

UNIVERSIDAD POLITÉCNICA DE MADRID



**MÁSTER UNIVERSITARIO EN
CIBERSEGURIDAD**

TRABAJO DE FIN DE MÁSTER

***Development of a system for traffic analysis of
smartphone apps for private data exfiltration
detection***

Lucas María-Tomé Laverón

2019

Trabajo Fin de Máster

TÍTULO: Development of a system for traffic analysis of smartphone apps for private data exfiltration detection

AUTOR: Lucas María-Tomé Laverón

TUTOR: José María del Álamo Ramiro

Escuela Técnica Superior de Ingenieros de Telecomunicación

Escuela Técnica Superior de Ingeniería y Sistemas de Telecomunicación

Escuela Técnica Superior de Ingenieros Informáticos

Escuela Técnica Superior de Ingeniería de Sistemas Informáticos

TRIBUNAL:

PRESIDENTE:

VOCAL:

SECRETARIO:

FECHA DE LECTURA:

CALIFICACIÓN:

Fdo: El Secretario del Tribunal

Acknowledgements

To Chema, Danny and Espe for this great year.

Resumen

Con el crecimiento en el uso de los sistemas inteligentes cada vez más información es recolectada y analizada. Nuevas compañías han sido creadas sobre la base de la recolección de la información y el seguimiento de los usuarios a través de los distintos servicios que utilizan. Como respuesta, en el interés de defender los derechos de dichos usuarios, se han establecido una serie de leyes y regulaciones, entre las que ha destacado la Reglamento General de Protección de Datos (GDPR).

Dentro del cambio de marco de la protección de datos en la Unión Europea es necesario analizar una gran serie de servicios y aplicaciones para determinar su impacto en la privacidad de los usuarios. En concreto las aplicaciones de teléfonos móviles son especialmente relevantes dado su amplio uso y la cantidad de datos sensibles con los que tratan.

Existen tres principales enfoques a la hora de analizar una aplicación. Análisis estático, en el cual se obtiene información sin que sea necesario ejecutarla. Análisis dinámico, en el cual se obtiene información en base a los comportamientos generados durante la ejecución. Y análisis de tráfico, en el cual se obtiene información en base a las conexiones realizadas con entidades externas.

En este trabajo se analiza una serie de aplicaciones Android mediante el enfoque de análisis de tráfico y se desarrolla un sistema para la automatización de este proceso. El objetivo es detectar exfiltraciones de información sensible para evaluar el riesgo que supone el uso de cada aplicación.

Abstract

With the rise in the usage of intelligent systems more private information is being collected for analysis. New companies have sprung up whose business model relies on tracking and collecting as much data as possible from users. In response, in the interest of safeguarding the rights of users, new regulations have been established to control the recollection and usage of private information, most recently the General Data Protection Regulation (GDPR).

It is necessary to analyze any number of services that could be collecting sensitive information in order to assess whether those services follow the most recent laws and regulations. From the multiple and varied services one could analyze, smartphone applications stand out because of the widespread usage and the sensitive information with which they operate.

There are three main approaches to the analysis of any program. Static analysis obtains information about the program without the need to execute it. Dynamic analysis obtains information by executing the program and observing the behaviours that it produces. Finally, traffic analysis obtains information by detecting and processing the connections that the program creates to external entities.

In this thesis will analyze a set of Android applications following the traffic analysis approach, and a system will be developed to automate the process. The objective is to detect and identify data exfiltration to evaluate the risk that an app can pose to its users. It will be necessary to implement traffic interception, encryption stripping, and data analysis to this end. Finally the system will be tested against a set of publicly available Android apps.

Contents

1	Introduction	1
1.1	Introduction	1
1.2	Objectives	2
1.3	Methodology	3
2	State of the Art	4
2.1	Introduction	4
2.2	Execution	5
2.2.1	Platform	5
2.2.2	App execution	6
2.3	Interception and capture	7
2.3.1	Interception	8
2.3.2	Capture	9
2.4	Analysis	9
2.4.1	TLS bypass	10
2.4.2	HTTP sections	11
2.5	Selection	12
2.5.1	Platform	12
2.5.2	App Execution	13
2.5.3	Interception	13
2.5.4	Capture	14
2.5.5	TLS bypass	14
3	Design	15
3.1	Tools	15
3.2	Platform setup	17

3.2.1	Rooting the phone	18
3.3	Network	20
3.3.1	Virtual Network	21
3.3.2	Physical Network	21
3.4	System design	22
3.4.1	Traffic capture module	22
3.4.2	Analysis module	24
3.5	Implementation	24
3.5.1	Traffic capture module	24
3.5.2	Analysis module	24
3.5.3	Control server	25
3.5.4	Docker	25
3.6	Developed apps	25
3.6.1	DeviceData	26
3.6.2	DirectHTTPLeak1	26
3.7	Target applications	26
4	Results	29
4.1	Tests	29
4.2	Final analysis	30
4.3	Limitations	38
5	User Manual	39
5.1	Installation	39
5.2	Execution	39
5.3	Application analysis	40
6	Conclusions	43
6.1	Conclusions	43
6.2	Future work	44
A	Result data	45
B	Best practices	74

List of Figures

3.1	Xiaomi unlock tool success. Obtained from [28]	19
3.2	Booted into TWRP. Obtained from [29]	20
3.3	Virtual network	21
3.4	Physical network	22
3.5	Flow of traffic analysis	23
3.6	Example output from the AppCensus application [30]	26
3.7	DirectHTTPLeak1 application code	27
4.1	Capture of personal data exfiltration in testing app	29
4.2	Analysis of personal data exfiltration in testing app	30
4.3	Root detection by the application	34
5.1	Build Docker image	39
5.2	Run Docker image	40
5.3	Restart stoped Docker container	40

List of Tables

2.1	Platform execution comparaison	5
2.2	App execution techniques	7
2.3	Analysis of interception techniques	9
2.4	Analysis of traffic capture techniques	9
2.5	Analysis of TLS bypass techniques	11
3.1	Analysis of interception tools	17
3.2	Analysis of traffic capture tools	17
3.3	Analysis of TLS bypass tools	17
4.1	Output of test analysis	30

Acronyms

ADB	Android Debug Bridge
AEPD	Agencia Española de Protección de Datos
API	Application Programming Interface
APK	Android Application Package
CA	Certificate Authority
CSV	Comma Separated Values
DH	Diffie-Hellman
DHCP	Dynamic Host Configuration Protocol
DLP	Data Leak Prevention Systems
DNS	Domain Name System
EEA	European Economic Area
EU	European Union
GDPR	General Data Protection Regulation
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
IDS	Intrusion Detection Systems
IoT	Internet of Things
IP	Internet Protocol

JSON	JavaScript Object Notation
OEM	Original Equipment Manufacturer
OS	Operating System
ROM	Read Only Memory
SSL	Secure Socket Layer
TLS	Transport Layer Security
TWRP	Team Win Recovery Project
UPM	Universidad Politécnica de Madrid
USB	Universal Serial Bus
VPN	Virtual Private Network

1 Introduction

1.1 Introduction

On the 6 of December of 2018 a new law for data protection was passed in Spain [1] ending a long process of implementing a new regulation for data protection in the European Union (EU). This law supplemented the new General Data Protection Regulation (GDPR) to the context of Spanish law and developed the interest in technical tools that allow the detection of its infringement.

EU laws concerning data protection have always been considered thorough and concerned with the protection of citizen's rights, but the spread of new technologies is faster than the development of laws protecting users' rights. To fix problems with the older directive a new regulation was approved in 2016 [2].

The GDPR can be supplemented in each country of the EU and gives power to the different data protection agencies to investigate and fine organizations that do not follow the new laws and regulations. In Spain this job is done by the Agencia Española de Protección de Datos (AEPD).

This master thesis has been developed in the context of a project between the Universidad Politécnica de Madrid (UPM) and the AEPD to develop a platform to analyze Android apps to detect exfiltration of sensitive information and to analyze a finite set of them. The apps to be analyzed will be selected from those intended for children and those intended for health and sports.

The project will approach the analysis of the selected applications from two different perspectives: a static analysis approach which aims to detect possible flows of information inside the application and traffic analysis approach to detect actual data exfiltration. This master thesis belongs to the second perspective of the project.

The Android ecosystem is dominated by the usage of the Hypertext Transfer Protocol (HTTP) which uses Transport Layer Security (TLS) to protect its connections. Because the thesis takes a traffic analysis approach, those protocols are the ones the thesis will analyze.

1.2 Objectives

The project aims to be able to analyze twenty Android applications with the objective of detecting private data exfiltration, those apps are selected from two main groups: apps intended for educating children and those intended for monitoring biometric and sports data. A secondary objective of the project is to develop an online platform, with the tools developed, to be able to analyze any arbitrary Android application.

The traffic analysis should be able to detect to which application does a connection belong to, whether it was made before or after the user started using the app, the type and category of the exfiltrated data, whether the connection was made with the app in foreground or background, the destination domain and which category does it belong to, the country in which the server is located and whether the connection uses some kind of encryption.

To be able to do all of this, it is necessary to be able to intercept a connection and to be able to bypass any protection mechanism that it uses. The thesis will be centered around HTTP connections, so it will be necessary to overcome the protection mechanisms implemented by TLS. This can include standard protections but also more advanced protections like certificate pinning.

Finally, because the system developed in the thesis is intended as one element in a bigger project, the system should be composable in order to work with the rest of the systems. Also it is going to be necessary to present an output that can be understood by non technical users.

1.3 Methodology

The first step will be to analyze the state of the art on interception and traffic analysis. Different techniques and tools that implement those techniques will be explored to determine the most appropriate ones for the job. It will also be necessary to explore different emulator and platform technologies where an app could be run.

It will also be necessary to determine the best way to execute the applications in a way that as much as possible traffic from the application can be generated to be intercepted. Different strategies will be analyzed and classified based on those situations they are most suited to.

Finally, different strategies to bypass TLS encryption will be considered. There will be strategies that will focus on overcoming plain TLS and those focused on bypassing different certificate pinning strategies.

The second step is to design the architecture of the system, the different modules and their interactions will be specified. Those modules will be explained, detailing their usefulness and the reasons behind the choices made during their design.

The third step is to detail the implementation. The tools and techniques selected in the previous steps will be used to create a single solution that can intercept and analyze HTTP connections determining if any sensitive data has been exfiltrated.

At the same time an Android app will be developed to test the effectiveness of the system. The app will send sensitive information through an encrypted HTTP connection to a trusted server, and the system will have to detect the exfiltration.

The last step will be to apply the developed system to real world applications. Twenty applications will be selected and analyzed, and a summary of the detected behaviours will be produced. This summary is intended for non-technical readers and will be as short and as straightforward as possible.

2 State of the Art

2.1 Introduction

Traffic analysis for network applications has matured and there are many tools and systems available today. To monitor enterprise networks there are systems like Intrusion Detection Systems (IDS) or Data Leak Prevention Systems (DLP) and for desktop applications there are programs like Wireshark. Many tools and techniques are designed to be able to work with any kind of terminal device, and can be used to analyze Android traffic.

In the Android ecosystem tools for traffic analysis are not as evolved. Some tools that have been developed specifically for Android are: PrivacyGuard [3], Lumen [4], AntMonitor [5] and ReCon [6]. There is also a platform to analyze Android apps which includes traffic analysis called AppCensus.

There are also multiple Android emulators, mostly designed with developers or playing games in mind, but they have some limitations that may make necessary to use a physical device.

In this section all of these tools and techniques will be explored with the intention to determine in the next one which ones will be most suited to the purpose of this thesis.

2.2 Execution

To be able to study an applications traffic it is necessary to generate traffic from the target application. To be able to do this the application has to be executed. The execution of an application has two main problems, where to execute and how to execute the application.

In this section this two problems will be analyzed to determine different possibilities on solving them. First the different options for a platform will be presented, analyzing their strengths and weaknesses. After that, different means of executing applications will be treated in the same way.

2.2.1 Platform

The different platforms for Android applications execution can be divided in two main groups: virtual devices and physical devices. The virtual devices are mostly android emulators, like Genymotion [7] or the Android Emulator [8], but can also be created with Android x86 [9] in traditional hypervisors.

Virtual devices are easier to use but can result in an analysis that does not generate all of the possible events of the application, especially in an environment that relies so heavily on access to the physical hardware. In applications that access the hardware a physical device will be a necessity.

In general an Android-x86 will be easier to use for traffic interception than emulators because the latter are developed with other purposes in mind. There will be also different aspects of the hardware that each one will emulate.

To detect which connections belong to which application it will be necessary to access information in the device that only a root user will be able to access. Most virtual devices are already rooted, but physical devices can be difficult or complex to root.

Table 2.1: Platform execution comparaisn

Platform	Ease of use	Ease of root	Hardware emulation	Price in euros
Phone	Low	Low	Optimal	100-400
Emulator	Medium	Easy	Medium	0-100
Android-x86	High	High	Medium	0

In table 2.1 there is a comparison between all the different platforms that have been presented. This table will be useful when determining which platform is most useful for different contexts and applications.

2.2.2 App execution

In a thorough dynamic or traffic analysis it is necessary to generate as many events and behaviours as possible in the target application. It is also necessary to generate them in a short period of time, to have an efficient process of analysis.

It is possible to distinguish between three types of generating behaviours [10]. The three types are: a manual, an automated and a semi-automated approach [10]. To evaluate the different approaches the following aspects have been taken into account: predictability, thoroughness, speed and ease of deployment.

The basic manual approach consists in having the analyst executing the application and trying to generate as many events as possible. The advanced manual approach consists in having a group of people execute the application as they normally would, and collecting the produced data over a considerable span of time [4].

The basic automated approach consists in using some tool capable of generating random events in the application, a monkey exerciser [11], in the hopes of generating useful events. The advanced one consists in creating a tool that tries every possible interaction with the application to generate every event possible [12].

Finally, the semi-automated approach is an improvement over the basic manual approach [10]. It consists in the recording of one session of the interaction with the application so it can be replayed at a later time.

The basic manual approach is fast and really easy to deploy, but it lacks in thoroughness and predictability. The basic automated approach has the same advantages and drawbacks, but pushed to the limit. It is faster, easier, less thorough and more unpredictable.

On the other hand the advanced approaches stand out in their thoroughness, and predictability but lack in ease of deployment and speed. The automated one is more predictable and faster than the manual one. Finally, the semi-automatic approach is very predictable and fast. But it is also not too easy to deploy and as thorough as the basic manual approach can be.

In table 2.2 there is a summary of the analysis of the different app execution techniques. It will be useful to determine which one is going to be used and in which context. The basic manual approach or the semi-automatic approach will work best when dealing with a small amount of applications to test. Meanwhile the advanced approaches will work best when having access to more time and resources. The basic automated approach will only be useful with really basic applications.

Table 2.2: App execution techniques

Technique	Predictability	Thoroughness	Speed	Ease of Deployment
Basic Manual	Medium	Medium	High	Very High
Basic Automated	Very Low	Low	Very High	High
Advanced Manual	Medium	High	Very Low	Very Low
Advanced Automated	Very High	Very High	Low	Low
Semi-Automated	Very High	Medium	Medium	Medium

2.3 Interception and capture

To be able to deploy traffic analysis in any kind of network application it is necessary to intercept and capture as much traffic as possible from the target application. To keep the target application running as usual this process should be as transparent to it as possible.

To intercept and capture network traffic it is possible to classify the tools and techniques in three categories. Interception tools and techniques route traffic through a chosen machine. Passive capture techniques store traffic that has been intercepted without interfering with it [13]. Active capture techniques interact with the traffic to be able to capture more detailed traffic [13].

2.3.1 Interception

There are two fundamental techniques to intercept traffic for any kind of terminal device. These are, the usage of a Virtual Private Network (VPN) or the usage of a proxy. In this subsection both of these techniques will be analyzed in the context of their usage in the thesis.

Both techniques will be analyzed in relation to four distinguishing factors: if they can be reused for different terminals or Android versions, if they require extra infrastructure, the interception layer and the latency overhead [10].

A VPN allows the creation of a private network over a public one by means of virtual tunnels between the nodes on the network [3] [5]. Android offers an Application Programming Interface (API) to create a VPN that can intercept traffic at the network layer.

Using this API it is possible to deploy a VPN in the Android terminal that will not need any more infrastructure, or to tweak with the network configuration. This will make any increase in latency irrelevant. [10]

On the other hand, it will have some drawbacks, because the interception will be done at the network layer it will be necessary to implement the rest of the protocol stack. Also, any developed VPN will be tied to the Android and API version which it targets. [10]

A proxy is a system that works as an intermediary between two or more systems in a network. There are many types of proxies [13], the most useful for the development of these theses will be the regular or the transparent proxy.

The proxy receives the packets sent from the terminal and forwards them to the desired destination. The proxy is independent of the terminal in which the target application is running but it will be necessary to deploy extra infrastructure and possibly to tweak the network to transfer the traffic through the proxy. [10]

It will also increase the latency of the connections because they will travel over a longer path through the network. But it will not be tied to any version of Android or any type of terminal, so it can easily be reused [10]. The proxy can work in any layer, but most of them work at the application layer [13].

In table 2.3 there is a summary of everything that has been analyzed of interception techniques. It will be useful to determine which one is going to be used and in which context.

Table 2.3: Analysis of interception techniques

Technique	Terminal Independent	Extra infrastructure	Extra latency	Layer
VPN	No	No	No	Network
Proxy	Yes	Yes	Yes	Any/Application

2.3.2 Capture

When capturing network traffic there are two basic approaches that one can take: a passive capture and an active capture [13]. The difference is in the level of interaction between the capture application and the target system.

When capturing traffic passively there is no interaction with the target system [13], all traffic that is intercepted is stored as it was intercepted. This technique is really easy to use, but suffers when modifications to the state of the target system are necessary, for example when dealing with TLS traffic.

On the other hand capturing traffic actively means that the capturing application interacts with the target system. This can allow a lot of flexibility, like serving different certificates for TLS connections, but is not as straightforward as the passive technique.

Table 2.4: Analysis of traffic capture techniques

Technique	Ease of use	Flexibility
Passive capture	High	Low
Active capture	Low	High

2.4 Analysis

After executing the target application, and intercepting and capturing its traffic it will be necessary to analyze the contents of the traffic that has been captured. This will present a series of challenges, including encryption bypassing, data location and data detection.

In this section these challenges will be analyzed and some techniques to solve them will be presented. Because this thesis deals with Android applications that generate HTTP traffic those challenges will be analyzed with that context in mind.

2.4.1 TLS bypass

HTTP is not a secure protocol, but Secure Socket Layer (SSL) was developed at Netscape with the aim of offering confidentiality and integrity to many application protocols like HTTP [14]. It was later renamed to TLS as it is known today.

TLS is used very frequently in Android applications [15] and it will be necessary to bypass it in order to be able to inspect the traffic. There is also another technique, called certificate pinning, that will be necessary to take into account when dealing with TLS encrypted traffic.

There are three main techniques to bypass TLS encrypted traffic: certificate supplantation, TLS downgrading and modifying the target application [10]. These techniques will be compared based on their ease of use, availability and effectiveness.

When a TLS connection is established a certificate is sent to the client to be verified and to confirm that the server is who it says it is [15]. When verifying a certificate the client checks if there is a valid chain and it has been signed by an Certificate Authority (CA) it trusts [15].

If the client trusts the certificate the server, or an attacker who is trying to supplant it, will be able to read all the traffic encrypted in that session. To be able to supplant the certificate of the server it is necessary to insert a certificate to the terminal that it will trust as a CA.

Between Android 4, named Ice Cream Sandwich, and Android 7, named Nougat, there was a user certificate store where any user could add a trusted CA [16]. But after Android 7, to improve security, the Android team decided to change it [16]. After that, any user added certificate has to be explicitly trusted by any application, so it becomes necessary to root the phone to be able to supplant a certificate.

A downgrade attack consist in intercepting the initial connection handshake to limit the options so that the client and server can only use an insecure version of TLS. This can be done by downgrading to an HTTP connection, to an older version of TLS or SSL or to an insecure option like anonymous Diffie-Hellman (DH) [10].

Finally, if it is possible to modify the target application it is possible to remove the certificate validation, make it validate any certificate, make it use pure HTTP connections and other things. This will allow to overcome plain TLS and also certificate pinning.

Between the three options certificate supplantation and downgrade attacks are easy to use, although neither of them can overcome certificate pinning. Downgrade attacks are more effective against a human user, because most clients wont accept a downgraded connection unless a human accepts it first. On the other hand, modifying the target application is really effective but can be really time-consuming.

Table 2.5: Analysis of TLS bypass techniques

Technique	Ease of deployment	Availability	Effectiveness
Certificate supplantation	Easy	Medium	Medium
Downgrade attack	Easy	Low	Medium
Target modification	Hard	Medium	High

In table 2.5 there is a summary of everything that has been analyzed of TLS bypass techniques. It will be useful to determine which one is going to be used and in which context.

2.4.2 HTTP sections

Analyzing the captured traffic it is possible to make a distinction of the different places where information is stored in an HTTP connection [10]. Based on this it is possible to categorize the information to be analyzed in three groups: information related to the domain or to the Internet Protocol (IP) address, information stored in the HTTP headers and information stored in the HTTP query or body [10].

From the domain or IP address of the server that is receiving the connection it is possible to obtain information of the geographical location of the server (transferences of information out of the EU is important in the GDPR because of lax privacy regulation in other countries) and about the usage of marketing or tracking software that may violate the users privacy [10].

In the headers there can be a lot of information related to the privacy of the user including, but not limited to, tracking cookies, obfuscated information, terminal identifiers and more[10]. In the android ecosystem it is not really common to modify the header information and it can be a source of false positives because the system adds phone identifiers to every connection by default.

Finally, most information will be found in the body or the query of the HTTP request. Most information in the body of the request will be transmitted as text, but different techniques have been developed to be able to send multimedia content like images or sound.

To detect multimedia content like images, video or sound the best approach is to look for magic numbers that identify different file formats in the captured traffic [17]. But this technique can still produce false negatives if the magic numbers are missing, or if the packets have been split in a certain way [17].

Text based information transmitted through in a HTTP connection is the easiest to analyze. If the analyst has access to the transmitted value (like the phone number or an email address) it is just a matter of matching patterns in the traffic [10]. But it is necessary to keep in mind that this information can be obfuscated prior to be sent.

2.5 Selection

After analyzing the different aspects and techniques the next step is to determine which ones are the most appropriate to the context of the analysis. In this section the techniques that are going to be used through the rest of the thesis will be selected, based on the previous analysis.

2.5.1 Platform

The target applications will correspond to 10 educational applications and 10 sports applications. The latter will need to access specialized and external hardware so this apps can only be run in a real phone. On the other hand, testing and development are much easier in a virtualized device.

Based on table 2.1 it is clear that the best option for those applications that need access to hardware is the phone. If the phone is selected it would not make a lot of sense to use another option for the rest of the applications because access to a better emulation will produce better results even if it is not needed.

On the other hand, testing and developing in the phone will be harder and more time consuming than using a virtualized device. Between the virtualized devices the preferred option is the Android x86 virtual machine because of the ease of integration with traditional hypervisors, making it easier to work with.

During the development of the thesis the virtual machine has been used to develop and test the system and a phone, Xiaomi Redmi Note 5, for the real analysis.

2.5.2 App Execution

All the target applications have an initial activity in which they ask for credentials or some other login scheme. This makes it impossible to use any of the automated approaches to application execution. This leaves three of the options in table 2.2.

Of those three the advanced manual technique it is really resource consuming, it also produces results after a long time. Both of those are limiting factors in the development of the thesis, so that option will be discarded as well.

The difference between the basic manual and semi-automated approach is the ability to reproduce the test and the ease of deployment. Because the different applications will only be run once for analysis, ease of deployment trumps reproducibility.

Throughout the thesis the basic manual approach will be used. This could present a problem if the number of applications to test was high, making it necessary to rethink the usage of an automated approach. In this case 20 apps is a manageable task with the basic manual approach.

2.5.3 Interception

The analysis and testing machines will run in a local network which makes the extra latency irrelevant in this context. Both machines will be Android devices and will share a version, making terminal independence not a priority.

Based on this both techniques from table 2.3 seem equally appealing. The proxy approach was selected because the work necessary to parse the top layers of the network stack and the abundance of tools made it a better choice. The separation of the interception from the target device also will be useful if the system is later expanded to other target systems, like Internet of Things (IoT) devices.

2.5.4 Capture

In the process of the execution of the target application it will contact any number of unknown domains over TLS. In this situation passive capture techniques are not enough to capture useful traffic, because it is necessary to strip the encryption layer. The only other option, see table 2.4, is active capture.

2.5.5 TLS bypass

To select a TLS bypass method those methods that are not readily available should be discarded first to not find oneself looking for a suitable alternative again. After that easy to use methods are preferred.

Based on this, CA supplantation will be selected, see table 2.5. In the case where the target application uses advanced protection like certificate pinning the target application should be modified to bypass those protection mechanisms.

3 Design

3.1 Tools

For the platform an Android x86 virtual machine and a Xiaomi Redmi Note 5 Pro will be used. An Anbox [18] setup was also considered but discarded because it is not mature enough yet, but it could be useful in the future.

Some tools that have been considered cover multiple of the aspects to consider (interception, capture and/or TLS bypass). With this in mind all of the different tools will be compared at the same time, but it does not mean that all of them are incompatible.

AndroidVPN is the most common solution used by applications of traffic analysis [3][5][17][6] or some tool based on it. With this tool it will be necessary to rebuild the upper layers of the network stack and add extra TLS bypass functionality.

This tool comes with a number of downsides [10]: 1) it is not portable to devices that do not run Android, 2) it executes in the target device, being incompatible with other VPNs and consuming device resources, and 3) the work necessary to implement the network stack and TLS bypass functionality.

On the other hand it also has a number of advantages [10]: 1) when used with a custom CA it is really easy to use, 2) it does not need any extra infrastructure because everything runs in the target device, and 3) the Operating System (OS) guarantees that all the traffic will be intercepted and captured by the VPN.

Mitmproxy is an interception and traffic capture tool for HTTP and HTTP over TLS, also called Hypertext Transfer Protocol Secure (HTTPS) [19]. It is mature and extensible and it needs to have its own CA certificate installed in the target device [19]. It is necessary to have in mind that applications that target API 24 or later, Android 7 or later, would only trust the certificate if explicitly designed or if it is in the system certificate store, only modifiable by the root user [20][21].

The biggest downside is the need to design a network that ensures that all the traffic would flow through the proxy [19][10]. It also has the downside that it is designed to work only with HTTP and HTTPS traffic, so other protocols like QUIC will not be detected.

On the other hand it has a number of advantages [10]: 1) if the installation of the CA certificate is not a problem it is really easy to use, 2) all the network logic is already implemented, and 3) it is possible to separate the interception and capture of the traffic from the target device.

Tcpdump is a tool for capturing traffic with a long history and widespread usage [22]. It does not have any TLS bypass mechanisms. Its main advantage is its widespread usage, and it has some disadvantages [10]: 1) it is necessary to implement a network to be sure all traffic is captured, and 2) it captures traffic passively.

Ssldump is a sister tool to tcpdump, the only difference is that it allows to bypass TLS protection [23]. To do this it is necessary for the target device to accept its CA certificate.

Wireshark is also a tool to capture traffic that has a widespread usage. It offers a graphical interface and a command line utility [24]. In advantages and downsides it is similar to tcpdump.

Frida [25] is a tool to insert code into a process allowing many reverse engineering tasks. In this case it could be used to bypass TLS protection, including certificate pinning. It is specially useful in the case that the certificate pinning is not left to the Android OS.

Apktool is a tool to disassemble Android applications, which follow the APK format. It is really useful to be able to inspect, analyze and modify an applications source code [26]. It can be used to determine in which ways an application obfuscates or protects its code, and complementing the usage of Frida.

In the tables 3.1, 3.2 and 3.3 all the analyzed tools are summarized in the relevant categories. Mitmproxy covers almost all of the techniques selected in section 2.5 making it the tool of choice, in the case that the CA supplantation is not enough the usage of Frida and/or apktool will also be an option.

Table 3.1: Analysis of interception tools

Tool	VPN	Proxy
AndroidVPN	Yes	No
Mitmproxy	No	Yes

Table 3.2: Analysis of traffic capture tools

Tool	Active	Passive	TLS capture
AndroidVPN	Has to be implemented	Yes	Has to be implemented
Mitmproxy	Yes	No	Yes
Tcpdump	No	Yes	No
Ssldump	No	Yes	Yes
Wireshark	No	Yes	Yes

Table 3.3: Analysis of TLS bypass tools

Tool	CA supplantation	Downgrade attack	APK modification
AndroidVPN	Has to be implemented	Has to be implemented	No
Mitmproxy	Yes	No	No
Frida	No	No	Yes
Apktool	No	No	Yes

3.2 Platform setup

To be able to start developing or testing any application it is first necessary to setup the platform or platforms in which it is going to be executed. In this case the three most important aspects are rooting the device (to be able to extract which connection belongs to which application), installing the proxy's CA and setting up the network connectivity.

The virtual machine with Android x86 already comes with a rooted setup, but it is necessary to configure its network interfaces. Android is built for devices that use a wireless connection because smartphones have network cards that lack an Ethernet port. Android x86 has to emulate a wireless connection through a wired one when ran inside a virtual machine and it is only capable of doing so when using Dynamic Host Configuration Protocol (DHCP).

On the other hand, in the physical phone the network will be configured almost automatically, the only manual modification will be the selection of the proxy. But getting root access is a lot harder. To archive this, the step to follow are:

1. Unlocking the bootloader.
2. Flashing a custom recovery like Team Win Recovery Project (TWRP).
3. Install rooting application.

To install the CA certificate, in either of the devices, it is necessary to access the root certificate store because from Android 7 any app that wants to trust any user certificate has to explicitly say so [20][21][16]. Because no application will trust the mitmproxy CA certificate it is necessary to manually add the certificate to the root store.

Android certificates have to be an x509 certificate and its name should correspond to its openssl 0.9 compatible hash [27]. After that, the system partition should be remounted with read-write permissions to be able to move the certificate to the appropriate path.

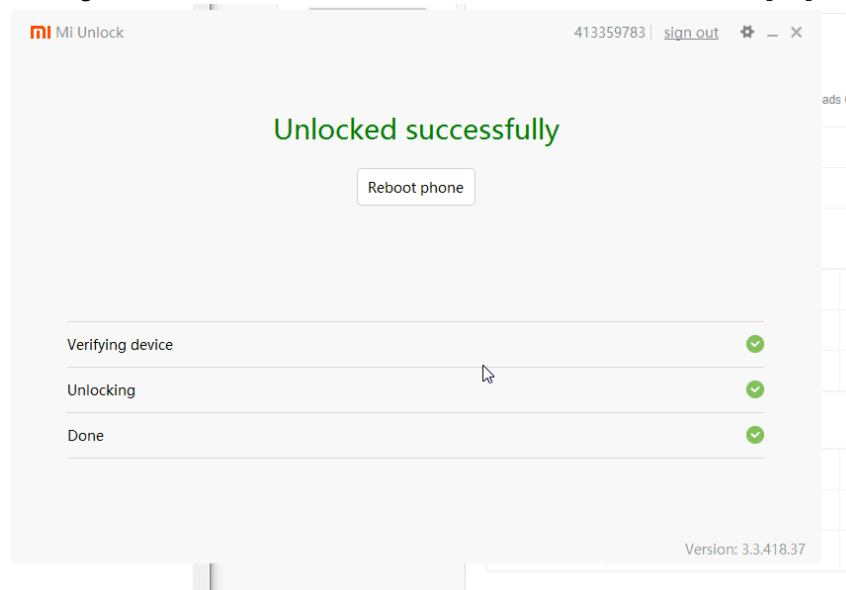
3.2.1 Rooting the phone

Unlocking the bootloader

To unlock the bootloader it is necessary first to enable Universal Serial Bus (USB) debugging and Original Equipment Manufacturer (OEM) unlock. This will allow a computer that connects through Android Debug Bridge (ADB) or fastboot to unlock the bootloader.

In the case of Xiaomi, which is the manufacturer of the selected phone, it is necessary to go to their unlock page (here) to apply for permissions. After approximately 15 days the permissions will have been granted and it is possible to download the Xiaomi Windows unlock tool from the same web page.

Figure 3.1: Xiaomi unlock tool success. Obtained from [28]



The Xiaomi Windows unlock tool will require the phone to be connected to the computer in fastboot mode and will offer an easy, guided way to unlock the bootloader. It may require to wait some more hours because of extra protections against spam or unauthorized access to the phone. The successful result of the tool can be seen in figure 3.1

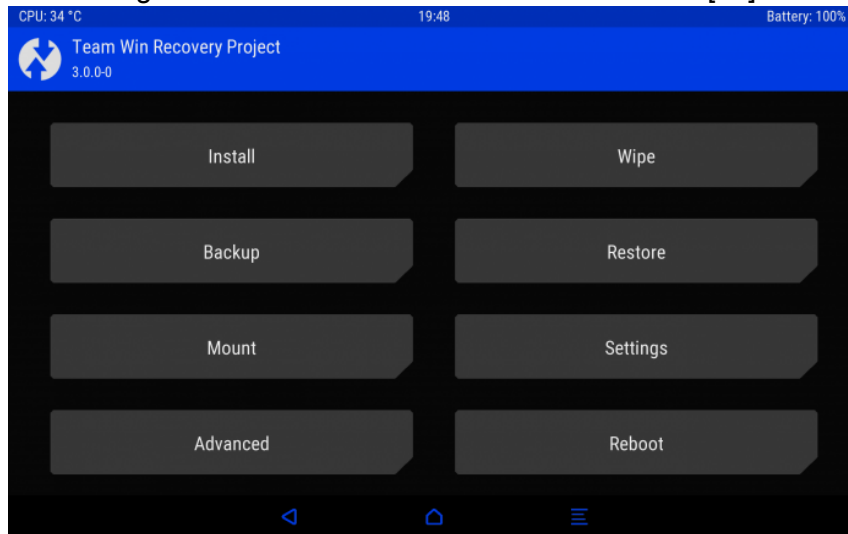
Flashing a custom recovery

The most used custom recovery in the Android ecosystem is TWRP, and it is the one used throughout this thesis to root the physical phones. It can allow to modify the installed firmware in the Read Only Memory (ROM) or to install other types of enhancements.

With an unlocked bootloader on any typical Android phone it is just necessary to flash the custom recovery file using fastboot to be able to continue. But the Xiaomi Redmi Note 5, the phone used in this thesis, has some extra protections, like anti-rollback protection, which complicate the process.

To be able to root the phone without causing any damage to the phone it is preferable to boot into a live TWRP without flashing it into the device. This can be done using the fastboot tool and will not trigger the extra protections. On the other hand it is necessary to live boot to the custom recovery every time something has to be modified. The result of booting into TWRP should look like figure 3.2.

Figure 3.2: Booted into TWRP. Obtained from [29]



Installing rooting application

Form TWRP it is now possible to install special applications like SudoSU or Magisk to get root access to the phone. The latter is preferable because it has the ability to hide the fact that the device has been rooted. This will be useful in the case that any application has any anti-reversing techniques implemented.

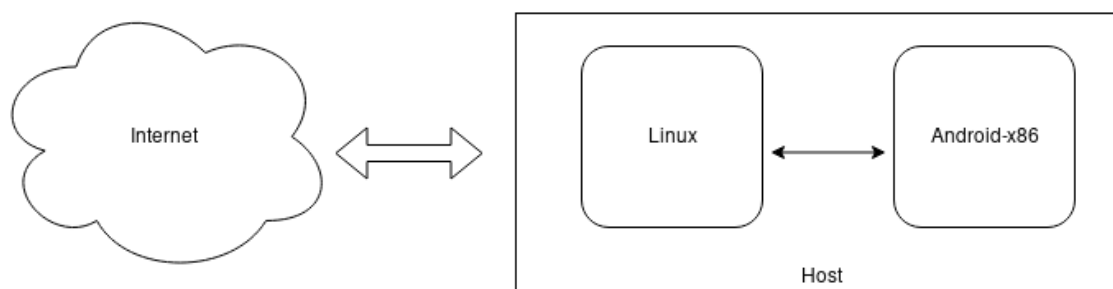
To install the magisk application it is necessary to copy the zip archive to the device and install it from TWRP. After that, reboot to Android and download the MagiskManager to control the different settings of the rooted device. MagiskManager has to allow root access form ADB to work properly.

3.3 Network

The structure and the design of the network is different for the virtual and physical devices to make it easier to intercept the traffic. For the virtual devices the proxy will work in a transparent mode and in the physical case it will work as a regular HTTP proxy.

The network for the virtual device will be called "the virtual network" and the one for the physical device will be called "the physical network" through this section.

Figure 3.3: Virtual network



3.3.1 Virtual Network

The virtual network will consist of two virtual machines, an Android-x86 machine where the applications will run and a Linux machine where the proxy will run. The Android machine will be connected through an internal network to the Linux machine which will also have a connection to the internet.

The Linux will also work as the DHCP and Domain Name System (DNS) server for the Android machine. The only option for the application traffic is through the Linux machine where it would be captured and routed to its destination.

This setup makes working with the proxy and the interception of traffic really easy, and will be used throughout the development process. The deployment is also much simpler than the physical setup. Figure 3.3 illustrates the design.

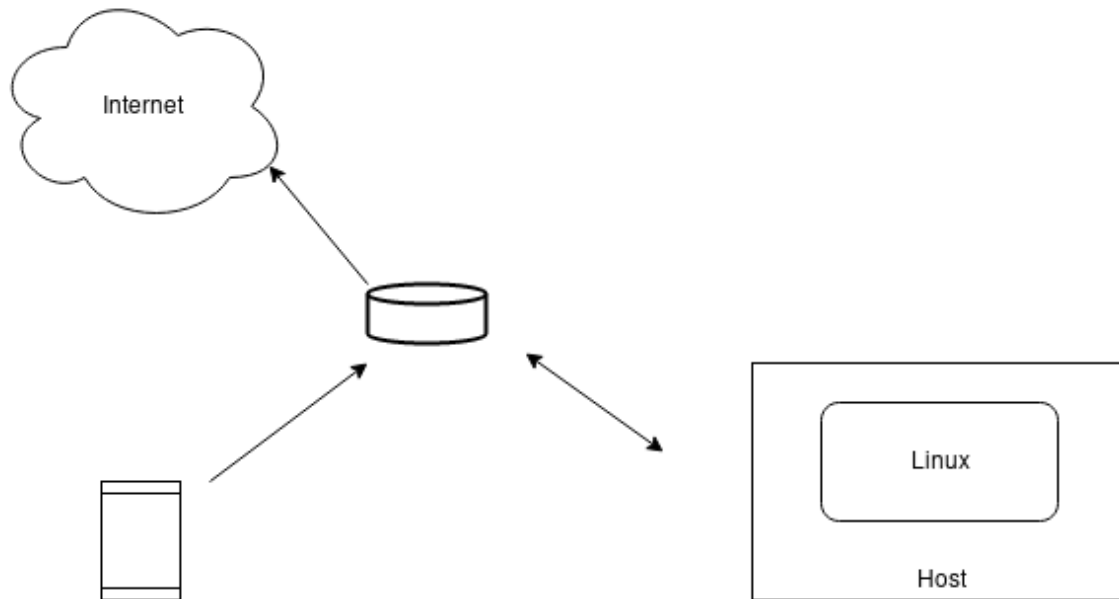
3.3.2 Physical Network

The physical network will consist of one virtual machine, a Linux machine where the proxy will run, and a physical phone where the applications will run, a Xiaomi Redmi Note 5. They will be connected in a private network through a router.

The router will also act as an DHCP and DNS server for both machines. To route the traffic through the proxy the Android device will be configured to send all its traffic through the virtual machine.

With this change the proxy should be changed from a transparent proxy to a regular one. This will be easier than trying to replicate the transparent proxy setup from the virtual network. Figure 3.4 illustrates the design.

Figure 3.4: Physical network



3.4 System design

The system designed for traffic analysis throughout this thesis has three main tasks or concerns that are orthogonal to each other: the installation of the target application, the interception and capture of traffic and the analysis of that traffic. The system has been developed with two main modules which encompass those task as can be seen in figure 3.5.

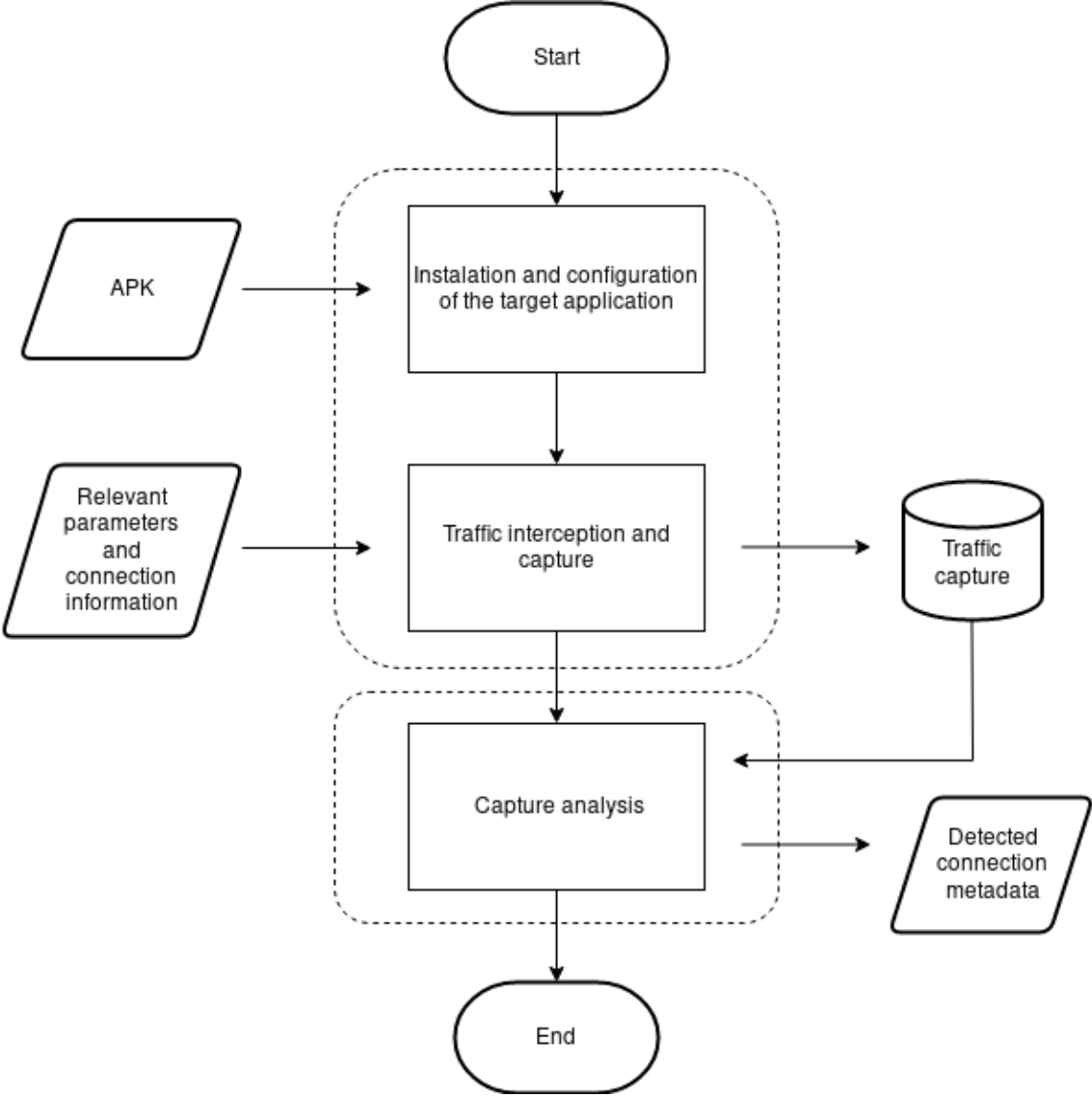
3.4.1 Traffic capture module

The traffic capture module will take care of the interception and capture of the traffic from the target application, if the target application is not already installed in the device it will also take care of the installation and configuration.

To install and configure the target application in the device it will need as input the target application in Android Application Package (APK) format. And for the interception and capture it will need some parameters the IP address, and it will also obtain other relevant information from the device while running.

As output it will generate an capture file with the relevant connections in a format that is comprehensible to the analysis module. Any connection made by the target is considered relevant, including those that produce errors because of lack of trust in the certificate or any other reason.

Figure 3.5: Flow of traffic analysis



3.4.2 Analysis module

The analysis module will take care of the analysis of the traffic capture generated by the traffic capture module. As input it will receive the capture file or files generated after the interception and as output it will generate a report detailing detected data exfiltration with important metadata.

For each detection it will detail if the detection was before or after user interaction, the exfiltrated data, its category, the destination domain and port, the domain category, the country where the destination server is located, if the domain is outside the European Economic Area (EEA) and if the connection used TLS. If the connection had an error it will be indicated as such.

3.5 Implementation

3.5.1 Traffic capture module

The traffic capture module is built around mitmproxy, with a wrapper script to set up the interception and a plugin to store the relevant connections from the target application. The wrapper script will configure the appropriate network parameters and forwarding and will start the proxy.

The mitmproxy plugin will have two main classes: one to check TLS failures and another one to intercept and store the appropriate traffic. The TLS failure detector will capture exceptions produced by the TLS layer and store an appropriate record when a connection cannot be established, including information such as server IP address or domain.

The interceptor class will use the ADB tool to match open ports to running applications in the target device. With this information it will only store those connections that belong to the target application in the log file.

3.5.2 Analysis module

The analysis module is more complex than the traffic capture module. It is written in Python and runs with 10 threads in parallel to reduce the input-output bottlenecks. In each thread one connection is analyzed building a dictionary with the relevant information which can later be printed in JavaScript Object Notation (JSON) or Comma Separated Values (CSV).

For each request its body and query will be inspected and a predetermined set of values, obtained from the target device, will be searched for. If any value is detected the connection is stored as an exfiltration and other information is analyzed.

To find the location of the server receiving the connection the online tool ipstack is used. Those connections are the biggest source of latency, so the responses are memoized, in a thread safe manner, in order to reduce execution time.

3.5.3 Control server

To integrate and control in a centralized manner the whole system a control server has been designed. To be able to integrate the system with a great variety of clients and other systems the control server will open an HTTP service with an API to receive commands.

3.5.4 Docker

For portability and ease of usage the system has been bundled in a Docker container. The container is based on the Ubuntu image and includes tools to interact with the Android device as well as the mitmproxy and both modules developed during this thesis.

The Docker container will be built using a custom image built in two stages. The first one will be used to compile the control server and the second one to install all the developed modules.

3.6 Developed apps

During the development of the system two Android applications have been developed to aid and test the system. The first one automates the retrieval of information from the Android device, data to be used in the pattern matching process. The second one will obtain personal information and try to send it to a domain through an HTTPS connection, it is going to be used in the validation stage.

Figure 3.6: Example output from the AppCensus application [30]

```
# EXAMPLE - this file itself could be a device file:
testerName: John Test
phone:
email: icsisensors@gmail.com
name: IOR Blues
imei: 357478061454986
wifimac: c4:9a:02:84:fc:38,02:00:00:00:00:00
aaid: f175da20-71fb-46a9-99a4-19d918fb5967,ea00cbdf-2cf0-487c-b5e2-706104caef48
gsfid: 353EDD229661B40F
androidid: 804608AEC9153C7F
hwid: 0e742a00037c8002
simid:
fingerprint: ioreyell300945
geolatlon:
routermac: 38:1c:1a:c4:ba:b0,ae:22:0b:8d:40:aa,48:5d:36:a3:d0:9a,2c:30:33:bd:34:53,94:62:69:70:50:c0,54:65:de:
routerssid: ICSI,IOR_guest_nomap,FiOS-LLKDU-5G,NETGEAR09,ATT4z75826,Leos,Redlion_Guest
photo:
video:
audio:
```

3.6.1 DeviceData

The application to retrieve the device information, based on the one developed for the AppCensus project [30], will consist in only one activity that will collect as much information as possible and store it in shared storage as a text file. Other data, like information introduced in a form, will be added manually.

Most of the collected data will be present in both devices (physical and virtual), but some of them may return a null pointer in the virtual devices if it was not emulated. Others will produce an exception in the virtual device. In figure 3.6 an example of the output file is shown.

3.6.2 DirectHTTPLeak1

The application will consist in only one activity that will exfiltrate first the build id of the device and then all of the email accounts that the user has configured. The exfiltration will be done to an TLS secured domain from the UPM. In the figure 3.7 a snippet of the code can be seen.

3.7 Target applications

After the implementation and testing of the system it has been used to analyze twenty different applications selected from two categories: educational and sports. The criterion for selection has been based in popularity, awards or recommendations by trusted third parties.

Figure 3.7: DirectHTTPLeak1 application code

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);

    exfiltrate(Build.ID);
    ActivityCompat.requestPermissions(this, new String[]{Manifest.permission.GET_ACCOUNTS}, 1);
    AccountManager accountManager = (AccountManager) getSystemService(ACCOUNT_SERVICE);
    Account[] list = accountManager.getAccounts();

    Log.e("Accounts --->", Integer.toString(list.length));
    for (Account acc: list) {
        exfiltrate(acc.name);
    }
}
```

The selected applications from the educational category are:

- School Planner
- Class Dojo
- Duolingo
- Edmodo
- Flipgrid
- Kahoot
- Moodle
- Remind
- Socrative
- TokApp School

The selected applications from the sports category are:

- Google Fit
- Tools & Mi Band
- Mi Fit
- Runtastic Free

- Runtastic Pro
- Samsung Health
- 7 Minute Workout
- Step Tracker
- Strava
- SyncMyTracks

4 Results

4.1 Tests

After the development of the traffic analysis system it is necessary to test it to confirm that it works like it is intended. To this end an application had been developed in section 3.6.2.

In this section the test application will be analyzed with the developed system, but with the raw modules to show their usage. The usage of the final dockerized tool will be explained in section 5.

After the application is installed, by hand or with the traffic capture module, and executed the traffic capture module will start intercepting all the device's connections. In figure 4.1 it is possible to see the capture of the connections that exfiltrate personal information.

Figure 4.1: Capture of personal data exfiltration in testing app

```
192.168.3.35:55064: clientconnect
192.168.3.35:55064: None
192.168.3.35:55064: clientdisconnect
192.168.3.35:43204: GET https://etsit.upm.es/?src=N2G48H
                  << 200 OK 7.9k
192.168.3.35:43202: GET https://etsit.upm.es/?src=beca.privapp@gmail.com
                  << 200 OK 7.91k
192.168.3.35:54058: clientconnect
192.168.3.35:59598: clientconnect
```

When the capture has been finished the next step is to start the analysis module. In this situation the module should detect two exfiltration attempts: the buildid of the device and the email address of the android account. In figure 4.2 this result can be seen, table 4.1 has been added for better reading of the generated output.

Figure 4.2: Analysis of personal data exfiltration in testing app

```
[I]
~/p/analyze|master✓
>>> ./analyze.py ../logs/pruebas/output.log "Fase 2"
Fase,Dato exfiltrado,Categoria,Dominio,Puerto,Cat Dominio,País,Transferencia Internacional,Https
Fase 2,buildid,Identificador de terminal,etsit.upm.es,443,-,Spain,No,Si
Fase 2,email,Información de contacto,etsit.upm.es,443,-,Spain,No,Si
```

Table 4.1: Output of test analysis

Phase	Data	Category	Domain	Port	Cat Dom	Country	Out of EEA	HTTPS
Phase 2	buildid	Terminal id	etsit.upm.es	443	-	Spain	No	Yes
Phase 2	email	Contact info	etsit.upm.es	443	-	Spain	No	Yes

4.2 Final analysis

In this section the selected applications will be analyzed. The process that has been followed is the same as the testing application, and the output of the traffic analysis system for each one of the target applications can be found in the appendix A.

School Planner

The application School Planner did not exfiltrate any personal information or it was not detected. The reasons that could have made an evasion possible will be explored in section 4.3, having in mind that the system has been developed looking to have as few false positives as possible even if this produces false negatives.

Class Dojo

When analyzing the application Class Dojo some of the connections have extra protections and its contents have not been analyzed because of this. The rest of the detected connections exfiltrate terminal ids or contact information.

The contact information that has been exfiltrated belong to one of two groups, name or email information introduced in a form field, and is always sent to a first party domain. The name group consists in names, surnames, usernames or similar information.

The terminal ids are Android account identifiers or a platform identifier. The Android identifier is always sent to a first party domain, but the platform identifier is sent to a third party domain. One of the connections was made before the user started to interact with the application.

Most of the connections were sent to servers outside the EEA, specially those of the first party domains. It is also relevant to notice that some of the connections were not protected by TLS, exposing that private information to anyone in the network.

Duolingo

When analyzing the Duolingo application a lot of information has been leaked. Most of the information is terminal identifiers sent to marketing or first party domains. And a lot of the connections are made to servers outside the EEA.

Sending terminal identifiers is commonly used to profile the user and track its usage of services and applications throughout the Android or web ecosystem. In this case most connections are made to a first party domain, a Facebook marketing domain or to Adjust.

A lot of the connections with personal information were made before the user could interact with the application and, most important, consent to the harvesting of its personal information. This could be an infraction of some GDPR protections.

Edmodo

When analyzing the application Edmodo not much personal information has been detected. Of the connections that could be read and that personal information was detected the information that they contained was introduced in the app's forms. That information could be an email, usernames, names, surnames or similar information. It is also relevant to note that all of the connections were made to servers outside the EEA.

Flipgrid

When analyzing the application Flipgrid not much personal information has been detected, in most of the relevant connections the client did not trust in the supplanted certificate because of extra protections.

All of the connections that could be read and that sent personal information were to an analytics domain, used to track the usage of the application and the application's behaviour. The information sent was terminal identifiers probably used to profile the platform where the application was running. Most of the servers contacted are outside the EEA.

Two of the connections with terminal identifiers were made before the user could interact with the application and, most important, consent to the harvesting of its personal information. This could be an infraction of some GDPR protections.

Kahoot

When analyzing the Kahoot application a lot of connections with personal information were detected. They were mostly sent to first party domains or to *amplitude.com* an analytics domain. Some of the servers were located outside the EEA and others inside, and some of the connections were made before the user could start interacting with the application.

Most of the connections sent terminal identifiers (like the Android advertisement identifier, platform identifiers, build id, etc) or contact information introduced by the user (like name, username, email address, etc). The analytics domain received terminal identifiers and the first party domains a mix of terminal identifiers and user form data.

Moodle

When analyzing the application Moodle not much personal information has been detected. All of the connections were made after the user started interacting with the application and the connections were made to the users' university Moodle server, introduced by the user and considered as a first party domain.

The personal information sent was email introduced in a form in the application, probably in the login process, and some terminal identifiers. This could be done to profile the user to track its activity or to try to detect unauthorized access from strange devices.

Remind

During the analysis of the Remind application personal data exfiltration was detected. Most of the exfiltration consisted in platform identifiers and form information sent to first party servers outside the EEA.

The most relevant group of connections are those that were produced before the user could start interacting with the application, and in consequence give its consent. In those connections, terminal identifiers were sent to first party and marketing domains.

Socrative

When analyzing the Socrative application not much personal information was detected, only one connection to a first party domain (with the server outside of the EEA) with information introduced in a form of the application, probably a user-name.

TokApp School

When analyzing the TokApp School application three groups of connections stand out. During the execution the app sends terminal identifiers and contact information obtained in user forms to first party servers outside of the EEA.

It is also relevant that the application sends personal information, in this case terminal identifiers, to third party marketing domains before the user interacts with the application or consents to this harvesting. This could be a violation of GDPR protections.

Finally, it is important to note that the application sends many times the phone introduced by the user to third party identity provider services. This is unknown to the user, and may result in more personal data collected if the user has an account in the identity provider.

Google Fit

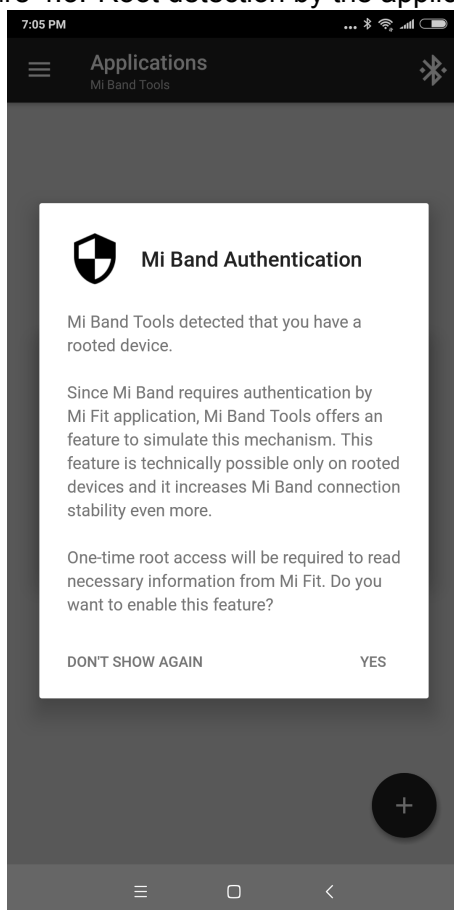
During the analysis of the Google Fit application not many relevant connections were detected. All of them had extra protections and were sent to first party servers outside the EEA. Google is notorious because it uses anti-reversing techniques in its Android apps, so the limitations explored in section 4.3 should be kept in mind.

Tools & Mi Band

During the analysis of the Tools & Mi Band application only one relevant connection was detected. It had extra protection and was sent to a Xiaomi server outside the EEA. As the application uses Xiaomi services this connection should not be a cause for alarm.

The application also detected the fact that the device had been rooted as can be seen in figure 4.3, which may mean the usage of advanced anti-reversing techniques. The limitations explored in section 4.3 should be kept in mind.

Figure 4.3: Root detection by the application



Mi Fit

When analyzing the Mi Fit application few relevant connections were detected. The application sent terminal identifiers to a Facebook domain used for marketing. The most interesting connections were three of them that sent, to a first party server outside the EEA, mac addresses of nearby routers. This information can be used to geolocate a user or to create a profile of it.

Runtastic Free

From the results of the analysis of the Runtastic Free application three groups of connections stand out in their relation to data privacy. One of the aspects to note is that many of those connections are made to servers outside the EEA.

One of the groups of connections that are relevant are those that transmit user form data to first party servers. The information sent consists in usernames, names or surnames and email or phone information.

The second group of connections is also related to user forms. In this case the application sends the name or username of the user to a third party server. That domain is a marketing and analytics domain.

It is also relevant to note the great amount of terminal identifiers to third party domains related to marketing or analytics tasks. This information can be used by those domains to track a user throughout the Android ecosystem.

The last group of connections to note are those sent in the first phase of the execution. Many terminal identifiers are sent to marketing and analytics domains inside and outside the EEA. Those connections are made before the user can consent, which can go against GDPR protections.

Runtastic Pro

From the results of the analysis of the Runtastic Pro application the result is really similar to the results of the free version of the app. The same groups apply and the same analysis has been conducted.

One of the groups of connections that are relevant are those that transmit user form data to first party servers. The information sent consists in usernames, names or surnames and email or phone information.

The second group of connections is also related to user forms. In this case the application sends the name or username of the user to a third party server. That domain is a marketing and analytics domain.

It is also relevant to note the great amount of terminal identifiers to third party domains related to marketing or analytics tasks. This information can be used by those domains to track a user throughout the Android ecosystem.

The last group of connections to note are those sent in the first phase of the execution. Many terminal identifiers are sent to marketing and analytics domains inside and outside the EEA. Those connections are made before the user can consent, which can go against GDPR protections.

Samsung Health

The application Samsung Health produced very few connections which were detected with personal information. All of them sent a platform identifier to a first party server outside the EEA.

The lack of detected connections should not make one conclude that the application does not exfiltrate private information, there could be false positives because of the limitations explored in section 4.3.

7 Minute Workout

The application 7 Minute Workout did not exfiltrate any personal information or it was not detected. The reasons that could have made an evasion possible will be explored in section 4.3, having in mind that the system has been developed looking to have as few false positives as possible even if this produces false negatives.

Step Tracker

The Step Tracker application is probably the one that produced the biggest number of connections detected with private information in them. Most of them are sent to the same third party server. The relevant connections can be separated in two groups for analysis.

The first group are those connections produced in the first phase, before the user can interact with the application and consent to the harvest of personal information. In this group there are a lot of connections which transmit terminal identifiers to analytics and marketing domains. The fact that the user has no option to consent may go against GDPR protections.

The second group of connections were produced while the user was using the application. Those connections contain identifiers that could be used to track the user, and where sent to an analytics domain. From the table it cannot be determined but those connections were made at the same time as the step counter was changing.

There is a possibility that the connections also contained information about the count of steps (considered medical information) that has not been detected. To understand why it may have fail to detect them it is necessary to understand the limitation explained in section 4.3.

Strava

From the traffic analysis of the Strava application some connections stand out. The first characteristic that stands out is the fact that all of the connections to the first party domains have extra protections so they could not be analyzed.

Then it is also relevant to note that the application sends terminal identifiers to a couple of marketing and analytics domains before the user starts interacting with the application and could give its consent. This could be an infringement of some protections of the GDPR.

Finally, it is also relevant to note that the application sends contact information, in this case an email, to a marketing domain outside of the EEA.

SyncMyTracks

When analyzing the application SyncMyTracks one connection was detected, which used extra protection mechanisms. That connection was made to the domain of one of the applications selected to synchronize during the execution of the application. There were also some connections that sent platform identifiers to Samsung servers over plain HTTP and to an analytics domain with servers outside the EEA.

4.3 Limitations

Although the system is able to detect a great number of connections with private data exfiltration it still has a number of important limitations. Because the proxy used is an HTTP proxy, exfiltration in other protocols (like QUIC or DNS, will not be detected.

Another limitation comes from the way the private data is matched in the connection. Because it is based in string matching binary data, obfuscated data, and data that does not have an univocal representation (like a location) are not detected. This makes the system detect mostly identifiers or form data.

Another limitation comes from data that it is not specific enough. In the case of a step counter, if it transmits every second the number of steps walked those numbers will be in a narrow range. If the system tries to look for them it will produce a lot of false positives because they may be part of an API key or similar data instead of truly exfiltrated data.

Another limitation comes from a data race when detecting if a connection belongs to the target application. In the case of connections which do not trust the supplanted CA this data race is most critical because those connections close faster than the system is able to determine if they belonged to the target application.

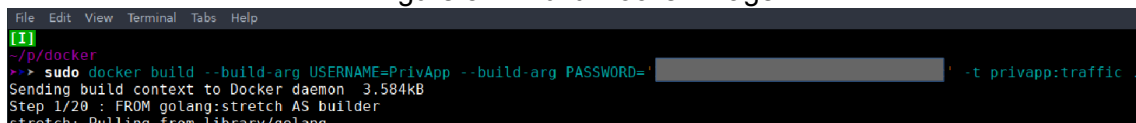
A final limitation is the inability to bypass certificate pinning in applications that make use of it. The method used to bypass TLS protection, CA supplantation, is not useful when the application only accepts a predetermined list of certificates.

5 User Manual

5.1 Installation

The developed system will run in an isolated Docker container, so it will need Docker to be installed. To build and install the Docker image in which the container is going to be based it is necessary to use the command in figure 5.1. This can take some time to run.

Figure 5.1: Build Docker image

A terminal window with a dark background and light-colored text. The window has a menu bar at the top with 'File', 'Edit', 'View', 'Terminal', 'Tabs', and 'Help'. The terminal shows a prompt 'root@kali:~/p#', followed by the command 'sudo docker build --build-arg USERNAME=PrivApp --build-arg PASSWORD=' followed by a redacted password field and '-t privapp:traffic .'. The output shows 'Sending build context to Docker daemon 3.584kB', 'Step 1/20 : FROM golang:stretch AS builder', and 'stretch: Pulling from library/golang'.

5.2 Execution

After the image has been built and installed, as can be seen in figure 5.2, the next step is to start the container. In figure 5.2 it can also be seen how to perform this step.

After that, the container will be running. It will have opened an HTTP server listening in port 7070. The API on the server will listen to a series of commands to execute the different stages of the analysis. Those stages will be explained in the following sections.

Figure 5.2: Run Docker image

```
File Edit View Terminal Tabs Help
[I]
~/p/docker
>>> sudo docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
privapp              traffic             d1d86151742b       About a minute ago 546MB
<none>               <none>             67b8313859aa       33 minutes ago    782MB
ubuntu               latest             7698f282e524       3 weeks ago       69.9MB
golang               stretch          7ced090ee82e       4 weeks ago       774MB
[I]
~/p/docker
>>> sudo docker run -p 7070:7070 -p 8080:8080 privapp:traffic
Listening in port 7070
```

If the container was stopped in the middle of an analysis Docker can restart the container and continue the analysis as shown in figure 5.3. Even if the output of the command does not indicate that the control server is running the service will be listening.

Figure 5.3: Restart stoped Docker container

```
[I]
~/p/docker
>>> sudo docker container ls -a
CONTAINER ID        IMAGE               COMMAND
9523bd9e9750       privapp:traffic    "./control"
[I]
~/p/docker
>>> sudo docker start 9523bd9e9750
9523bd9e9750
[I]
~/p/docker
>>>
```

5.3 Application analysis

Configuration

The server will need to know some information to be able to fulfill its purpose. The two critical pieces of information are the name of the package of the application to be analyzed and the IP address of the target device.

To do this the API has the */config* endpoint. It will receive two parameters the *ip* with the IP address and the optional *name* with the package name. In the case where the application is going to be installed from the proxy the *name* parameter is not needed.

Certificate installation

For the TLS secured connections to be read the target device should have the CA certificate that the proxy is going to use installed. To do this the control server has the */cert* endpoint. To work it will need the IP address to have been set previously.

The device has to be rooted and allow ADB debugging for this step to succeed. If the installation was successful the target device should reboot when the process has finished.

Upload

The next step in the analysis is to upload the target application to the control server. For this propose the control server has the */upload* endpoint. It receives in the HTTP connection the *apk* to be uploaded. If the APK is going to be manually installed this step can be skipped.

First phase

Before starting the first phase of the interception it is necessary to make sure the device is configured to use the system as a proxy. This can be done from the Settings applications, with the port 8080 where the proxy is going to be listening.

To start the first phase another command has to be sent to the control server, in this case the */phase-one* endpoint is used. The control server needs the IP address of the target device and the package name to have been already configured.

Second phase

Before starting to interact with the target application it is necessary to start the second phase. To do this the control server has the */phase-two* endpoint. This endpoint does not need any extra parameters.

Analysis

To finish the second phase of the interception and start the analysis the */analysis* endpoint has been developed. It will kill the proxy and analyze the output from the different phases. It also does not need any parameters to be sent. It will return the result of the analysis.

Result

Finally, to retrieve the results without the need to run the analysis again, the */result* endpoint has been created. It will send the results of the analysis. This step will not erase any data from the container. A fresh container should be run to repeat the interception and analysis or to analyze a different application.

6 Conclusions

6.1 Conclusions

During the development of this thesis the developed system for traffic analysis in Android applications has been presented. Also twenty applications have been selected and have been analyzed with the developed system to look for private data exfiltration.

The developed system has been able to detect private data exfiltration in almost all of the applications, being the most characteristic private data to exfiltrate some kind of terminal identifier sent to some data analytics or marketing domain. This probably means the tracking of the user through the Android ecosystem.

The system has been developed with ease of use in mind and the user manual included in the thesis shows what any user would have to understand. The system should be able to be used by itself or to work with other different systems.

Also some limitations have been detected that should be overcome in future work. The one that stands out the most is the one that limits the detected information to elements that have an univocal representation as a character string.

Finally, based on the results of the analysis of the different applications it is safe to say that the objectives in section 1.2 have been accomplished.

6.2 Future work

There are three main areas where future work would be most useful to the development of traffic analysis for Android applications. Those are: tackling certificate pinning, finding more efficient ways to execute the target application and developing better systems for data detection.

Most of the target applications studied in this thesis had some connection which did not trust the CA certificate that was introduced in the phone. This means that many connections could not be analyzed properly, being a source of possible false negatives. Overcoming this certificate pinning will be very useful.

The target applications studied during the thesis were executed using a basic manual approach. This was enough for the number of applications and the context of the analysis but it is a major limitation in the case that great amounts of applications want to be analyzed. Finding some automated and intelligent execution approach would allow to analyze many more applications.

Finally, most types of information are not static or do not have an univocal representation as a string of characters. During the thesis the detection was based on string matching which allowed to detect many terminal identifiers but it is probably not enough to detect other type of data like user location. Finding other ways to detect private data in a connection would allow to understand better the risks associated with the usage of any Android application.

A Result data

School Planner

Phase	Data	Category	Domain	Port	Domain category	Country	Outside EEA	HTTPS

Class Dojo

Phase	Data	Category	Domain	Port	Domain category	Country	Outside EEA	HTTPS
Phase 1	androidid	Terminal id	api.classdojo.com	443	App	United States	Yes	Yes
Phase 2	form-email	Contact info	teach.classdojo.com	443	App	United States	Yes	Yes
Phase 2	form-email	Contact info	api.classdojo.com	443	App	United States	Yes	Yes
Phase 2	form-email	Contact info	teach.classdojo.com	443	App	United States	Yes	Yes
Phase 2	form-name	Contact info	teach.classdojo.com	443	App	United States	Yes	Yes
Phase 2	form-name	Contact info	teach.classdojo.com	443	App	United States	Yes	Yes
Phase 2	androidid	Terminal id	teach.classdojo.com	443	App	United States	Yes	Yes
Phase 2	androidid	Terminal id	teach.classdojo.com	443	App	United States	Yes	Yes
Phase 2	platform	Terminal id	classdojo-1.pubnub.com	80	-	Germany	No	No
Phase 2	platform	Terminal id	classdojo-1.pubnub.com	80	-	Germany	No	No
Phase 2	platform	Terminal id	classdojo-1.pubnub.com	80	-	Germany	No	No
Phase 2	platform	Terminal id	classdojo-1.pubnub.com	80	-	Germany	No	No
Phase 2	platform	Terminal id	classdojo-1.pubnub.com	80	-	Germany	No	No
Phase 2	platform	Terminal id	classdojo-1.pubnub.com	80	-	Germany	No	No
Phase 2	platform	Terminal id	pubsub-1.pubnub.com	80	-	Germany	No	No
Phase 2	platform	Terminal id	pubsub-1.pubnub.com	80	-	Germany	No	No
Phase 2	Certificate Pinning	-	pstatic.classdojo.com	-	App	United States	Yes	Yes
Phase 2	form-email	Contact info	teach.classdojo.com	443	App	United States	Yes	Yes
Phase 2	form-email	Contact info	teach.classdojo.com	443	App	United States	Yes	Yes
Phase 2	form-name	Contact info	teach.classdojo.com	443	App	United States	Yes	Yes
Phase 2	form-name	Contact info	teach.classdojo.com	443	App	United States	Yes	Yes
Phase 2	form-email	Contact info	api.classdojo.com	443	App	United States	Yes	Yes
Phase 2	Certificate Pinning	-	toolkit.classdojo.com	-	App	United States	Yes	Yes

Duolingo

[illegible]

Development of a system for traffic analysis of smartphone apps for private data exfiltration detection

[illegible]

Phase	Data	Category	Domain	Port	Domain category	Country	Outside EEA	HTTPS
Phase 2	aaaid	Terminal id	graph.facebook.com	443	Marketing	Ireland	No	Yes
Phase 2	aaaid	Terminal id	excess.duolingo.com	443	App	United States	Yes	Yes
Phase 2	platform	Terminal id	excess.duolingo.com	443	App	United States	Yes	Yes
Phase 2	form-name	Contact info	excess.duolingo.com	443	App	United States	Yes	Yes
Phase 2	aaaid	Terminal id	excess.duolingo.com	443	App	United States	Yes	Yes
Phase 2	platform	Terminal id	excess.duolingo.com	443	App	United States	Yes	Yes
Phase 2	form-name	Contact info	excess.duolingo.com	443	App	United States	Yes	Yes
Phase 2	aaaid	Terminal id	excess.duolingo.com	443	App	United States	Yes	Yes
Phase 2	platform	Terminal id	excess.duolingo.com	443	App	United States	Yes	Yes
Phase 2	form-name	Contact info	excess.duolingo.com	443	App	United States	Yes	Yes
Phase 2	aaaid	Terminal id	publisher-config.unityads.unity3d.com	443	Marketing	-	-	Yes
Phase 2	platform	Terminal id	publisher-config.unityads.unity3d.com	443	Marketing	-	-	Yes
Phase 2	buildid	Terminal id	auction.unityads.unity3d.com	443	Marketing	United States	Yes	Yes
Phase 2	aaaid	Terminal id	auction.unityads.unity3d.com	443	Marketing	United States	Yes	Yes
Phase 2	platform	Terminal id	auction.unityads.unity3d.com	443	Marketing	United States	Yes	Yes
Phase 2	form-email	Contact info	android-api.duolingo.com	443	App	United States	Yes	Yes
Phase 2	form-email	Contact info	android-api.duolingo.com	443	App	United States	Yes	Yes
Phase 2	form-email	Contact info	android-api.duolingo.com	443	App	United States	Yes	Yes
Phase 2	form-email	Contact info	android-api.duolingo.com	443	App	United States	Yes	Yes

Edmodo

Phase	Data	Category	Domain	Port	Domain category	Country	Outside EEA	HTTPS
Phase 2	form-email	Contact info	api.edmodo.com	443	App	United States	Yes	Yes
Phase 2	form-name	Contact info	api.edmodo.com	443	App	United States	Yes	Yes
Phase 2	form-name	Contact info	api.edmodo.com	443	App	United States	Yes	Yes
Phase 2	form-email	Contact info	api.edmodo.com	443	App	United States	Yes	Yes
Phase 2	form-name	Contact info	api.edmodo.com	443	App	United States	Yes	Yes
Phase 2	form-name	Contact info	api.edmodo.com	443	App	United States	Yes	Yes
Phase 2	form-email	Contact info	api.edmodo.com	443	App	United States	Yes	Yes
Phase 2	form-email	Contact info	api.edmodo.com	443	App	United States	Yes	Yes
Phase 2	form-name	Contact info	api.edmodo.com	443	App	United States	Yes	Yes
Phase 2	form-name	Contact info	api.edmodo.com	443	App	United States	Yes	Yes

Flipgrid

Phase	Data	Category	Domain	Port	Domain category	Country	Outside EEA	HTTPS
Phase 1	platform	Terminal id	gate.hockeyapp.net	443	Analytics	United Kingdom	No	Yes
Phase 1	Certificate Pinning	-	admin.flipgrid.com	-	App	United States	Yes	Yes
Phase 1	platform	Terminal id	gate.hockeyapp.net	443	Analytics	United States	Yes	Yes
Phase 2	Certificate Pinning	-	admin.flipgrid.com	-	App	United States	Yes	Yes
Phase 2	Certificate Pinning	-	secure.flipgrid.com	-	App	United States	Yes	Yes
Phase 2	Certificate Pinning	-	secure.flipgrid.com	-	App	United States	Yes	Yes
Phase 2	Certificate Pinning	-	ruby.flipgrid.com	-	App	United States	Yes	Yes
Phase 2	Certificate Pinning	-	ruby.flipgrid.com	-	App	United States	Yes	Yes
Phase 2	Certificate Pinning	-	ruby.flipgrid.com	-	App	United States	Yes	Yes
Phase 2	Certificate Pinning	-	secure.flipgrid.com	-	App	United States	Yes	Yes
Phase 2	Certificate Pinning	-	secure.flipgrid.com	-	App	United States	Yes	Yes
Phase 2	platform	Terminal id	gate.hockeyapp.net	443	Analytics	United Kingdom	No	Yes
Phase 2	Certificate Pinning	-	media.flipgrid.com	-	App	United States	Yes	Yes
Phase 2	Certificate Pinning	-	media.flipgrid.com	-	App	United States	Yes	Yes
Phase 2	Certificate Pinning	-	secure.flipgrid.com	-	App	United States	Yes	Yes
Phase 2	Certificate Pinning	-	media.flipgrid.com	-	App	United States	Yes	Yes
Phase 2	Certificate Pinning	-	media.flipgrid.com	-	App	United States	Yes	Yes
Phase 2	Certificate Pinning	-	media.flipgrid.com	-	App	United States	Yes	Yes
Phase 2	Certificate Pinning	-	media2.giphy.com	-	-	United States	Yes	Yes

Development of a system for traffic analysis of smartphone apps for private data exfiltration detection

Phase	Data	Category	Domain	Port	Domain category	Country	Outside EEA	HTTPS
Phase 2	Certificate Pinning	-	media2.giphy.com	-	-	United States	Yes	Yes
Phase 2	Certificate Pinning	-	media3.giphy.com	-	-	United States	Yes	Yes
Phase 2	Certificate Pinning	-	admin.flipgrid.com	-	App	United States	Yes	Yes
Phase 2	Certificate Pinning	-	ruby.flipgrid.com	-	App	United States	Yes	Yes
Phase 2	Certificate Pinning	-	static.flipgrid.com	-	App	United States	Yes	Yes
Phase 2	Certificate Pinning	-	ruby.flipgrid.com	-	App	United States	Yes	Yes
Phase 2	Certificate Pinning	-	ruby.flipgrid.com	-	App	United States	Yes	Yes
Phase 2	Certificate Pinning	-	static.flipgrid.com	-	App	United States	Yes	Yes
Phase 2	Certificate Pinning	-	static.flipgrid.com	-	App	United States	Yes	Yes
Phase 2	Certificate Pinning	-	static.flipgrid.com	-	App	United States	Yes	Yes
Phase 2	Certificate Pinning	-	static.flipgrid.com	-	App	United States	Yes	Yes
Phase 2	platform	Terminal id	gate.hockeyapp.net	443	Analytics	United States	Yes	Yes

Kahoot

Phase	Data	Category	Domain	Port	Domain category	Country	Outside EEA	HTTPS
Phase 1	platform	Terminal id	tap-nexus.appspot.com	443	-	United States	Yes	Yes
Phase 1	aaaid	Terminal id	api.amplitude.com	443	Analytics	United States	Yes	Yes
Phase 1	platform	Terminal id	api.amplitude.com	443	Analytics	United States	Yes	Yes
Phase 1	aaaid	Terminal id	api.amplitude.com	443	Analytics	United States	Yes	Yes
Phase 1	platform	Terminal id	api.amplitude.com	443	Analytics	United States	Yes	Yes
Phase 2	Certificate Pinning	-	create.kahoot.it	-	App	United States	Yes	Yes
Phase 2	aaaid	Terminal id	api.amplitude.com	443	Analytics	United States	Yes	Yes
Phase 2	platform	Terminal id	api.amplitude.com	443	Analytics	United States	Yes	Yes
Phase 2	Certificate Pinning	-	create.kahoot.it	-	App	United States	Yes	Yes
Phase 2	aaaid	Terminal id	api.amplitude.com	443	Analytics	United States	Yes	Yes
Phase 2	platform	Terminal id	api.amplitude.com	443	Analytics	United States	Yes	Yes
Phase 2	buildid	Terminal id	api.amplitude.com	443	Analytics	United States	Yes	Yes
Phase 2	platform	Terminal id	api.amplitude.com	443	Analytics	United States	Yes	Yes
Phase 2	buildid	Terminal id	api.amplitude.com	443	Analytics	United States	Yes	Yes
Phase 2	platform	Terminal id	api.amplitude.com	443	Analytics	United States	Yes	Yes
Phase 2	buildid	Terminal id	api.amplitude.com	443	Analytics	United States	Yes	Yes
Phase 2	platform	Terminal id	api.amplitude.com	443	Analytics	United States	Yes	Yes
Phase 2	buildid	Terminal id	api.amplitude.com	443	Analytics	United States	Yes	Yes
Phase 2	platform	Terminal id	api.amplitude.com	443	Analytics	United States	Yes	Yes
Phase 2	form-name	Contact info	create.kahoot.it	443	App	Ireland	No	Yes
Phase 2	form-email	Contact info	create.kahoot.it	443	App	Ireland	No	Yes
Phase 2	form-name	Contact info	create.kahoot.it	443	App	Ireland	No	Yes
Phase 2	form-name	Contact info	create.kahoot.it	443	App	Ireland	No	Yes
Phase 2	form-email	Contact info	create.kahoot.it	443	App	Ireland	No	Yes
Phase 2	form-name	Contact info	create.kahoot.it	443	App	Ireland	No	Yes
Phase 2	form-name	Contact info	create.kahoot.it	443	App	Ireland	No	Yes
Phase 2	form-name	Contact info	apis.kahoot.it	443	App	Spain	No	Yes
Phase 2	form-name	Contact info	create.kahoot.it	443	App	Ireland	No	Yes
Phase 2	form-email	Contact info	create.kahoot.it	443	App	Ireland	No	Yes
Phase 2	form-email	Contact info	create.kahoot.it	443	App	Ireland	No	Yes
Phase 2	form-name	Contact info	apis.kahoot.it	443	App	Spain	No	Yes
Phase 2	form-name	Contact info	apis.kahoot.it	443	App	Spain	No	Yes
Phase 2	buildid	Terminal id	api.amplitude.com	443	Analytics	United States	Yes	Yes
Phase 2	platform	Terminal id	api.amplitude.com	443	Analytics	United States	Yes	Yes
Phase 2	buildid	Terminal id	api.amplitude.com	443	Analytics	United States	Yes	Yes
Phase 2	platform	Terminal id	api.amplitude.com	443	Analytics	United States	Yes	Yes
Phase 2	form-name	Contact info	api.amplitude.com	443	Analytics	United States	Yes	Yes
Phase 2	aaaid	Terminal id	api.amplitude.com	443	Analytics	United States	Yes	Yes
Phase 2	platform	Terminal id	api.amplitude.com	443	Analytics	United States	Yes	Yes
Phase 2	form-name	Contact info	api.amplitude.com	443	Analytics	United States	Yes	Yes
Phase 2	aaaid	Terminal id	api.amplitude.com	443	Analytics	United States	Yes	Yes
Phase 2	platform	Terminal id	api.amplitude.com	443	Analytics	United States	Yes	Yes
Phase 2	aaaid	Terminal id	api.amplitude.com	443	Analytics	United States	Yes	Yes

[illegible]

[illegible]

Socrative

Phase	Data	Category	Domain	Port	Domain category	Country	Outside EEA	HTTPS
Phase 1	Certificate Pinning	-	b.socrative.com	-	App	United States	Yes	Yes
Phase 1	Certificate Pinning	-	b.socrative.com	-	App	United States	Yes	Yes
Phase 2	form-name	Contact info	api.socrative.com	443	App	United States	Yes	Yes

TokApp School

Phase	Data	Category	Domain	Port	Domain category	Country	Outside EEA	HTTPS
Phase 1	aaid	Terminal id	graph.facebook.com	443	Marketing	Ireland	No	Yes
Phase 1	aaid	Terminal id	graph.facebook.com	443	Marketing	Ireland	No	Yes
Phase 1	aaid	Terminal id	graph.facebook.com	443	Marketing	Ireland	No	Yes
Phase 1	aaid	Terminal id	graph.facebook.com	443	Marketing	Ireland	No	Yes
Phase 1	aaid	Terminal id	graph.facebook.com	443	Marketing	Ireland	No	Yes
Phase 1	aaid	Terminal id	graph.facebook.com	443	Marketing	Ireland	No	Yes
Phase 2	aaid	Terminal id	graph.facebook.com	443	Marketing	Ireland	No	Yes
Phase 2	form-phone	Contact info	graph.accountkit.com	443	Id provider	Ireland	No	Yes
Phase 2	form-phone	Contact info	graph.accountkit.com	443	Id provider	Ireland	No	Yes
Phase 2	form-phone	Contact info	graph.accountkit.com	443	Id provider	Ireland	No	Yes
Phase 2	form-phone	Contact info	graph.accountkit.com	443	Id provider	Ireland	No	Yes
Phase 2	form-phone	Contact info	graph.accountkit.com	443	Id provider	Ireland	No	Yes
Phase 2	form-phone	Contact info	graph.accountkit.com	443	Id provider	Ireland	No	Yes
Phase 2	form-phone	Contact info	graph.accountkit.com	443	Id provider	Ireland	No	Yes
Phase 2	form-phone	Contact info	graph.accountkit.com	443	Id provider	Ireland	No	Yes
Phase 2	form-phone	Contact info	graph.accountkit.com	443	Id provider	Ireland	No	Yes
Phase 2	aaid	Terminal id	graph.facebook.com	443	Marketing	Ireland	No	Yes
Phase 2	form-phone	Contact info	graph.accountkit.com	443	Id provider	Ireland	No	Yes
Phase 2	form-phone	Contact info	graph.accountkit.com	443	Id provider	Ireland	No	Yes

[illegible]

[illegible]

Google Fit

Phase	Data	Category	Domain	Port	Domain category	Country	Outside EEA	HTTPS
Phase 1	Certificate Pinning	-	android.clients.google.com	-	App	United States	Yes	Yes
Phase 2	Certificate Pinning	-	people-pa.googleapis.com	-	App	United States	Yes	Yes

Phase	Data	Category	Domain	Port	Domain category	Country	Outside EEA	HTTPS
Phase 2	Certificate Pinning	-	fit.googleapis.com	-	App	United States	Yes	Yes
Phase 2	Certificate Pinning	-	fit.googleapis.com	-	App	United States	Yes	Yes
Phase 2	Certificate Pinning	-	fit.googleapis.com	-	App	United States	Yes	Yes
Phase 2	Certificate Pinning	-	fit.googleapis.com	-	App	United States	Yes	Yes
Phase 2	Certificate Pinning	-	fit.googleapis.com	-	App	United States	Yes	Yes
Phase 2	Certificate Pinning	-	fit.googleapis.com	-	App	United States	Yes	Yes
Phase 2	Certificate Pinning	-	fit.googleapis.com	-	App	United States	Yes	Yes
Phase 2	Certificate Pinning	-	fit.googleapis.com	-	App	United States	Yes	Yes

Tools & Mi Band

Phase	Data	Category	Domain	Port	Domain category	Country	Outside EEA	HTTPS
Phase 2	Certificate Pinning	-	api-mifit-us.huami.com	-	-	United States	Yes	Yes

Mi Fit

Phase	Data	Category	Domain	Port	Domain category	Country	Outside EEA	HTTPS
Phase 2	Certificate Pinning	-	api-mifit.huami.com	-	App	United States	Yes	Yes
Phase 2	Certificate Pinning	-	account.xiaomi.com	-	App	Singapore	Yes	Yes
Phase 2	routermac	Location info	account.huami.com	443	App	United States	Yes	Yes
Phase 2	routermac	Location info	account.huami.com	443	App	United States	Yes	Yes
Phase 2	Certificate Pinning	-	account.xiaomi.com	-	App	Singapore	Yes	Yes
Phase 2	routermac	Location info	account.huami.com	443	App	United States	Yes	Yes
Phase 2	form-name	Contact info	api-mifit-us.huami.com	443	App	United States	Yes	Yes
Phase 2	form-name	Contact info	api-mifit-us.huami.com	443	App	United States	Yes	Yes
Phase 2	form-name	Contact info	api-mifit-us.huami.com	443	App	United States	Yes	Yes
Phase 2	form-name	Contact info	api-mifit-us.huami.com	443	App	United States	Yes	Yes
Phase 2	aaaid	Terminal id	graph.facebook.com	443	Marketing	Ireland	No	Yes
Phase 2	aaaid	Terminal id	graph.facebook.com	443	Marketing	Ireland	No	Yes
Phase 2	aaaid	Terminal id	graph.facebook.com	443	Marketing	Ireland	No	Yes
Phase 2	aaaid	Terminal id	graph.facebook.com	443	Marketing	Ireland	No	Yes

Runtastic Free

Phase	Data	Category	Domain	Port	Domain category	Country	Outside EEA	HTTPS
Phase 1	platform	Terminal id	mobile-collector.newrelic.com	443	Analytics	United States	Yes	Yes
Phase 1	aaaid	Terminal id	graph.facebook.com	443	Marketing	Ireland	No	Yes
Phase 1	aaaid	Terminal id	graph.facebook.com	443	Marketing	Ireland	No	Yes
Phase 1	aaaid	Terminal id	graph.facebook.com	443	Marketing	Ireland	No	Yes
Phase 1	aaaid	Terminal id	graph.facebook.com	443	Marketing	Ireland	No	Yes
Phase 1	aaaid	Terminal id	graph.facebook.com	443	Marketing	Ireland	No	Yes
Phase 1	aaaid	Terminal id	graph.facebook.com	443	Marketing	Ireland	No	Yes
Phase 1	aaaid	Terminal id	graph.facebook.com	443	Marketing	Ireland	No	Yes
Phase 1	platform	Terminal id	mobile-collector.newrelic.com	443	Analytics	United States	Yes	Yes
Phase 1	platform	Terminal id	mobile-collector.newrelic.com	443	Analytics	United States	Yes	Yes
Phase 1	platform	Terminal id	mobile-collector.newrelic.com	443	Analytics	United States	Yes	Yes
Phase 1	aaaid	Terminal id	app.adjust.com	443	Marketing	Germany	No	Yes
Phase 1	buildid	Terminal id	app.adjust.com	443	Marketing	Germany	No	Yes
Phase 1	platform	Terminal id	app.adjust.com	443	Marketing	Germany	No	Yes
Phase 1	aaaid	Terminal id	app.adjust.com	443	Marketing	Germany	No	Yes
Phase 1	buildid	Terminal id	app.adjust.com	443	Marketing	Germany	No	Yes

[illegible]

Development of a system for traffic analysis of smartphone apps for private data exfiltration detection

[illegible]

Samsung Health

Phase	Data	Category	Domain	Port	Domain category	Country	Outside EEA	HTTPS
Phase 2	platform	Terminal id	api.samsungknowledge.com	443	App	United States	Yes	Yes
Phase 2	platform	Terminal id	collector.samsungknowledge.com	443	App	United States	Yes	Yes
Phase 2	platform	Terminal id	collector.samsungknowledge.com	443	App	United States	Yes	Yes
Phase 2	platform	Terminal id	collector.samsungknowledge.com	443	App	United States	Yes	Yes

7 Minute Workout

Phase	Data	Category	Domain	Port	Domain category	Country	Outside EEA	HTTPS

Step Tracker

Phase	Data	Category	Domain	Port	Domain category	Country	Outside EEA	HTTPS
Phase 1	aaaid	Terminal id	graph.facebook.com	443	Marketing	Ireland	No	Yes
Phase 1	aaaid	Terminal id	t.appsflyer.com	443	Analytics	Ireland	No	Yes
Phase 1	buildid	Terminal id	t.appsflyer.com	443	Analytics	Ireland	No	Yes
Phase 1	aaaid	Terminal id	graph.facebook.com	443	Marketing	Ireland	No	Yes
Phase 1	aaaid	Terminal id	graph.facebook.com	443	Marketing	Ireland	No	Yes
Phase 1	aaaid	Terminal id	graph.facebook.com	443	Marketing	Ireland	No	Yes
Phase 1	platform	Terminal id	eu.ime.cooteck.com	443	-	Germany	No	Yes
Phase 1	aaaid	Terminal id	events.appsflyer.com	443	Analytics	Ireland	No	Yes
Phase 1	buildid	Terminal id	events.appsflyer.com	443	Analytics	Ireland	No	Yes
Phase 1	aaaid	Terminal id	eu.ime.cooteck.com	443	-	Germany	No	Yes
Phase 1	platform	Terminal id	eu.ime.cooteck.com	443	-	Germany	No	Yes
Phase 1	aaaid	Terminal id	events.appsflyer.com	443	Analytics	Ireland	No	Yes
Phase 1	buildid	Terminal id	events.appsflyer.com	443	Analytics	Ireland	No	Yes
Phase 1	aaaid	Terminal id	events.appsflyer.com	443	Analytics	Ireland	No	Yes
Phase 1	buildid	Terminal id	events.appsflyer.com	443	Analytics	Ireland	No	Yes
Phase 1	aaaid	Terminal id	eu.ime.cooteck.com	443	-	Germany	No	Yes
Phase 1	aaaid	Terminal id	events.appsflyer.com	443	Analytics	Ireland	No	Yes
Phase 1	buildid	Terminal id	events.appsflyer.com	443	Analytics	Ireland	No	Yes
Phase 1	aaaid	Terminal id	eu.ime.cooteck.com	443	-	Germany	No	Yes
Phase 1	aaaid	Terminal id	events.appsflyer.com	443	Analytics	Ireland	No	Yes
Phase 1	buildid	Terminal id	events.appsflyer.com	443	Analytics	Ireland	No	Yes
Phase 1	aaaid	Terminal id	events.appsflyer.com	443	Analytics	Ireland	No	Yes
Phase 1	buildid	Terminal id	events.appsflyer.com	443	Analytics	Ireland	No	Yes
Phase 1	aaaid	Terminal id	events.appsflyer.com	443	Analytics	Ireland	No	Yes
Phase 1	platform	Terminal id	eu.ime.cooteck.com	443	-	Germany	No	Yes
Phase 1	aaaid	Terminal id	events.appsflyer.com	443	Analytics	Ireland	No	Yes
Phase 1	buildid	Terminal id	events.appsflyer.com	443	Analytics	Ireland	No	Yes
Phase 1	aaaid	Terminal id	events.appsflyer.com	443	Analytics	Ireland	No	Yes
Phase 1	buildid	Terminal id	events.appsflyer.com	443	Analytics	Ireland	No	Yes
Phase 1	aaaid	Terminal id	events.appsflyer.com	443	Analytics	Ireland	No	Yes
Phase 1	buildid	Terminal id	events.appsflyer.com	443	Analytics	Ireland	No	Yes
Phase 1	aaaid	Terminal id	events.appsflyer.com	443	Analytics	Ireland	No	Yes
Phase 1	platform	Terminal id	eu.ime.cooteck.com	443	-	Germany	No	Yes
Phase 1	platform	Terminal id	eu.ime.cooteck.com	443	-	Germany	No	Yes
Phase 1	aaaid	Terminal id	t.appsflyer.com	443	Analytics	Ireland	No	Yes
Phase 1	buildid	Terminal id	t.appsflyer.com	443	Analytics	Ireland	No	Yes
Phase 1	aaaid	Terminal id	register.appsflyer.com	443	Analytics	Ireland	No	Yes
Phase 1	aaaid	Terminal id	register.appsflyer.com	443	Analytics	Ireland	No	Yes
Phase 1	aaaid	Terminal id	events.appsflyer.com	443	Analytics	Ireland	No	Yes
Phase 1	buildid	Terminal id	events.appsflyer.com	443	Analytics	Ireland	No	Yes

[illegible]

Development of a system for traffic analysis of smartphone apps for private data exfiltration detection

[illegible]

Development of a system for traffic analysis of smartphone apps for private data exfiltration detection

[illegible]

Development of a system for traffic analysis of smartphone apps for private data exfiltration detection

[illegible]

[illegible]

Development of a system for traffic analysis of smartphone apps for private data exfiltration detection

[illegible]

Development of a system for traffic analysis of smartphone apps for private data exfiltration detection

Phase	Data	Category	Domain	Port	Domain category	Country	Outside EEA	HTTPS
Phase 1	aaid	Terminal id	api.branch.io	443	Analytics	United States	Yes	Yes
Phase 1	platform	Terminal id	api.branch.io	443	Analytics	United States	Yes	Yes
Phase 2	Certificate Pinning	-	m.strava.com	-	App	United States	Yes	Yes
Phase 2	Certificate Pinning	-	m.strava.com	-	App	United States	Yes	Yes
Phase 2	Certificate Pinning	-	m.strava.com	-	App	United States	Yes	Yes
Phase 2	Certificate Pinning	-	analytics.strava.com	-	App	United States	Yes	Yes
Phase 2	Certificate Pinning	-	analytics.strava.com	-	App	United States	Yes	Yes
Phase 2	Certificate Pinning	-	analytics.strava.com	-	App	United States	Yes	Yes
Phase 2	platform	Terminal id	api.branch.io	443	Analytics	United States	Yes	Yes
Phase 2	form-email	Contact info	api.iterable.com	443	Marketing	United States	Yes	Yes
Phase 2	platform	Terminal id	api.branch.io	443	Analytics	United States	Yes	Yes
Phase 2	aaid	Terminal id	app.adjust.com	443	Marketing	Germany	No	Yes
Phase 2	platform	Terminal id	app.adjust.com	443	Marketing	Germany	No	Yes
Phase 2	aaid	Terminal id	app.adjust.com	443	Marketing	Germany	No	Yes
Phase 2	platform	Terminal id	app.adjust.com	443	Marketing	Germany	No	Yes
Phase 2	platform	Terminal id	api.branch.io	443	Analytics	United States	Yes	Yes
Phase 2	form-email	Contact info	api.iterable.com	443	Marketing	United States	Yes	Yes
Phase 2	form-email	Contact info	api.iterable.com	443	Marketing	United States	Yes	Yes
Phase 2	form-email	Contact info	api.iterable.com	443	Marketing	United States	Yes	Yes

SyncMyTracks

Phase	Data	Category	Domain	Port	Domain category	Country	Outside EEA	HTTPS
Phase 2	platform	Terminal id	hub.samsungapps.com	80	-	Ireland	No	No
Phase 2	platform	Terminal id	hub.samsungapps.com	80	-	Ireland	No	No
Phase 2	Certificate Pinning	-	appws.runtastic.com	-	-	Austria	No	Yes
Phase 2	platform	Terminal id	mobile-collector.newrelic.com	443	Analytics	United States	Yes	Yes

B Best practices

For developers of Android applications it is useful to have some guidance on the development of secure and privacy aware apps. Based on the analysis of the selected applications there are some best practices that can be uncovered.

The most basic practice is the usage of HTTPS on all of the connections. Any connection that is not encrypted is vulnerable to be read by anyone in the network and any private data will be exposed. Before implementing other measures any developer should make sure that connections are secure.

The next best practice is the usage of certificate pinning in the connections. This will stop the techniques used throughout the thesis to vulnerate the encryption of the target connections. The only way to read the connections with the usage of certificate pinning is to modify the application.

To protect and comply with the protections of the privacy of any user it is also recommended to avoid creating any connection that it is not necessary or that contains personal data after the user consents to the privacy policy. In other words avoid as far as possible connections in the phase one.

It is also important to consider which libraries and pieces of personal information are really necessary. Many libraries create connections exfiltrating private data without the knowledge of the developer. Also, any information that is not collected is less information to protect.

Finally, it is not recommended to establish the application's servers in a country outside of the European Economic Area. With the recent data protection regulations the EU gives a much better environment for the protection of user's data.

Bibliography

- [1] B. O. del Estado. (2018). Ley orgánica 3/2018, de 5 de diciembre, de protección de datos personales y garantía de los derechos digitales., [Online]. Available: <https://boe.es/boe/dias/2018/12/06/pdfs/BOE-A-2018-16673.pdf>.
- [2] EDPS. (2018). The history of the general data protection regulation, [Online]. Available: https://edps.europa.eu/data-protection/data-protection/legislation/history-general-data-protection-regulation_en.
- [3] Y. Song and U. Hengartner, "Privacyguard: A vpn-based platform to detect information leakage on android devices," in *Proceedings of the 5th Annual ACM CCS Workshop on Security and Privacy in Smartphones and Mobile Devices*, ACM, 2015, pp. 15–26.
- [4] A. Razaghpanah, R. Nithyanand, N. Vallina-Rodriguez, S. Sundaresan, M. Allman, C. Kreibich, and P. Gill, "Apps, trackers, privacy, and regulators: A global study of the mobile tracking ecosystem," 2018.
- [5] A. Le, J. Varmarken, S. Langhoff, A. Shuba, M. Gjoka, and A. Markopoulou, "Antmonitor: A system for monitoring from mobile devices," in *Proceedings of the 2015 ACM SIGCOMM Workshop on Crowdsourcing and Crowdfunding of Big (Internet) Data*, ACM, 2015, pp. 15–20.
- [6] J. Ren, A. Rao, M. Lindorfer, A. Legout, and D. Choffnes, "Recon: Revealing and controlling pii leaks in mobile network traffic," in *Proceedings of the 14th Annual International Conference on Mobile Systems, Applications, and Services*, ACM, 2016, pp. 361–374.
- [7] (2019). Genymotion, [Online]. Available: <https://www.genymotion.com/>.
- [8] (2019). Run apps on the android emulator, [Online]. Available: <https://developer.android.com/studio/run/emulator>.
- [9] (2019). Android x86, [Online]. Available: <https://www.android-x86.org/>.

- [10] J. M. del Álamo Ramiro, D. S. Guamán, L. María-Tomé Laverón, and E. Zamora Beamud. (2018). Análisis de los flujos de información en android. herramientas para el cumplimiento de la responsabilidad proactiva., [Online]. Available: <https://www.aepd.es/media/estudios/estudio-flujos-informacion-android.pdf>.
- [11] A. Developers. (). Ui/application exerciser monkey, [Online]. Available: <https://developer.android.com/studio/test/monkey>.
- [12] M. Diamantaris, E. P. Papadopoulos, E. P. Markatos, S. Ioannidis, and J. Polakis, "Reaper: Real-time app analysis for augmenting the android permission system," 2019.
- [13] J. Forshaw, *Attacking network protocols: a hackers guide to capture, analysis, and exploitation*. No Starch Press, 2018.
- [14] Netscape. (1996). Ssl version 3.0, [Online]. Available: <https://web.archive.org/web/19970614020952/http://home.netscape.com/newsref/std/SSL.html>.
- [15] A. Developers. (). Security with https and ssl, [Online]. Available: <https://developer.android.com/training/articles/security-ssl>.
- [16] A. S. t. Chad Brubaker. (2016). Changes to trusted certificate authorities in android nougat, [Online]. Available: <https://android-developers.googleblog.com/2016/07/changes-to-trusted-certificate.html>.
- [17] E. Pan, J. Ren, M. Lindorfer, C. Wilson, and D. Choffnes, "Panoptispy: Characterizing audio and video exfiltration from android applications," *Proceedings on Privacy Enhancing Technologies*, vol. 2018, no. 4, pp. 33–50, 2018.
- [18] (2019). Anbox, [Online]. Available: <https://anbox.io/>.
- [19] (2018). Mitmproxy documentation, [Online]. Available: <https://docs.mitmproxy.org/stable/>.
- [20] (2019). Network security configuration, [Online]. Available: <https://developer.android.com/training/articles/security-config>.
- [21] serializethoughts. (2016). Intercepting https traffic of android nougat applications, [Online]. Available: <https://serializethoughts.com/2016/09/10/905/>.
- [22] (2019). Tcpdump, [Online]. Available: <http://www.tcpdump.org/>.
- [23] (2019). Sslldump, [Online]. Available: <https://linux.die.net/man/1/ssldump>.
- [24] (2019). Wireshark, [Online]. Available: <https://www.wireshark.org/>.
- [25] (2019). Frida, [Online]. Available: <https://www.frida.re/>.

- [26] (2019). Apktool, [Online]. Available: <https://ibotpeaches.github.io/Apktool/>.
- [27] (2019). Import root cert, [Online]. Available: http://wiki.cacert.org/FAQ/ImportRootCert#Android_Phones_.26_Tablets.
- [28] (), [Online]. Available: <https://forum.xda-developers.com/redmi-note-5-pro/how-to/solved-bypassing-360h-wait-to-unlock-t3791083>.
- [29] (), [Online]. Available: <https://techjeep.com/2016/03/02/change-colours-twrp-3-0-recovery/>.
- [30] (), [Online]. Available: <https://github.com/io-reyes/appcensus-devfilegen>.