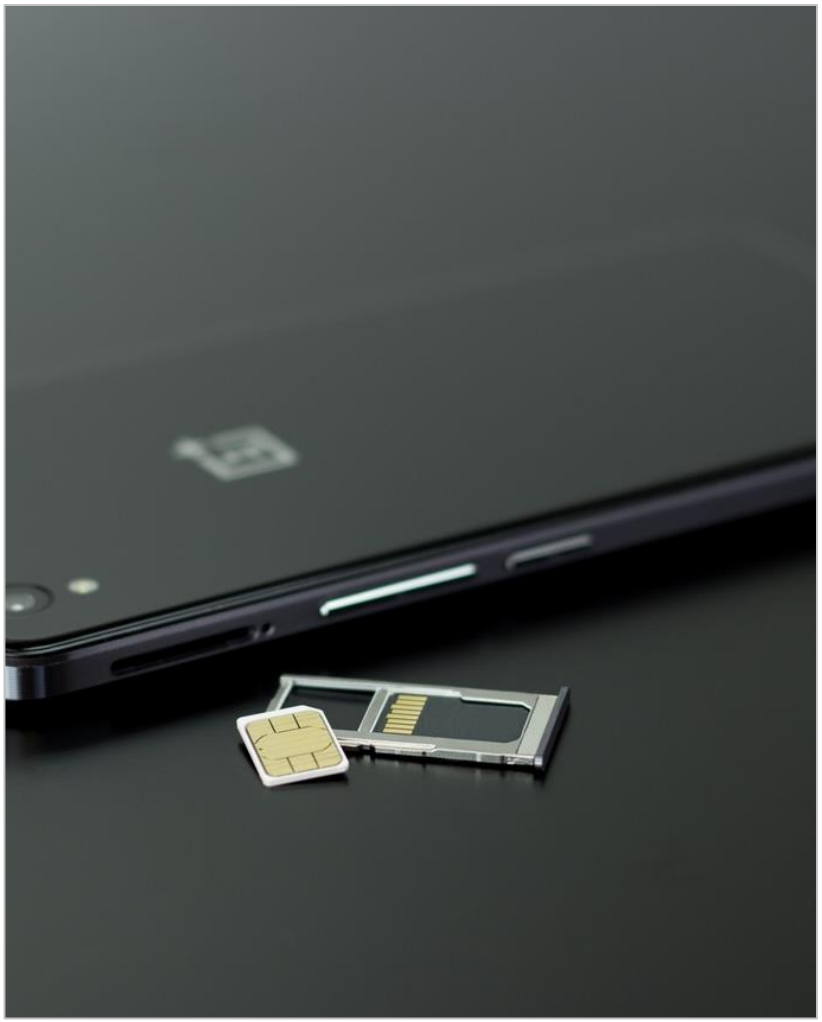
- Technical consequences: sensitive data related to the users of the service exposed.
- Consequences for the business:



- Incident response by Evenbrite:
  - The discoverer of the vulnerability was informed that the failure would be fixed in 6 months.
  - It took 18 months to fix the failure and users of the application were not informed.

Specific Protection Measures

- Measures to implement would include:
  - Appropriate access control measures in all the components of the application that include access to operations or sensitive information.
  - Integrate one of the secure development of application methodologies to the applications development methodology.
  - Perform analysis and penetration test tasks, not only in the mobile application but also in the application's back-end equally.




Vulnerability in the UPnP Library

Vulnerability in the UPnP Library

Overview

- Further information available at:

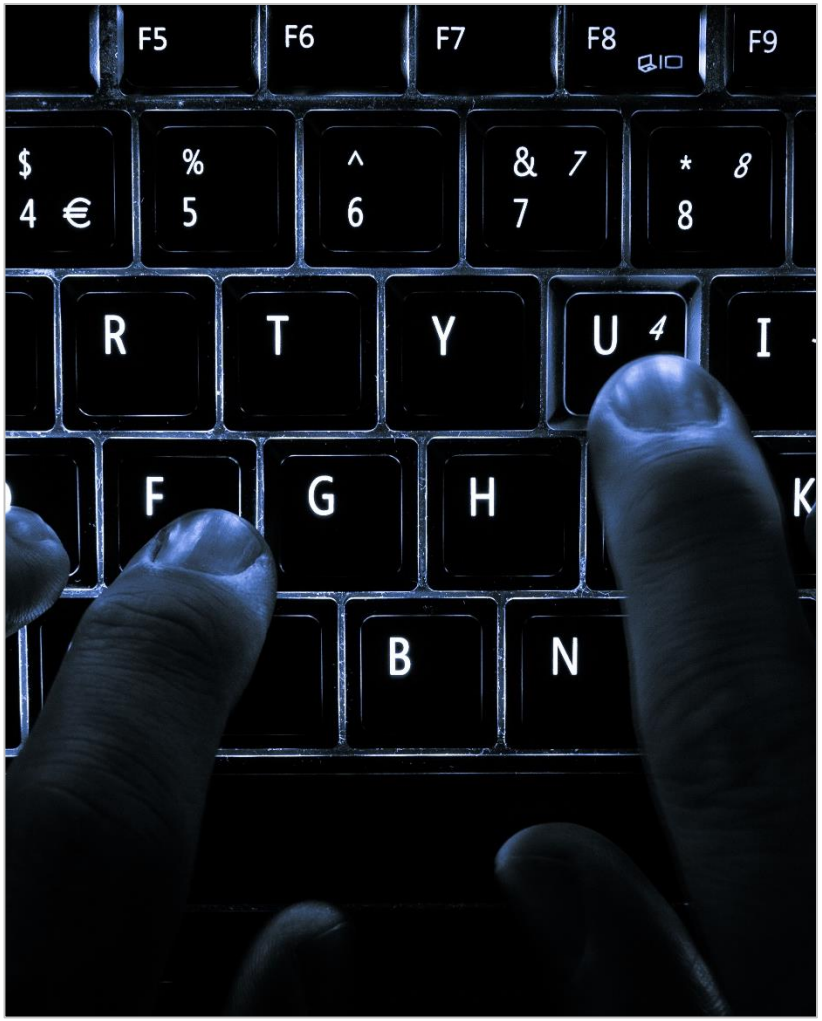
  http://blog.trendmicro.com/trendlabs-security-intelligence/high-profile-mobile-apps-at-risk-due-to-three-year-old-vulnerability/
  https://www.cvedetails.com/cve/CVE-2012-5965/

- UPnP libraries are used for video streaming between devices.
- The library is written in C, not in Java, that is why it is integrated in apps via the Android NDK.
- There are more than 6 million devices that use such libraries in their devices: TV, smartphones, etc.
- It is not included in the operative system, applications should include the library in order to use it.
- The vulnerability enabled the execution of random code in a device that had one of those applications installed.

Vulnerability in the UPnP Library

Top Ten OWASP Mobile

- The main vulnerability involved within this case is: **Client Side Injection.**
  - The verification of entry data should not only be performed for the code developed by us, but also for third parties' libraries.
  - In the case of devices with no preventive measures to avoid privilege escalation, this vulnerability can be useful to execute any kind of random code and take control of the device.
  - In the case of Android or other mobile devices, the vulnerability only affects the context of the application itself.



Vulnerability Exploitation

- The vulnerability is found due to the way that the affected library handle SSDP protocol (Simple Service Discovery Protocol) packets.
- This protocol is part of the UPnP (Universal Plug and Play) standard, aimed at video data streaming among other functions.
- It is possible to create a buffer overflow in the C library, execute random code and take control of the application or device by using the library.

– CVSS Scores & Vulnerability Types

CVSS Score	**10.0**
Confidentiality Impact	Complete (There is total information disclosure, resulting in all system files being revealed.)
Integrity Impact	Complete (There is a total compromise of system integrity. There is a complete loss of system protection, resulting in the entire system being compromised.)
Availability Impact	Complete (There is a total shutdown of the affected resource. The attacker can render the resource completely unavailable.)
Access Complexity	Low (Specialized access conditions or extenuating circumstances do not exist. Very little knowledge or skill is required to exploit.)
Authentication	Not required (Authentication is not required to exploit the vulnerability.)
Gained Access	None
Vulnerability Type(s)	Execute Code Overflow
CWE ID	119

Consequences

- Consequences are different according to the type of device.
- Devices with no mitigation measures facing a buffer overflow:

1	Total control of the device or the application. It depends on whether the application is being executed in a closed environment.
2	Access any piece of information stored in the device or application.

- The discoverer of the vulnerability did not verify whether it was possible to execute random code in devices that included mitigation measures against buffer overflow.

Specific Protection Measures

- The only specific protection measure possible would consist of patching the vulnerability.
- Since this vulnerability is used by multiple applications, it is necessary to update all the applications that use the library (problems with applications that are not updated anymore).

Car Parking Applications

Car Parking Applications

Overview

- Further information available at:

  http://www.theregister.co.uk/2015/12/11/mobile_parking_apps_audit/

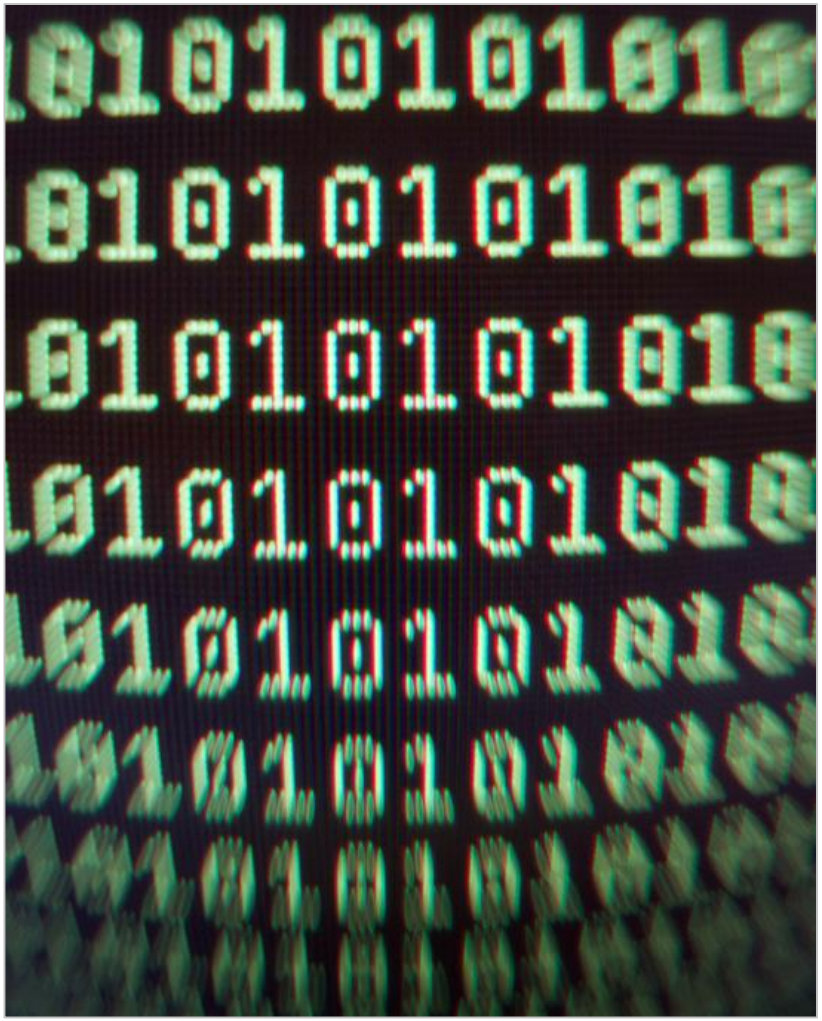
- Audit conducted on 6 applications aimed at paying the car parking in United Kingdom. All applications analysed on Android version.

Results:

- All the applications used SSL connections, but none of them verified the certificate received by the server.
- One of the applications used its own encryption suite, with keys permanently stored in the device itself.
- Applications stored credentials in non-encrypted files.
- One of the applications used a **WebView** (application provided to developers to launch a browser integrated in the application), vulnerable to code injection aimed at displaying content.

Top Ten OWASP Mobile

- Applications analysed by NCC were affected by various vulnerabilities included in OWASP Top Ten:
  - Insecure Data Storage.
  - Insufficient Transport Layer Protection.
  - Unintended Data Leakage.
  - Poor Authorization and Authentication.
  - Broken Cryptography.
  - Client Side Injection.
  - Lack of Binary Protections.



Vulnerability Exploitation

- There are multiple ways for an attacker to conduct an attack on applications.
- The easiest way does not require physical access to the device and takes advantage of the insufficient transport layer protection:

The attacker, who controls the Wireless network, creates a proxy and an SSL certificate.

The traffic is redirected from the affected domain to the attacker's SSL proxy.

The mobile application accepts the certificate, since it does not verify the origin.

The attacker may intercept and modify all the traffic between the application and its server.

Consequences

- The number of vulnerabilities that affect this application is specially high.
- Technical consequences:
  - Application's credentials exposed.
  - Economic data were transmitted non-encrypted, therefore, exposed.
  - Possible economic losses for clients.
  - Credentials used in other applications may pose a risk for other services in which the client uses the same credentials.
- Consequences for the business:
  - Loss of reputation for the application's creators.
  - Possible fines related to the improper personal data processing.

Specific Protection Measures

- Multiple measures can be implemented to solve the great amount of issues that affect such applications, the most relevant ones are described below:

Proper implementation of connections performed via SSL:

- Validation of certificates.

Secure storage of credentials within the device:

- Avoid the use of non-encrypted files.
- Use mechanisms to avoid other applications or processes to access data.

Disable apps installation in the card.


Banking Applications

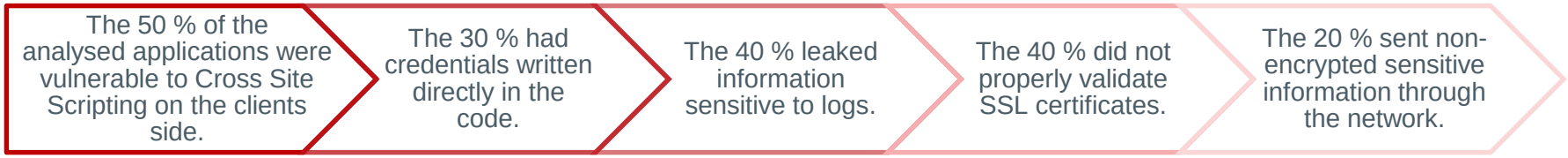
Banking Applications

Overview

- Further information available at:

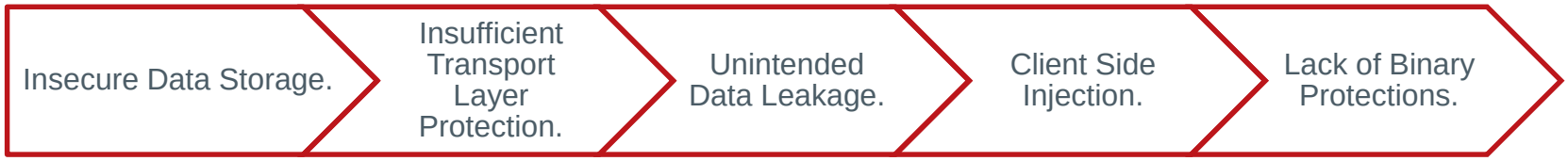
http://blog.ioactive.com/2014/01/personal-banking-apps-leak-info-through.html

- Analysis of 40 iOS banking applications.
- 40-hours analysis performed by an expert.
- Noteworthy results:



Top Ten OWASP Mobile

- As in car parking applications, the whole of banking applications is affected by the following vulnerabilities included in OWASP Top Ten:



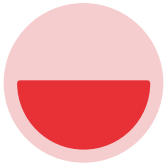
- In some cases, applications store data on the internal infrastructure of the bank:
  - Internal severs IP addresses.
  - Access credentials to certain servers.
- In this case, as this application is aimed at managing banking data, the situation is far more critical.

Vulnerability Exploitation

- There are multiple ways of exploiting the vulnerabilities found by the researchers:
  - Use information of the organization's internal servers to conduct an attack on the bank's structure itself.
- Create a proxy server to:
  - Intercept authentication messages sent from the server to the application.
  - Conduct a "man-in-the-middle" attack to SSL communications by using an own certificate.
- Inject malicious code in any of the web views used by the applications to steal users' credentials.

Consequences

- The number of vulnerabilities that affect this application is specially high.



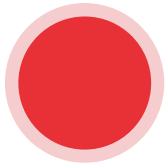
Technical consequences:

Application's credentials exposed.

Clients' economic data related to the bank account exposed.

Possible economic losses for clients.

Credentials used in other applications may pose a risk for other services in which the client uses the same credentials.



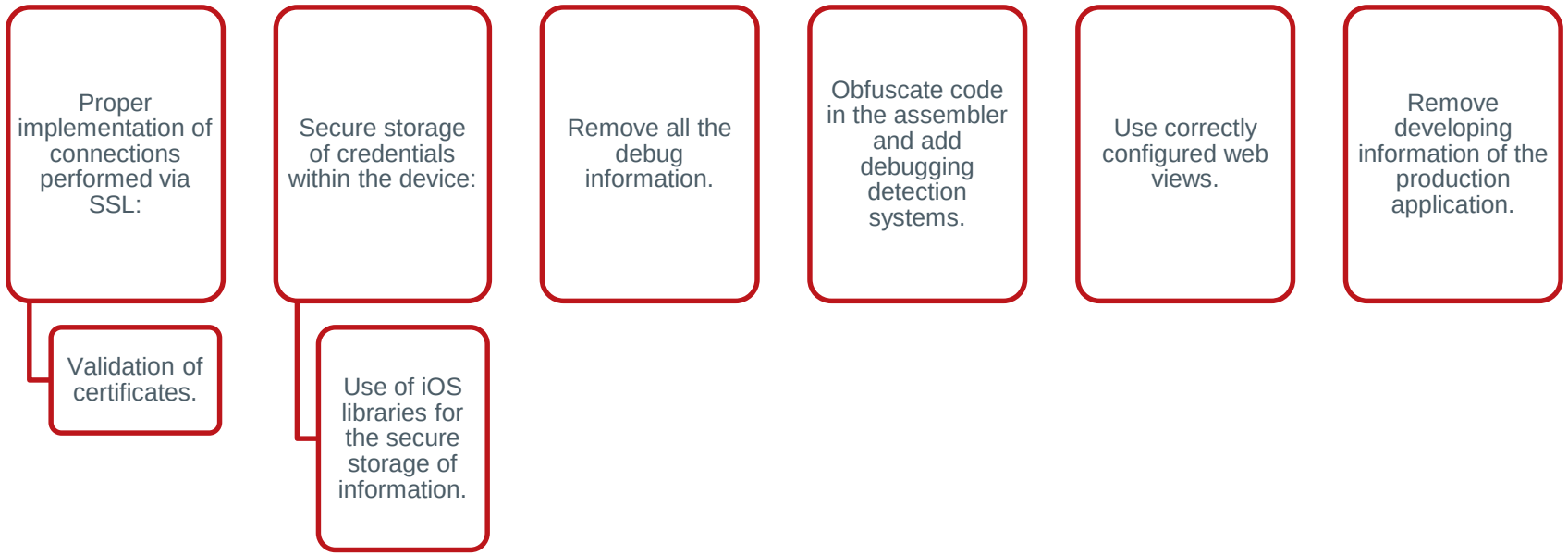
Consequences for the business:

Loss of reputation for the application's creators.

Fines due to non-compliance with regulations on banking and personal data processing.

Specific Protection Measures

- Multiple measures can be implemented to solve the great amount of issues that affect such applications, the most relevant ones are described below:



Instructions for the Forum

- It is the moment to demonstrate, by writing in the forum, that knowledges related to vulnerabilities that affect mobile applications and the risks that the use of mobile devices may involve have been acquired.
- The proposed exercise consists of analysing risks on one of the following situations:
  - Application with pay content that verifies client access verifications.
  - Messaging application that transmits data via non-encrypted connections.
  - Problems described in the following links:
    - http://www.pcworld.com/article/2018187/instagram-vulnerability-on-iphone-allows-for-account-takeover.html
    - http://www.scmagazine.com/researchers-discover-vulnerability-in-ios-app-allowing-malicious-file-attack/article/441874/
- The following elements should be described in the analysis:
  - Top Ten vulnerabilities that affect the situation.
  - Assets within the affected telephone/application.
  - Attackers and attack vectors (including the purpose).
  - Technical and business consequences of the attack.
  - Mitigation measures of the attack.