الجمـــهوريـــة الجـــزائريـــة الديـمقـراطيـــــــة الشـــعبيـــــة
PEOPLE'S DEMOCRATIC REPUBLIC OF ALGERIA
وزارة التعـلـيـم الـعـالـــي و الـبـحـث الـعـلـمي
MINISTRY OF HIGHER EDUCATION AND SCIENTIFIC RESEARCH
جـامـعـة عمّـــار ثـلـيـــجـــي بالأغـــواط
UNIVERSITY OF AMAR TELIDJI LAGHOUAT
كـلـيـــــة الـعـلـــــوم
FACULTY OF SCIENCES
COMPUTER SCIENCE DEPARTMENT

MASTER THESIS

Domain : Mathematics and Computer Science
Field : Computer Science
Option : Networks, Systems and Distributed Applications

Par:
HAMDI Imane

THEME

Permission Analysis of Mobile Applications

Thesis defended publicly on *07/07/2021*

Jury Members

Dr. Kerrache Chaker Abdelaziz	*Associate Professor*	President
Dr. Bensaad Lahcen Mohamed	*Associate Professor*	Examiner
Dr. BENAROUS Leila	*Assistant Professor*	Supervisor

N …… University Year 2020/2021

Dedication

To my parents.
To my brother and sisters.
To my nephews.

Acknowledgment

First and foremost, I would like to thank Allah the great for blessing my time and efforts and for enabling me to finish this work.

I would like also to thank my supervisor Dr. BENAROUS Leila who has entrusted me by supervising me in this work. I would like to thank her for being so gentle and for the advice she gave me during the fulfillment of this thesis.

I would like to express my gratitude to Dr. Kerrache Chaker Abdelaziz, the president of the jury and Dr. Bensaad Lahcen Mohamed who have accepted to evaluate this modest work and enhance its quality.

I am so grateful to Dr. Ameur Mohamed El Amine, for his availability and help.

ملخص

ظهرت تطبيقات الهاتف المحمول مع ظهور الهواتف الذكية. يتم تثبيت العديد من التطبيقات في هواتف المستخدمين وفقًا لاهتماماتهم وتلبية احتياجاتهم اليومية الأساسية، وتتوفر متاجر التطبيقات للسماح للمستخدمين بتنزيلها وتثبيتها على أجهزته.

قد تكون متاجر التطبيقات منصة للمهاجمين، ويمكنهم إخفاء البرامج الضارة وأحصنة طروادة الخبيثة في التطبيقات وتحميلها ، وفقًا لهذا الافتراض ، يمكن لمستخدمي الهاتف المحمول تنزيل التطبيقات دون التشكيك في مستوى الأمان لديهم. تتطلب التطبيقات المحمولة بعض التراخيص ليتم منحها قبل التثبيت. يتعين على المستخدمين إما منح هذه الاخيرة إلى التطبيق ومتابعة التثبيت أو رفض هذه الأذونات وإلغاء تثبيت التطبيق. يمكن منح تراخيص أخرى بعد تثبيت التطبيق و يمكن تعطيلها من قبل المستخدمين.

قد تكون هناك حاجة لبعض هذه التراخيص للحصول على الوظائف الصحيحة للتطبيق وفي أوقات أخرى لا تتعلق حتى باحتياجات تشغيل التطبيق. قد تختلف مخاطر الأذونات الممنوحة من انتهاك الخصوصية وتداول البيانات إلى التحكم في الهاتف واستخدامه كروبوتات في الأعمال الإجرامية على حد سواء الجرائم الإلكترونية والجسدية.

الهدف من هذه الأطروحة هو 1) توعية المستخدمين بالمخاطر المتعلقة بمنح التراخيص وحالات إساءة الاستخدام الخاصة بهم، 2) التحقق من وصول التراخيص لـ 17 تطبيقًا معروفًا للهاتف، 3) إنشاء تطبيق من شأنه التحقق من وصول التراخيص المحفوفة بالمخاطر للتطبيقات المثبتة وتمكن المستخدم من منعها.

الكلمات الرئيسية: التطبيقات الهاتفية، تحليل التراخيص، التراخيص الخطرة، الأفعال الإجرامية، وانتهاك الخصوصية، botnet ، الاتجار البيانات.

Abstract

Mobile phone applications emerged with the introduction of smartphones. Various applications are installed in users' phones following their interests and satisfying their basic daily needs, app stores are available to allow users to download and install them on their devices.

App stores may be a platform for attackers, they can hide malwares and malicious Trojan in applications and upload them, under such assumption mobile phone users can download applications without questioning their level of security .mobile apps require some permission to be granted before being installed. Users have either to grant these accesses to the app and proceed with the installation or reject these permissions and abort the installation of the app. Other permissions can be granted after the installation of the app and users. can disable them.

Some of these permissions may be needed for the correct functionality of the application and other times are not even related to the app functioning needs. The risks of the granted permissions may vary from privacy violation and data trading to controlling and using the phone as botnet in criminal acts both cyber and physical crimes.

The aim of this thesis is 1) enlighten users about the risks related to permission grants and their misuse cases, 2) investigate the permission accesses of 17 well known phone applications, 3) establish an application that would check the risky permission accesses of installed applications and enables the user to prevent them.

Key-words: mobile applications, permissions analysis, risky permissions, criminal acts, privacy violation, botnet, data trading.

Résumé

Les applications mobiles ont émergé avec l'introduction des smartphones. Diverses applications sont installées sur les téléphones des utilisateurs en fonction de leurs intérêts et répondant à leurs besoins quotidiens de base, des magasins d'applications sont disponibles pour permettre aux utilisateurs de les télécharger et de les installer sur leurs appareils.

Les magasins d'applications peuvent être une plate-forme pour les attaquants, ils peuvent masquer les logiciels malveillants et les chevaux de Troie malveillants dans les applications et les télécharger, dans une telle hypothèse, les utilisateurs de téléphones mobiles peuvent télécharger des applications sans remettre en question leur niveau de sécurité. Les applications mobiles nécessitent des autorisations avant d'être installées. Les utilisateurs doivent soit accorder ces accès à l'application et procéder à l'installation, soit rejeter ces autorisations et interrompre l'installation de l'application. D'autres autorisations peuvent être accordées après l'installation de l'application et des utilisateurs. peut les désactiver.

Certaines de ces autorisations peuvent être nécessaires pour le bon fonctionnement de l'application et d'autres fois ne sont même pas liées aux besoins de fonctionnement de l'application. Les risques des autorisations accordées peuvent aller de la violation de la vie privée et du commerce de données au contrôle et à l'utilisation du téléphone comme botnet dans le cadre d'actes criminels, à la fois cybernétique et physique.

L'objectif de cette thèse est 1) d'éclairer les utilisateurs sur les risques liés aux octrois d'autorisations et à leurs cas d'utilisation abusive, 2) d'enquêter sur les accès aux autorisations de 17 applications téléphoniques bien connues, 3) d'établir une application qui vérifierait les accès aux autorisations risqués des applications installées et permet à l'utilisateur de les empêcher.**Mots-clés** : applications mobiles, analyse des autorisations, autorisations dangereuses, actes criminels, violation de la confidentialité, botnet, le commerce de données.

Table of Content

List of figures

List of tables

Introduction

Mobile phones are becoming a necessity rather than a luxury. We rely on them to organize and satisfy basic daily needs. They are no longer used for calling only; their uses are extended to provide internet access, to do calculations, to run various service applications related to banking, to electronic payments and to infotainments.

The fast spread of mobile applications is tied to the evolution of smart phone technology as well as the diversity of users' needs. However, these mobile applications although are necessary and facilitate the execution of daily life tasks, they may also be a source and risk just like any technology. There have been different complaints from applications users regarding the noticing of their phones acting weirdly post installation of some applications such as being slow or non-responsive, their batteries drain faster, their network data is over consumed or their temperature rises quickly (heating issues). Some even claimed that the advertisement popping in their phones are targeted to the one using it, favoring the theory that their applications are sending inappropriate ads basing on the gender and age of users which is likely due to the behavior or face recognitions usage. The users also notice that when using specific apps or games, ads promoting similar apps will continue to appear. Some ads appear in specific period of time and continue to do so every month [2]. These claims hint on a serious issue which is that some of the mobile applications are collecting and using data of the users; they may even steal data, and control and exploit the phone.

The aim of this thesis is to follow a scientific process to investigate the security of mobile applications in terms of privacy violation, botnet exploitation and targeted ads emissions. For this, we analyze the permissions granted to the applications post installation. These permissions are the ones granting the access to sensitive phone resources, such as the camera, microphone, data storage, network interfaces, privileged phone access and control. Mobile app developers have to state these permissions in the application for it to get the needed access. Although they are obliged to mention them in app stores when deploying the application there, they may deliberately not mention about accesses to permissions not related to the application functionality to avoid alerting the users, while other developers intend to mention them to clean their responsibility and hold it on the user, as s/he was aware of the required permissions and granted them regardless of their related risks. Since the application does not give much choice to the users, they tend to accept these accesses without reading them, just because they need to use the application. Some users are not even aware of the related risks. The main target of this

thesis is to enlighten the users about these risks and help him/her detect them and protect against them either by disabling the dangerous permission post the installation of the app or by uninstalling risky apps.

The thesis will be organized on two parts with two chapters each, the first contains the literature review and the second explains our contribution. The first part is organized as follows:

Chapter 01 entitled mobile applications will help the reader understand more about the mobile applications, mobile operating systems, and the building process of a mobile application.

Chapter 02 entitled security and privacy risks which explains the security related mobile applications risk, scandals and a review on realized works to detect and to protect against them.

The second part is related to our contribution which is organized as follows:

Chapter 03 entitled risk analysis for android mobile applications permissions in which we investigate 17 commonly used applications using both the static and dynamic approaches in order to study their permission-related risks.

Chapter 04 entitled permission checker application in which we explain the building process of our developed application to check risky permissions and enable the users to disable/enable them basing on his/her needs.

Part I
LITERATURE REVIEW

CHAPTER 1

SECURITY RISKS OF MOBILE APPLICATION

1.1 Introduction

Mobile apps aims to facilitate the user's daily tasks, by delivering quick results. According to studies, there were 5.11 billion unique mobile users worldwide in 2019, and 2.71 billion of them use smartphones [3]. Most users are unaware of the security and privacy threats that mobile application brings. They are unconscious about the amount of risks engendered from the collection of their real time geolocation data, allowing access to their microphone, camera, credit card information or bank account details. Not to forget the health information such as user's vitals, fitness training and sleeping patterns. Symantec [4] showed 57 % of adult users are unaware of the existence of security issues for mobile devices. Reports also show that half of the attacks are intended to steal personal information and track users, these applications aim to obtain sensitive data [5]. Therefore, they must be protected from unauthorized access. Mobile security is centered around the security of its software and applications across various operating systems such as: Android, iOS, and Windows Phone.

1.2 Formal definition of a mobile app

Mobile applications allows users to interact with each other's, mobile apps can be used in multiple areas. Today, mobile apps are getting more and more attention. Mobile apps are just applications that run on a mobile device. Mobile apps are currently rapidly evolving. However, they have specific common characteristics such as: being developed for a special task, running on a mobile device, consuming network data [6].

1.3 Security breaches in mobile apps: examples

The security of our smartphone and data has been the concern of researchers and users spatially after scandals that reveals a privacy violation and infected devices using malwares. The major issue of privacy is permissions abuse and exploitation of users' data for financial gains and other purposes. Mobile apps are the new weak link for privacy and user data abuse. In a recent study concerning permissions usage among VPN apps on the Android Play Store, it was observed that more than 60% of these apps require "dangerous" permissions such as camera, location and managing the user external storage, that are not needed for their function. Researchers from Oxford University found that 90% of free apps on the Google Play store share data with organizations. [7]

In table 1.1, we will see some of the biggest recent mobile security breaches categorized by app, its security issue, details and scale of damage. [8]

Table 1.1 : Some of the largest app related to data breaches *[8]*

App name	Security issue	Details	Scale of damage
WhatsApp	Malware injection through insecure call function	Contained a vulnerability in its VOIP function that allowed attackers to inject malware into the victim's device by calling their phone.	-
Walgreens	Customers could see each other's messages.	The mobile security breach exposed private information, including names, prescription numbers and medication names, store numbers, and shipping addresses.	-
MyFitnessPal app	Mobile security breach reveals user details and passwords.	The app was hacked and most of the passwords had been encrypted.	150 million users had been affected.
Wishbone	Hack revealing data	Exposure of data related to the young users.	40 million account data were stolen
CoronApp-Colombia	Transmission of both personal health information and personally identifiable information in clear	Tracks symptoms related to COVID-19 Colombia. The app only used HTTP for communication. Data was captured by third parties using man in the middle attack.	Impacted more than 100,000 users

1.4 Android Apps VS iOS apps security

App security issues are the concern of not only the mobile users but also the software developers. The two famous mobile OS nowadays is Android and iPhone Operating System (iOS). Table 1.2 is an outline of the difference between Android and iOS based on security features: [9]

Table 1. 2: Difference between Android and iOS based on security features

Features	Comparison of Security in Mobile OS	
	Android	iOS
Application Sandboxing	Each app has its own sandbox	All apps shared same sandbox
Memory randomization	Fully applied in Jelly Bean Release, later than iOS.	Already applied in 4.3 releases.
Code signing Technology	Not used	Applied
Encryption	Disk encryption	Hardware encryption
Data Storage Format	Have an external storage and can be accessible by unwanted code	No external storage and difficult for the unwanted code to access storage
Antivirus	Antivirus can be downloaded from the Google store. Easier virus attack since no protection and checking is done before outside web source application been downloaded	No antivirus is required since the checking is done in the Apps Store.

iOS may be considered more secure, but it is not impossible for cybercriminals to hack iPhones or iPads. The owners of both Android and iOS devices need to be aware of possible malware and viruses, and be careful when downloading apps from third-party app sources. It is assumed to be safe to download apps from trusted sources, such as Google Play and Apple Store. However, regardless of how safe the applications are, they may still be vulnerable to social engineering attacks, in which cybercriminals trick users into giving their login information, access to bank accounts, and other personal data. Both iOS and Android are equally vulnerable to these types of attacks.

1.5 Permissions in Android apps

Android security is built upon a permission-based mechanism. A permission is simply a unique text string, which can be defined by Android or third-party developers. These permissions are found in the manifest file of the application's package. The application declares the required permissions to ensure its correct functionality, and the security of its own components and resources. The permission-based mechanism is criticized by developers, marketers, and end-users for its application's permissions control and management policy.

In the current Android permission framework, accesses to critical resources on smartphones are controlled according to permissions given to applications at install time. Many applications tend to request more permissions than their needs. The applications impose the principal of ***the all or the nothing***, i.e. a user has to either grant all permissions an application requests or is obliged to abort the installation process. Upon installing the application, its granted permissions are kept during its whole lifetime without the need to request them again at run-time. Another type of permission exist: the runtime permissions, which are permission, requested during the usage of the application.

Recently researchers observed permissions misuse and abuse by malicious developers whom took advantage of permission grants to exploit the smartphone resources and violate the users' privacy. Some of them focused on mobile botnet and malware apps detection, noting that botnets are networks of hijacked computer devices used to transmit various and malwares, the bots serve as a tool to automate attacks, such as data stealing, server crashing, and malware spreading. [10]

An example of permission abuse is a calculator app requesting SEND_SMS or

READ_SMS, is a malicious app, because this app neither do it need to send SMS nor to read it considering its normal functionality which the calculation. Another example is a game application that requests for the SEND_SMS permission, which is it does not need for its functioning; the permission may be exploited to send messages without users' knowledge. The content of message may vary from exporting (leaking) user's data to sending threats. This breaks the Principle of Least Privilege (PLP). [11]

1.6 Literature review on permission analysis

As the android app threats continue to increase, researchers are exploring a variety of security strategies for mobile platforms to detect mobile malware. Some work focuses on the analysis, detection, and evaluation of malicious applications. Researchers are starting to question the permission granting process being non-selective, they are also enquiring about the basis of application's permissions requests and demanding play stores to verify the permissions requested by each application and its correspondence with its intended functionality.

According to researchers [12] permission analysis is done using three approaches: static, dynamic, or hybrid analyses.

1.6.1 The static approach

The app is analyzed the application without running it: extract some features (Permissions and API calls) from the source code (AndroidManifest File) and examine them. The main goal is to check the code of the potentially malicious app, and its suspicious functionality and behaviors. It is not only capable of detecting unknown malware, but also of discover potential vulnerabilities in the source code. This approach has been implemented by many researchers. Such as: Z. Abdullah and M. Mohd Saudi [12], when they proposed a risk assessment method to evaluate the level of android Apps risks(very low, low, moderate, high, and very high) in term of privacy, integrity, and availability using the static approach. It was also used by M. Mohd Saudi, et al. [13] to develop a new botnet classification based on permissions and API calls which can be used as an input for mobile botnet detection. Some researchers used the comparison of permissions usage by malware apps and benign apps to classify applications. While others focused on designs meant to improve data security for example, controlling permission usage or isolating the execution environment.

1.6.2 The dynamic approach

The app is analyzed during run time, for any suspicious behavior. As an example of the dynamic approach, the Access dots app is developed to emulate the security policy followed by iOS of Apple. It adds to Android phones, a feature that can detect phones' resources usage without the knowledge of the user, by adding an orange indicator in the upper corner of the screen in case of microphone use (see figure 1.1), and a green one in case of active camera (see figure 1.2).

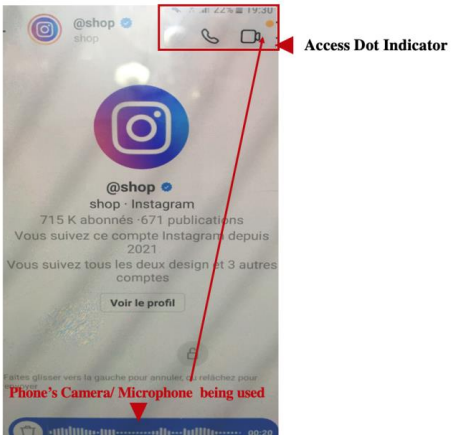

Figure 1. 1*: Access dots app detecting misuse of microphone*

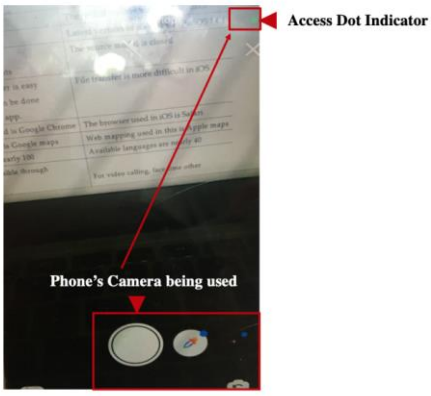

Figure 1. 2: *Access dots app detecting misuse camera*

1.6.3 The Hybrid approach

This approach combines static and dynamic analysis. A few researchers used this approach. By combining prior installing analysis with post installing analysis relying of behaviors monitoring of the app [12].

1.7 Conclusion

Mobile apps are part of our daily lives. 2021 will continue to be a big year for the mobile app industry. This is encouraging for anyone who has a mobile app or plans to build one. This chapter introduced the reader to some famous security related scandals occurred on mobile phones in the few past recent years. It also highlighted the research community efforts in investigating the permission related risks, raising awareness about them, and protecting against them. The short lesson we want to emphasize is that android apps may expose smart phones to risks and it can abuse and misuse user's granted permissions. Therefore, although it is challenging to investigate and analyze those apps, detect the threats, and ensure that our devices remain free from those risks, it is mandatory to do so.

PART II
CONTRIBUTION

CHAPTER 2

RISKS ANALYSES FOR ANDROID PERMISSION

2.1 Introduction

After highlighting the issues related to random permission grants upon applications requests. In this section, we follow the footsteps of researchers and carry out an investigation relating to 16 commonly used applications by our youngsters and elders. Although, we did not carry an extensive survey to obtain these applications, however, we relied on our direct observation to mobile phone users and also on the play store app ratings and rankings. The chosen applications belong to various categories starting from social media networks, to entertainments, communication, gaming and books. The investigation was carried for android operating system and its compatible applications. It followed these steps: first downloading the application packages, downloading investigation-needed tools, extracting permissions, filtering them then classifying them basing on our set of defined criteria's, we noted this phase as the investigation by static approach. After sorting out the initial risk hypothesis, we further analyzed the application using the dynamic approach which follows their behavior post installation, we focused on analyzing the network traffic when the phone is idle. We concentrated on this because the risks we are interested to study are the privacy violation, emission of targeted ads and the control of phone from afar and turning it into botnet, all of which rely on the internet to exchange data and/or commands to succeed and this will most likely occur when the phone is idle to avoid alerting the phone user. The investigation steps, details and results will be discussed in this chapter.

2.2 List of the chosen Applications

The chosen applications to investigate are the famous ones among smartphone users basing on play store rating and ranking and on our observations. The applications belong to social media category such as: Twitter, TikTok, SnapChat, Instagram and Facebook. The second category we cared for is communication applications such as: Whatsapp, TrueCaller and Viber. The video players category included applications like YouTube, YouTube Kids, Netflix. For the game category, we choose: "كلمات كراش", Subway Surfers, Candy Crush Saga, Magic Tiles 3 and Garena Free Fire World Series. Lastly, for the book's category GoodReads is analyzed. Details about these applications can be found in Table 2.1.

Table 2. 1: List of the chosen applications to investigate

App name	Brief description	Category	Number of Installs	Rating
Twitter [14]	Twitter allows share short posts called "tweets" including links, videos or photos.	**Social**	1B+	3.5
TikTok [15]	Video-sharing app that allows users to create and share 15-second videos on any topic.		1B+	4.4
SnapChat [16]	Messaging app that lets users exchange pictures and videos "snaps" that are meant to disappear after they are viewed.		1B+	4.3
Instagram [17]	Photo-sharing application and social network platform, acquired by Facebook since 2012.		1B+	3.8
Facebook [18]	Social network platform enabling users to connect share status, personal photos and other items.		5B+	2.3
WhatsApp [19]	Allows users to send text/voice messages, make free voice/video calls, share images and documents.	**Communication**	5B+	4.4
Viber [20]	Allows users to make free calls, send messages (texts, pictures and video).		5B +	4.3
TrueCaller [21]	It allows the caller-identification, call blocking, messaging, call recording, using the internet.		5B +	4.5
Netflix [22]	Subscription-based streaming service that allows watching TV shows and movies without commercials using internet.	**Video Players**	1B +	4.4
YouTube [23]	Free video sharing service, with the options of subscribing, watching, liking, sharing, commenting.		5B +	4.1
YouTube Kids [24]	YouTube service for kids offering safer content and allowing parents to control their kids viewed content.		100M+	4.3
Goodreads [25]	allows users to search books, explanations, quotes, and reviews	**Books**	10M +	3.7
كلمات كراش [26]	An Arabic word game puzzle.	**Games**	10M+	4.7
Subway Surfers [27]	an endless runner mobile game		1B +	4.4
Candy Crush Saga [28]	Free-to-play match-three puzzle video game.		1B+	4.6
Magic Tiles 3 [29]	Game app that simulates playing musical instruments where players have to tap the (piano-like) screen to match the played notes.		100M+	4.1
Garena Free Fire World Series [30]	Survival multiplayer shooter game.		500M +	4.2

2.3 Investigation and analysis of Applications

In this section, an investigation is performed about the level of the security of the chosen apps in terms of privacy, mobile botnet, and targeted advertisement.

2.3.1 Investigation Methodology- Static Approach

The static approach is applied for the android app classification, where the app is being analyzed without running it. First, the application is downloaded. Then, its package is decompressed into several files among which is the AndroidManifest.xml file. We used this file to investigate the permissions accessed by this application. The objective is to investigate

application's permission from the perspective of our privacy protection, and to ensure the safety of our data and devices when using this application. As we mentioned before, apps have access to sensitive and personal data stored for a long period of time in smartphone, which may be used to send us targeted advertising basing on our likings and location, they may also be leaked and traded putting us at the threat of theft, impersonation, blackmailing, and scapegoating. Apps also have the possibility and the capability to use our phone as a botnet , by sending or receiving infected SMS "Botnets are a network of hijacked devices used to hold various or to perform attacks" [31], it also have access to several phone resources: camera, audio, location, and Bluetooth that generates an impotent data: our location can be tracked, app can secretly take photos or recording videos. To classify thus permission, the android developer's classification is taken as a starting point and a base for our classification. We expect to find results that confirm our hypothesis or contradict them and obtain a result and classification of the application.

2.3.1.1 Classification based on the static technique

Figure 2.1 shows the process of the analysis using the static technique, which will be explained in details in the coming subsections.

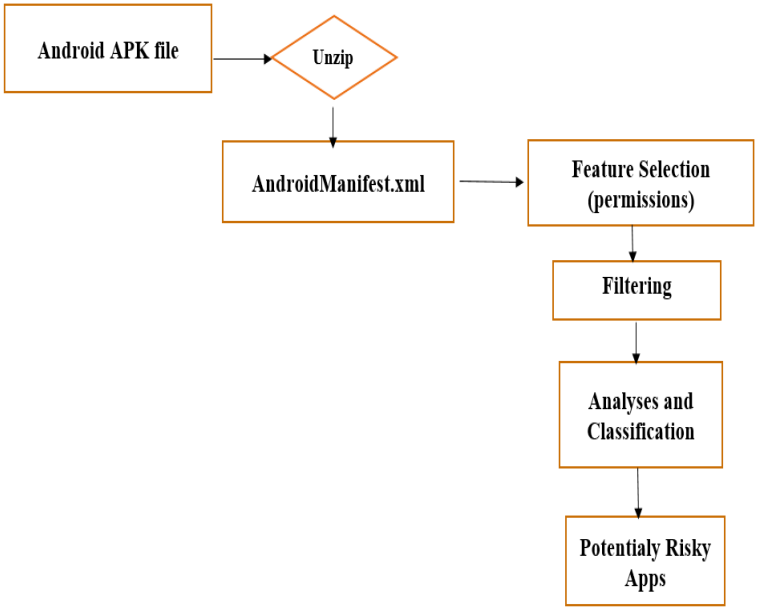

Figure 2. 1: Process of the static Investigation Approach

A) Feature extraction

AirDroid app is used to store the Apk files on the computer: Android Application Package(APK) gives a compressed (zipped) file of any Android application. Generally, an APK file contains .dex files, resources, assets, certificates, and Android manifest file [12]. We

use Apktool for decompiling the apk files. Once the apk files is decompiled, we extract the app permissions from the AndroidManifest.xml

App permissions on android protect access to the following:

- Restricted data, such as user's contact information.
- Restricted actions, such as recording audio.

Some permissions, known as install-time permissions (see figure 2.2), are automatically granted when your app is installed. Other permissions, known as runtime permissions, require your app to go a step further and request the permission at runtime.

Install-time permissions give your app limited access to restricted data, and they allow your app to perform restricted actions that minimally affect the system or other apps. Android includes several sub-types of install-time permissions, including normal permissions and signature permissions [32].

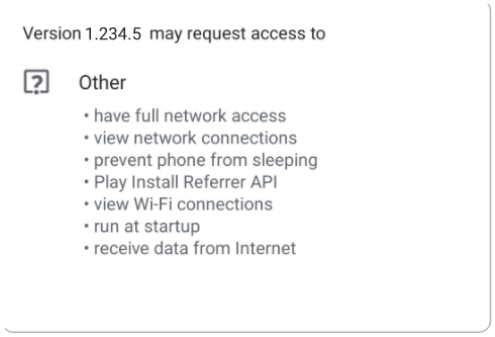

Figure 2. 2: Install-time Permissions *[32]*

Runtime permissions (Figure 2.3), also known as dangerous permissions, give your app additional access to restricted data.

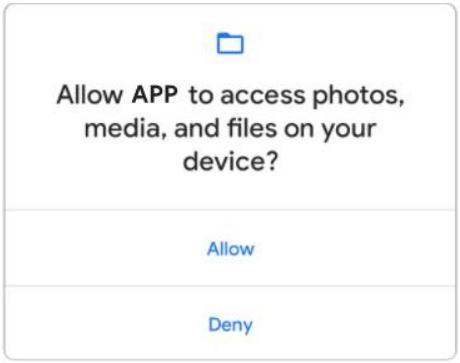

Figure 2. 3: Example of Run-Time Permission *[32]*

B) Permission Filtering

The level of protection of android permission is categorized into four groups: Normal, Dangerous, Signature, and Special/privileged (see Table 2.2). We focused on normal and

dangerous permissions that are not managed by the android Operating System. Grouping permissions by this way, enable us to decide which permission should be granted access or not, and classify the permission: safe, potentially risky and risky.

Table 2. 2: Android Protection levels categories *[32]*

Level	**Needs user approval**	**Description**	**Example**
Normal	No	Provides access to data or resources outside the app sandbox. Does not induce any risk to private data or other apps operations.	WiFi state, Internet, Bluetooth, etc
Signature	No	These permissions are granted at install time.	Battery stats, carrier services, etc
Dangerous	Yes	These permissions provide access to sensitive data or resources. They could access the user's stored data or operations by other apps. The user must explicitly authorize the usage of these permissions, only then the functionality that depends on these permissions work correctly.	Read contacts, camera, capture audio, etc
Special / Privileged	Yes	Similar to dangerous permissions, but the authorization of these permissions is managed by Android Operating system.	Write settings, system alert windows, etc

C) Analyses and classification

We analyze each application permission's that include but not limited to: sending SMS, receiving SMS, reading call logs, accessing the Internet, location and smartphone Files. Figure 2.4 shows the process of the permission classification. Just like we illustrated in the figure, our classification is based on, the android taxonomy in which permissions are categorized into normal and dangerous permissions

The permission is classified potentially risky if it is

- A normal permission used by the app and not declared in Google store.
- A normal permission used by the app and not match the apps' purpose.
- Normal or dangerous permission used by the app that can be used in suspicious purposes.

The permission is classified risky if it is

- Dangerous permission not matching the use of app.
- Dangerous permission not declared in Google store.

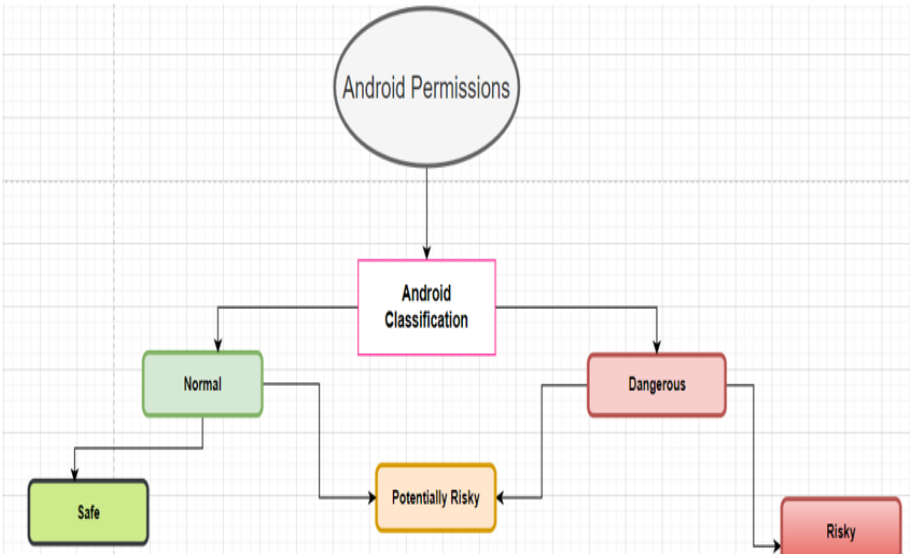

Figure 2. 4: Permission Classification Process

Before we classify android permissions, we need to understand the functionality of these permissions, table 2.3 shows the usage of extracted permissions.

Table 2. 3: Permissions usages and meanings *[32]*

No.	Permission	Use
1	**READ_CONTACTS**	Allows an application to read the user's contacts data.
2	**WRITE_CONTACTS**	Allows an application to write the user's contacts data.
3	**READ_EXTERNAL_STORAGE**	Allows an application to read from external storage
4	**WRITE_EXTERNAL_STORAGE**	Allows an application to write from external storage.
5	**ACCESS_MEDIA_LOCATION**	Allows an application to access any geographic locations persisted in the user's shared collection.
6	**ACCESS_FINE_LOCATION** **ACCESS_COARSE_LOCATION**	Allows an app to access approximate location
7	**RECORD_AUDIO**	Allows an application to record audio
8	**MODIFY_AUDIO_SETTINGS**	Allows an application to modify global audio settings.
9	**RECEIVE_SMS**	Allows an application to receive SMS messages
10	**READ_SMS**	Allows an application to read SMS messages.
11	**SEND_SMS**	Allows an application to send SMS messages.
12	**WRITE_SMS**	Allows an application to read SMS messages.
13	**USE_CREDENTIALS**	Use accounts on the device. Allows the app to request authentication tokens.
14	**CAMERA**	Required to be able to access the camera device
15	**GET_ACCOUNTS**	Allows access to the list of accounts in the Accounts Service.
16	**AUTHENTICATE_ACCOUNTS**	One of the strongest permissions any app can get access to. Using this permission, an app can retrieve or update an account password, retrieve an account authentication token and even create new accounts.
17	**MANAGE_ACCOUNTS**	add or remove accounts. Allows the app to perform operations like adding and removing accounts, and deleting their password.
18	**ACCOUNT_MANAGER**	Allows applications to call into AccountAuthenticators.
19	**READ_PROFILE**	Read your own contact card. Allows the app to read personal profile information stored on your device, such as your name and contact information. This means the app can identify you and may send your profile information to others.

20	**READ_PHONE_STATE**	Allows read only access to phone state, including the current cellular network information, the status of any ongoing calls, and a list of any PhoneAccounts registered on the device.
21	**READ_PHONE_NUMBERS**	Allows read access to the device's phone number(s). This is a subset of the capabilities granted by READ_PHONE_STATE
22	**CALL_PHONE**	Allows an application to initiate a phone call without going through the Dialer user interface for the user to confirm the call.
23	**USE_BIOMETRIC**	Allows an app to use device supported biometric modalities.
24	**USE_FINGERPRINT**	Allows an app to use fingerprint hardware
25	**MANAGE_DOCUMENTS**	Allows an application to manage access to documents, usually as part of a document picker.
26	**BLUETOOTH**	Allows applications to connect to paired Bluetooth devices
27	**BLUETOOTH_ADMIN**	Allows applications to discover and pair Bluetooth devices.
28	**CHANGE_NETWORK_STATE**	Allows applications to change network connectivity state.
29	**CHANGE_WIFI_STATE**	Allows applications to change Wi-Fi connectivity state.
30	**RECEIVE_ADM_MESSAGE**	Amazon device messaging.
31	**READ_GSERVICES**	Allows an application to modify the Google service map
32	**GET_PACKAGE_SIZE**	Allows an application to find out the space used by any package
33	**WRITE_SYNC_SETTINGS**	Allows applications to write the sync settings.
34	**ACCESS_WIFI_STATE**	Allows applications to access information about Wi-Fi networks
35	**SYSTEM_ALERT_WINDOW**	Allows an app to create windows using the type WindowManager.LayoutParams. TYPE_APPLICATION_OVERLAY, shown on top of all other apps. Few apps should use this permission; these windows are intended for system-level interaction with the user
36	**RECEIVE_BOOT_COMPLETED**	Allows an application to receive the Intent. ACTION_BOOT_COMPLETED that is broadcast after the system finishes booting.
37	**NFC**	NFC Near Field Communication technology allows users to make secure transactions, exchange digital content, and connect electronic devices with a touch. Allows applications to perform I/O operations over NFC.
38	**INTERNET**	Allows applications to open network sockets
39	**ACCESS_NETWORK_STATE**	Allows applications to access information about networks.
40	**FOREGROUND_SERVICE**	Allows a regular application to use Service.startForeground (an advanced Android concept which allows you to display notifications to your users when running long lived background tasks. The notification acts like any other notification, however it cannot be removed by the user and lives for the duration of the service)

To analyze the 17 selected apps, we unzipped their package file using APKtool to extract permissions from the Manifest file. Unfortunately, we had to eliminate the Facebook app from the selected app list, because we faced problem decompiling its 320.0. 0.34 apk version with Apktool. Therefore, we continued our investigation on the rest 16 apps organized into: 4 social media apps, 3 communication apps, 3 entertainment apps, 1 book app and 5 game apps. We filtered the obtained permissions by eliminating those needed by the system and leaving those related to the apps. The classification of permission relies on comparing permissions declared by the app developer in Google Store and the permissions found in the Manifest file. Our initial

classification based on the android classification then it was enhanced using the above-mentioned criteria related to the potentially risky and risky permissions.

D) Result and discussion

After analyzing the applications basing in criteria in section C, we calculated in percentage the ratio of safe, potentially risky and risky permissions found in each app, using the equations (1), (2) and (3) respectively.

For each application:

- Rating of safe permissions: $\frac{number\ of\ safe\ permissions}{total\ number\ of\ permissions}$*100 ……..*(1)*
- Rating of potential risky permissions: $\frac{number\ of\ potentielly\ risky\ permissions}{total\ number\ of\ permissions}$*100…..*(2)*
- Rating of risky permissions: $\frac{number\ of\ \ risky\ permissions}{total\ number\ of\ permissions}$*100 ……..*(3)*

Table 3.4 recaps in numbers the total permission for each application, the number of safe, potentially risky and risky permissions as well as their ratios. We classified the applications from the riskiest to the safest taking on consideration descending sorting of risky apps ratio, followed by potentially risky apps ratios. In the table, colors are degraded from dangerous applications in red, to safe app in green.

Initial analysis

From the feature extraction of permissions, with the help of the static approach and via filtration and classification processes, we obtain table 2.4 that summarize the classification of permissions for each of the applications into, safe, potentially risky and risky.

As a result, we deduce that the following applications were classified as risky because they contained risky permissions: Instagram, Good reads, YouTube, WhatsApp, Viber, Truecaller, Wood block puzzle, Snapchat, Garena Free fire, Twitter, TikTok, Netflix, and YouTube kids.

As for, Subway surfer, magic tales 3, كلمات كراش and Candy Crush Saga were classified into potentially risky applications because they contained potentially risky permission. These claims need further investigation to be confirmed, which will be done using the network traffic monitoring.

Table 2. 4: Selected Apps' Permission Analysis in numbers

Apps	Total number of permissions	Number of			Ratio of		
		Safe Permissions	Potentially Risky Permission	Risky Permissions	Safe	Potentially Risky	Risky
Instagram	29	6	14	9	21 %	49%	32%
Goodreads	20	1	14	5	5%	70%	25 %
Whatsapp	46	11	20	11	24%	43%	24%
Youtube	26	4	18	6	15%	69%	23%
Garena Freefire	13	3	9	3	23%	69%	23%
Viber	39	8	23	8	21%	60%	21%
Snapchat	28	8	14	6	29%	50%	21%
True Caller	54	6	37	11	11%	69%	20 %
Wood Block Puzzle	5	1	3	1	20%	60%	20%
Twitter	32	8	20	4	25%	63%	13%
Tiktok	23	8	12	3	35%	52%	13%
Netflix	10	2	7	1	20%	70%	10 %
Youtube Kids	13	2	10	1	15%	77%	8%
كلمات كراش	4	0	4	0	0%	100%	0%
Candy Crush Saga	5	1	4	0	20%	80%	0%
Subway Surfers	5	1	4	0	20%	80%	0%
Magic Tiles 3	9	4	5	0	45 %	56%	0%

As we mentioned earlier, we are interested to study the risks related to the privacy violation, exploiting and misusing the phone as a botnet and to taking advantage of the users' data to send him/her targeted ads. Therefore, we wanted to further analyze the type of risk found in each application, and whether or not these risky permissions are of install-time or runtime type. For this purpose, we classify in table 3.5 the permissions basing on their type and the type of risk they engender. We relied initially for the classification of botnet related permission on the work of [12]. To facilitate the classification of these permissions for each application, we designated a short codification preceding each permission in table 2.5. Example: P1 is the codification of READ_CONTACTS, it is written in the table as P1 READ_CONTACTS. This codification was used in table 2.6 to represent the permissions related to each risk, then we calculated the ratio of risk related to privacy violation, botnet exploitation and targeted advertisement. The ratios were calculated from the total number of permissions for each application using equations (4-6).

For each application:

- Privacy violation ratio : $\frac{number\ of\ privacy-related\ permissions}{total\ number\ of\ permissions}$*100 ……..*(4)*
- Botnet ratio : $\frac{number\ of\ botnet-related\ permissions}{total\ number\ of\ permissions}$*100 ……..*(5)*
- Targeted ADS ratio : $\frac{number\ of\ targetted-ads-related\ permissions}{total\ number\ of\ permissions}$*100 ……..*(6)*

As a result, from the analysis made in table 3.6, the majority of applications used permissions that are related to privacy violation, botnet exploitation or targeted ads. These claims need further investigation to be confirmed, which will be done using the network traffic monitoring in the next section.

E) Investigate using Traffic monitoring:

In the previous section, we used the static approach to investigate the permission abuse, before installing the applications. Accordingly, we resumed a set of claims related to the permissions risks by violation type which are privacy, botnet and targeted ads. All these risks rely on internet usage to exchange data and/or commands. Leaked data is sent via internet to the application developer, targeted ads are sent from advertisers via internet to users basing on their likings and phones may be controlled by attackers from afar also via internet when used as a botnet. Therefore, monitoring network traffic is fundamental to track the applications' use of internet, which would help us to further prove our permissions risk claims, especially when the application is idle, and exchanging data over internet. This approach is known as the dynamic approach, which can be conducted after the installation of the applications. To do the monitoring, we created a laboratory environment using a real Samsung phone device equipped with monitoring tools, to test the 17 applications (illustrated in figure 2.5). The device was connected to the internet and was idle during the whole investigation period fixed to 36 hours. The characteristics of the device used in the laboratory environment can be found in table 2.7.

Table 2. 5: Permission classification per risk type

Permissions related to Privacy		**Permissions related to Botnet**		**Permissions related to Targeted ADS**	
Install-time permissions	***Runtime permissions***	***Install-time permissions***	***Runtime permissions***	***Install-time permissions***	***Runtime permissions***
P32 READ_GSERVICES	P1 READ_CONTACTS	P29 CHANGE_NETWORK_STATE	P10 RECEIVE_SMS	P31 RECEIVE_ADM_MESSAGE	P15 CAMERA
P35 ACCESS_WIFI_STATE	P3 READ_EXTERNAL_STORAGE	P30 CHANGE_WIFI_STATE	P12 SEND_SMS	P41 FOREGROUND_SERVICE	P21 READ_PHONE_STATE
P43 READ_SETTINGS	P5 ACCESS_MEDIA_LOCATION	P36 SYSTEM_ALERT_WINDOW	P13 WRITE_SMS		P3 READ_EXTERNAL_STORAGE
P60 READ_SYNC_SETTINGS	P6 ACCESS_FINE_LOCATION	P37 RECEIVE_BOOT_COMPLETED	P14 USE_CREDENTIALS		P20 READ_PROFILE
P61 READ CALENDER	P7 ACCESS_COARSE_LOCATION	P38 NFC	P18 MANAGE_ACCOUNTS		P24 USE_BIOMETRIC
P61 WRITE_CALENDER	P8 RECORD_AUDIO	P34 WRITE_SYNC_SETTINGS	P19 ACCOUNT_MANAGER		P8 RECORD_AUDIO
P62 READ_SYNC_STATE	P11 READ_SMS	P39 INTERNET	P23 CALL_PHONE		P0 COLLECT_METRICS
P66 READ_DATA	P15 CAMERA	P40 ACCESS_NETWORK_STATE	P24 USE_BIOMETRIC		
P69 ADD_VOICE MAIL	P16 GET_ACCOUNTS	P35 ACCESS_WIFI_STATE	P25 USE_FINGERPRINT		
P70 WRITE_VOICE MAIL	P17 AUTHENTICATE_ACCOUNTS	P44 WRITE_SETTINGS	P4 WRITE_EXTERNAL_STORAGE		

P71 READ_VOICE MAIL	P18 MANAGE_ACCOUNTS	P67 READ_SYNC_SETTINGS	P2 WRITE_CONTACTS		
	P19 ACCOUNT_MANAGER		P42-READ_CALL_LOG		
	P20 READ_PROFILE	P65 CHANGE_WIFI_MULTICAST_STATE	P48 CALL_PRIVILEGED		
	P21 READ_PHONE_STATE	P72 MANAGE_DEVICE_AND_USER_DATA	P49 MODIFY_PHONE_STATE		
	P22 READ_PHONE_NUMBERS		P50 RECEIVE_MMS		
	P24 USE_BIOMETRIC		P51 WRITE_VOICEMAIL		
	P25 USE_FINGERPRINT		P52 MANAGE_DEVICE_AND_USER_DATA		
	P26 MANAGE_DOCUMENTS				
	P42 READ_CALL_LOG		-		
	P45 ACCESS_MEDIA_LOCATION				
	P46 ANSWER_PHONE_CALLS				
	P47 PROCESS_OUTGOING_CALLS		-		
	P52 MANAGE_DEVICE_AND_USER_DATA				

Table 2. 6: Classification of app permissions basing on their engendred risk type

	Privacy		Botnet		Targeted ADS	
	Permissions	**Ratio**	**Permissions**	**Ratio**	**Permissions**	**Ratio**
Wood Block Puzzle	/	**0%**	P39; P35; P4; P40	**80%**	/	**0%**
Candy Crush Saga	P35	**20 %**	P40; P35; P37	**60%**	P41	**20%**
Subway Surfers	P35	**20%**	P40; P35; P37	**60%**	P41	**20%**
Netflix	P8; P35	**20%**	P40; P35; P4; P65; P39	**50%**	P41; P8	**20 %**
Garena Free Fire World Series	P35; P8	**15 %**	P37; P40; P30; P29; P44; P39	**46%**	P41; P8	**15 %**
Magic Tiles 3	P43	**11%**	P40; P35; P37; P39	**45%**	/	**0%**
كلمات كراش	/	**0%**	P35; P39	**50%**	P41; P21	**50%**
YouTube	P35; P26; P16; P18; P32; P15; P25; P24; P1; P6; P7; P8; P21; P43	**54%**	P40; P35; P 4; P37; P18; P14; P38; P25; P24; P36; P39	**42%**	P15; P41; P8; P24	**15%**
WhatsApp	P21; P22; P24; P25; P6; P7; P35; P17; P16; P8; P 1; P20; P3; P32 P46; P42	**35%**	P10; P24; P25; P35; P29; P30; P18; P38; P12; P14; P34; P44; P23; P42; P67; P39; P60; P63	**39%**	P15; P41; P8.P21.P24; p20; P3	**15%**
True caller	P42; P15; P21; P22; P46; P47; P1; P16; P20; P6; P35; P7; P6; P17; P8; P11; P8; P43; P70	**35.19 %**	P42 P23; P 48; P49; P2; P35; P29; P4; P18; P14; P10; P12; P13; P50; P51; P37; P36; P44; P39; P69; P70	**39%**	P15; P41; P8.P21.p20; P3	**11,1%**
YouTube Kids	P35; P32; P16	**23%**	P40; P35; P37; P4; P39	**39%**	P41; P 8	**15%**
Viber	P35; P1; P21; P42; P8; P29; P17; P16; P18; P 7; P15; P32; P6; P15; P3; P60; P63	**44%**	P35; P40; P 2; P42; P 4; P29; P44; P18; P14; P34; P36; P37; P39; P67	**36%**	P15; P 41; P8.P21.P3	**13%**
Snapchat	P15; P8; P6; P21; P42; P1; P3; P16; P20; P35; P42; P43	**13%**	P4; P42; P40; P37; P30; P35; P42; P44; P39	**32%**	P15; P 41; P8; P21; P20; P3	**21%**
Goodreads	P1; P 21 P15; P3; P7; P6	**30%**	P40; P 14; P2; P37; P4; P39; P72	**35%**	P15; P41; P0; P21; P3	**25%**
Twitter	P20; P1; P32; P21; P15; P8; P16; P6; P7; P17; P18; P35; P19	**44%**	P4; P40; P36; P18; P14; P34; P35; P37; P19; P39	**31%**	P15; P 41; P 8; P21; P20	**16%**
Tiktok	P3; P35; P8; P1; P17; P62; P61	**30%**	P39; P40; P4; P35; P37; P34; P31; P67; P39	**39%**	P15; P41; P8; P31; P3	**22%**
Instagram	P16; P1; P20; P 6; P8; P45; P21; P22; P43; P24; P25	**41%**	P40; P14; P4; P23 P30; P24; P25; P39	**28%**	P41; P15; P8; P21; P31; P24; P20	**24%**

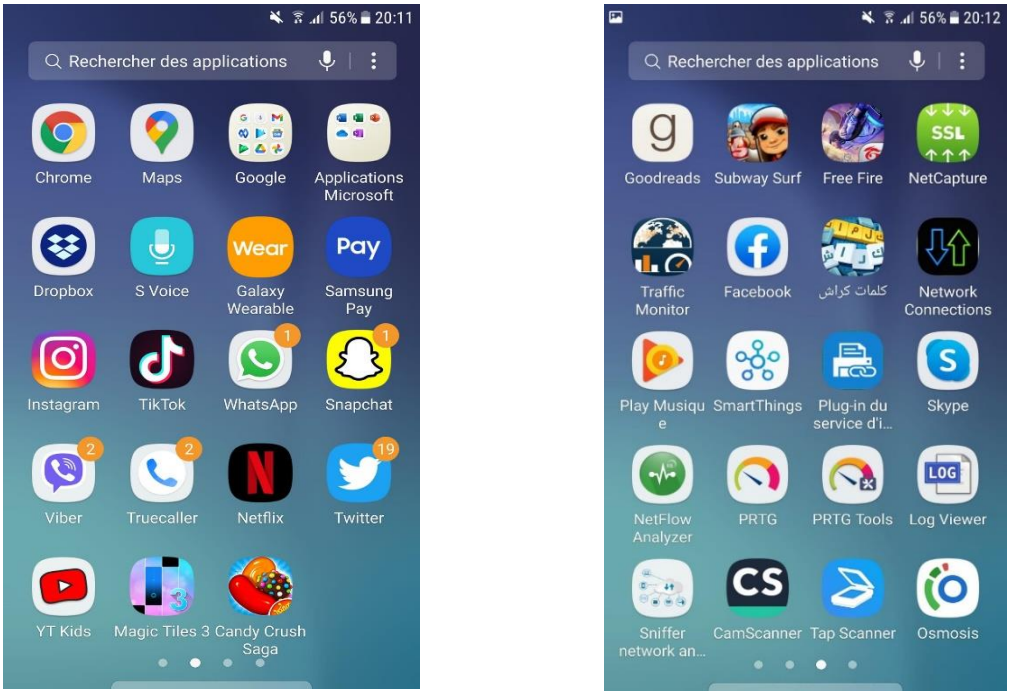

Figure 2. 5: laboratory environment with the 17 selected apps installed

Table 2. 7: Specification and characteristics of the used device

Brand	Samsung
Model	Galaxy S6
Battery capacity	(mAh)2550
Processor	octa-core (4x2.1GHz + 4x1.5GHz)
RAM	3GB
Internal storage	32GB
Camera	Rear camera and Front camera
Connectivity	Wi-Fi /GSM 3G 4G/LTE
Software Operating system	Android 7.0
Sensors	Fingerprint sensor

- **Apps used in monitoring**

Various monitoring exist such as but not limited to sniffer network analyzer, NetCapture, data usage monitor. We used NetCapture app because it displays connection from and to the device by application. Unfortunately, the app requires fees payment for continuous long-term monitoring. As such, we were obliged to divide the monitoring period into small chunks of 1h30 each. Furthermore, the application did not provide detailed information about the type of exchange data. We tried also to use Sniffer Network Analyzer to get more details about the

type of data per application, however, it did not allow the extraction of data related to capture traffic in a readable manner (encrypted traffic captures). As a result, we relied on the results obtained from NetCapture tool. Figure 2.6 illustrates a screenshot of the captured network traffic by NetCapture tool.

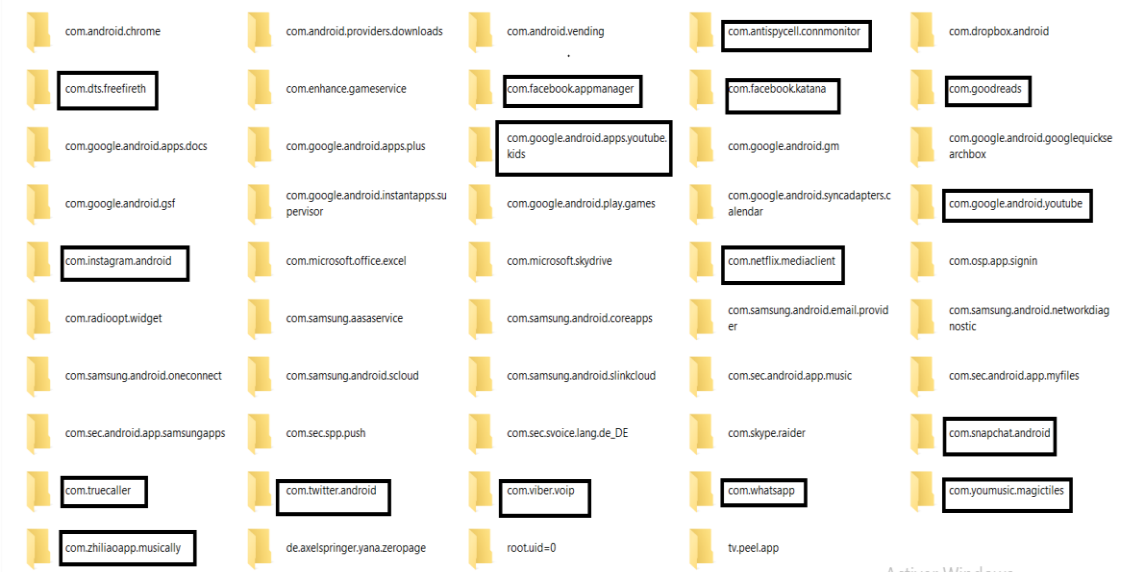

Figure 2. 6: Screenshot of the captured network traffic by NetCapture tool

- **Result and discussion**

After the process of capturing network traffic and inspecting it, the obtained results are presented in figures 2.7, 2.8. We found also that the behavior of some apps is suspicious: even that we do not use the app or the device, a network usage is detected. In this network traffic capture, we focused on the number of internet connections made by the app, in the whole period of the capture and not on the duration the app was connected. The fact that the application made an internet connection when we did not use this application, and our device is idle, means that this app is a potentially risky application.

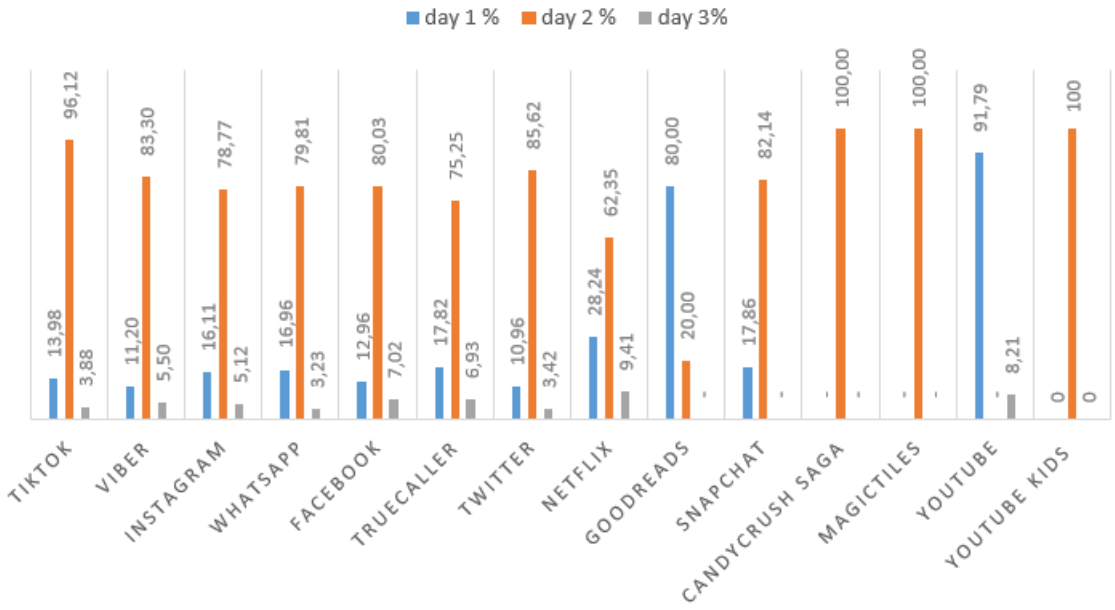

Figure 2. 7: Ratio of internet connections made by each application during the 3 days of observation

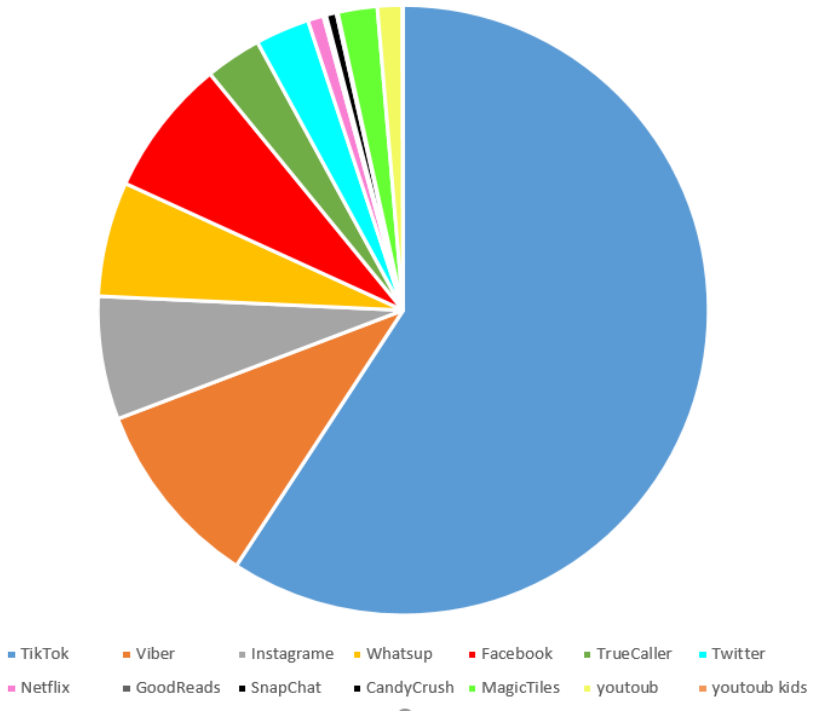

Figure 3. 8: Global ratio of internet connections made by apps during the 3 days

- **Post monitoring analysis**

Upon monitoring and analyzing the network traffic exchanged by the applications, we found that the applications: **TikTok, Viber, Instagram, WhatsApp, Truecaller, Twitter, Netflix, Good reads, Snapchat, YouTube, YouTube kids and Garena Free Fire** were accessing internet even when they were not used (inactive) and the phone was idle. This confirm the hypothesis of been classified **potentially risky/risky**.

For the following applications:

- **Wood Block puzzle** did not access internet during the observation period and was classified risky by a ratio of 20% in the first classification, because of dangerous permission “Write external storage” a run time permission requiring the user approval, which does not contradict with the app, intended purpose. Thus, we reclassify this app as safe application instead of risky.
- **Subway Surfer and كلمات كراش**: Those applications were classified potentially risky tending to be safe because of the Forground_service (install permission) can be used in targeted ADS. After the monitoring these apps did not access internet during the observation period, which makes them safe.
- **Candy Crush Saga and Magic Tales 3:** these two applications were classified potentially risky, because of their permissions that are related to botnet exploitation. After the monitoring, the app accessed internet even when they were inactive which makes them Risky, potentially risky apps. Table 2.8 summarizes our final classification after applying both the static and dynamic approaches.

Table 2. 8: Final risk-based classification for applications

App Name	Classification prior monitoring	Monitoring Results	Post monitoring classification
Tiktok	Risky/ Potentially risky	Connected to Internet	Risky
Viber	Risky/ Potentially risky	Connected to Internet	Risky
Instagram	Risky/ Potentially risky	Connected to Internet	Risky
Whats App	Risky/ Potentially risky	Connected to Internet	Risky
True caller	Risky/ Potentially risky	Connected to Internet	Risky
Twitter	Risky/ Potentially risky	Connected to Internet	Risky
Good Reads	Risky/ Potentially risky	Connected to Internet	Risky
Snap Chat	Risky/ Potentially risky	Connected to Internet	Risky
Candy Crush Saga	Potentially Risky/Safe	Connected to Internet	Risky
Magic Tiles 3	Potentially Risky/Safe	Connected to Internet	Risky
Youtube	Risky/ Potentially risky	Connected to Internet	Risky
Youtube Kids	Risky/ Potentially risky	Connected to Internet	Risky
Subway Surfers	Potentially Risky/Safe	Connected to Internet	Risky
Wood Block Puzzle	Potentially Risky/Safe	No internet connection	Safe
كلمات كراش	Potentially Risky/Safe	No internet connection	Safe
Garena Free Fire	Risky/ Potentially risky	Connected to Internet	Risky
Facebook	No classification for this App	Connected to Internet	Risky

Noting that these results aim to enlighten the users about the potential risks related to these applications and to emphasize the importance of not leaving the phone connected to the internet while idle nor to enable sensitive accesses to resources unless needed. These results are not

final, extensive analysis to identify the type of data exchanged by these data over the internet and long monitoring periods will give more accurate, irrefutable and definitive results.

2.4 Conclusion

In this work, we used the static approach to extract safe, potentially risky and risky permissions; this result is achieved by using the feature selection based on permissions, and the traffic monitoring. As a result of this classification, 21 permissions that are related to mobile privacy, 27 to botnet and 9 permissions related to targeted advertisement has been extracted.

As an example, many attackers have used GPS to exploit smartphones. Through GPS, they can know the victim information's such as geolocation information's. They can remotely monitor the user's smartphone through malware when the location permission is granted. Attackers also use the call log to get valuable information on contact list and call history for their own benefit. Smartphones are targets for cybercriminals to gain information of the users. Malware nowadays have the abilities to steal information stored in the smartphone, with the help of the permissions granted to the mobile application.

Many concerns about targeted advertisement also have been discussed in mobile security. Online advertising is currently a rich source of revenue for many internet giants and attackers may take advantage of these ads when a publisher is under attack from another source. This is also risky for children who uses smartphone if the app installed have access to the camera it may take photos or videos and use a facial recognition system to identify the user and send inappropriate ads for this category of users. At last, this study has presented a classification to differentiate a risky from benign (safe) applications, and apps that can potentially be a botnet, a privacy breach, or can send targeted ads.

CHAPTER 03
PERMISSION CHECKER APPLICATION

3.1. Introduction

The previous chapters emphasized the importance of mobile phone permission analysis and control. The abuse of permissions may allow the application developers to violate the user's privacy and exploit his/her phone endangering his/her security. The outcome of such actions may have moral impact that varies from blackmailing by extorted phone data, to sending targeted data. Or, it may have legal impact such as using the phone to send threats or as a botnet to take part in cyber-criminal acts such as hacking. To prevent the misuse of permissions we followed in the previous chapter the footsteps of other researchers in analyzing the permissions given to the applications and their correspondence with the needs of these apps' correct functionality. The main aims were to further prove the insecurity claims related to the abuse of permissions, to spread awareness about this issue and to check the security of some well-known applications used by the majority of smartphone users on daily basis. The approach we followed was static, semi-manual and can be replicated only by experts. This means that a regular user unaware of the steps cannot easily check whether or not the applications installed on his/her phone granted with a set of permissions represent a threat on his/her security and privacy. Therefore, a fully automatic permission related risk detection mechanism is needed. In this chapter, we implemented an android application that would help the normal users without security background know the potential risks related to permissions grants to the applications installed on his/her phone. We relied on automatizing the risk classification from the previous chapter. The application implementation process is composed of three steps. The first is related to selecting and learning the developments tools for an android application. The second is implementing the application that detects risk related to permissions misuse basing on what we learnt in the previous chapter. The third and last steps, is testing the functionality of our application and measuring its resources consumption before delivering it to end users.

3.2. Development tools

To implement our android mobile application names "***Permission Checker***". We first installed few tools and software needed for the successful implementations. We developed the application on a Windows 10 Laptop which we supplemented with the following tools to be able to develop on it the application that is intended to run on Android platforms. The tools are the Integrated Development Environment (IDE) Android Studio, Java runtime environment, emulators and profiler.

The development of Android Application requires the knowledge of JAVA or Koltin programming languages and XML markup language.

To test the mobile application two methods exists. The first is by running it on an android device such as a smartphone or a tablet. The developer mode options need to be enabled on the testing device as well as the USB debugging option. The second method is by using an emulator on PC. This method is safer especially when still in developing phase and it reduces the risk of damaging the mobile devices, it also enables the developer test the compatibility and the functionality of the developed application on various emulated devices and operating systems versions.

3.3. The Android Studio

Before discussing the implementation of our application, we first introduce the android studio and explain the main components of an android project. A project is composed of manifest xml file, gradle, xml layout files, Graphical User Interface (GUI) and views known as activities, the functioning codes and the resources folder containing images, icons, etcs. In what follows, we detail each of these components [33]:

- **Android Manifest**

The file AndroidManifest.xml, contains the necessary information to a successful run of the application code, the most important ones are:

  - **Activities**: the enumeration of the application screens
  - **Permissions**: to ensure the Security, it is required to grant specific permission to the application when performing certain tasks
  - **API level:** minimum API level of the targeted device
  - **Java package specification:** the naming of the package and its path containing Java classes needed, that represent a unique identifier
  - **Linked libraries:** the necessary libraries that needs to be linked for the application to work

- **Gradle**

Known as **build.gradle** is an automated JVM (Java Virtuel Machine) that takes the entire source file needed (java files, xml files) to create a compressed APK file, without the need of script to link these files together.

- **Layout XML files**

XML layout files define the graphic design of the application. With XML, a programmer can define how the layout of the activity appears. Each activity has a certain .xml file containing various layout components. These components are later linked from within the Java code of the activity.

- **Activities**

Every application has an Activity as its main element, where all **buttons, textviews, imageviews** and other components are located. The activities represent the screens (views or GUIs) a user sees when interacting with an Android application. Noting that these activities are needed to be specified in the AndroidManifest.xml file otherwise the application would crash. It is necessary to understand the activity lifecycle illustrated in Figure 3.1 to develop an android application. Each activity has a certain method that is called in a particular chronological order and on specific user-actions [34].

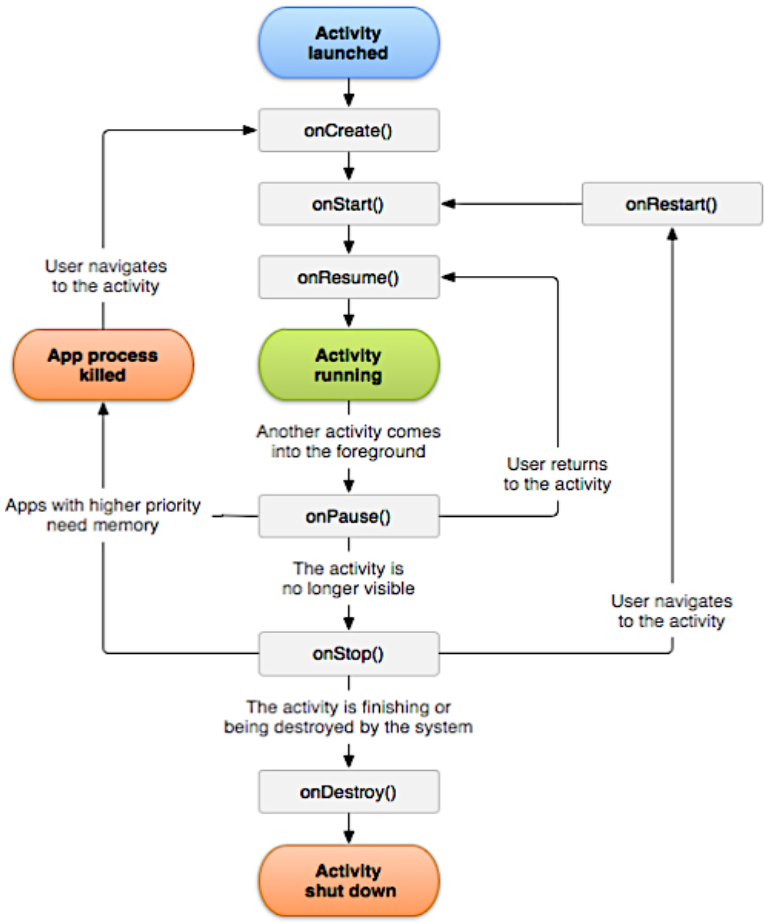

Figure 3. 1: Activity Life Cycle *[34]*

At the start, when an activity is created, the method onCreate() is called. This is where the developer defines the activity components such as buttons or ImageViews. When the activity is visible to the user, means that the onStart() method is invoked. Once the activity is started it invokes the onResume() method this is when the user can interact with the activity, the app stays in this state until another action happens. The method onPause()is invoked, when another activity interrupts the current activity. When this last finishes its execution, the paused activity resumes its execution by calling the onResume() method. In case the activity is not visible to the user, it is in the stop state and the onStop() method is called by the system, from this state the activity can go back to its running activity by calling the onResume() function or it is dismissed if the onDestroy() or finish() functions are called. Noting that the latter (finish()) can also be called when the system is destroying the activity because of configuration changes.

- **Resources**

Every android studio project contains a resource folder, it enables us organize the project by separating files like images and icons from the Java source code to have a clear project structure. Instead of referencing the files directly, relative path is used in the source code and this make changing value easier, without the need to look for the values manually in the source code.

3.4. Our Permission Checker App

In this part, we follow the same mobile app building process introduced in Chapter 01 to describe the steps of developing our application.

3.4.1. Step 1: idea of the app

The main idea behind building this app is to automate the process of permission checking for regular users. The application enables a non-security expert to check whether or not the applications installed on his/her phone and used by him/her on daily basis represent any danger on his/her security and privacy. It allows a user with zero-security knowledge to be aware of the consequences behind granting permission accesses to sensitive phone resources. The app focuses on detecting dangerous permissions such as: permissions allowing access to camera, contacts, location and storage. The application checks the permissions of the installed application which means a post installation detection. The followed detection approach is static approach because the permission analysis does not require that the application is running.

3.4.2. Step2: identification of targeted users

This application is designed for all android mobile phone users, regardless of their age, gender, ethnicity or religion. It does not require any particular knowledge or level of expertise of security concepts.

3.4.3. Step 3: the chosen approach

The approach used in the development of this app is the native approach which allows to develop application to work on android phones. Native apps are built using SDK's platforms and development tools.

To implement our android application, we starting by designing it (conception phase) by precising its main offered functionalities which are illustrated in the use case diagram in figure 3.2. Our application provides the user with three menus: 1) choose details, 2) open app and 3) app info. The first displays the risky permissions of the selected application, the second opens the selected application and the third displays the details of the application regarding its resource usage and allows the enabling/disabling of access to permissions.

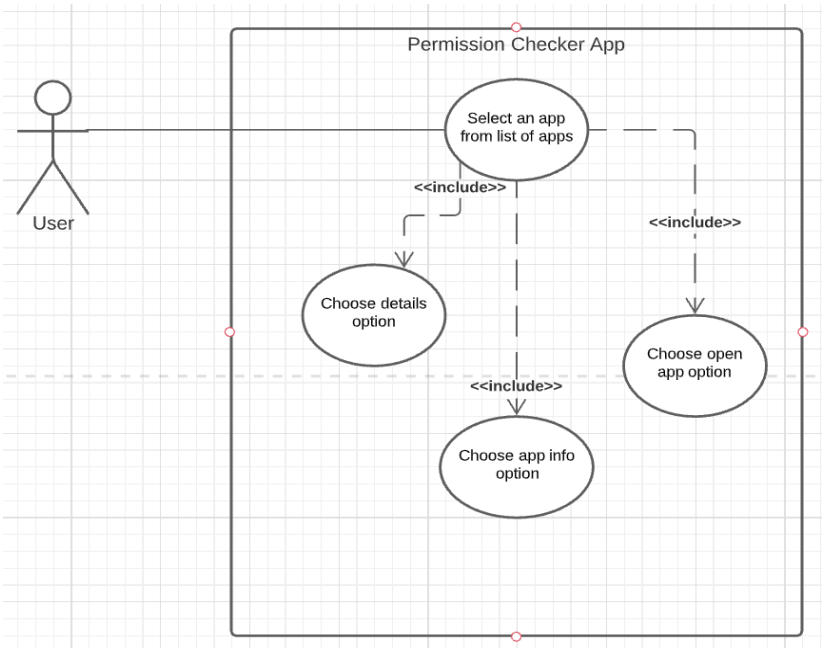

Figure 3. 2: Permission Checker app use case diagram

Our application has two main activities, the first is the main view of our application aka the launcher activity, the one that is displayed upon clicking on the checker application, it displays all the user-installed applications on the subject device. Upon selecting an application to analyze, the second main activity is launched, it is responsible for extracting the app permissions and investigating whether or not they are risky. This activity displays as an output

of the risky accesses to resources in natural language to facilitate alerting the non-expert users. Both of the activities are depicted in the class diagram of figure

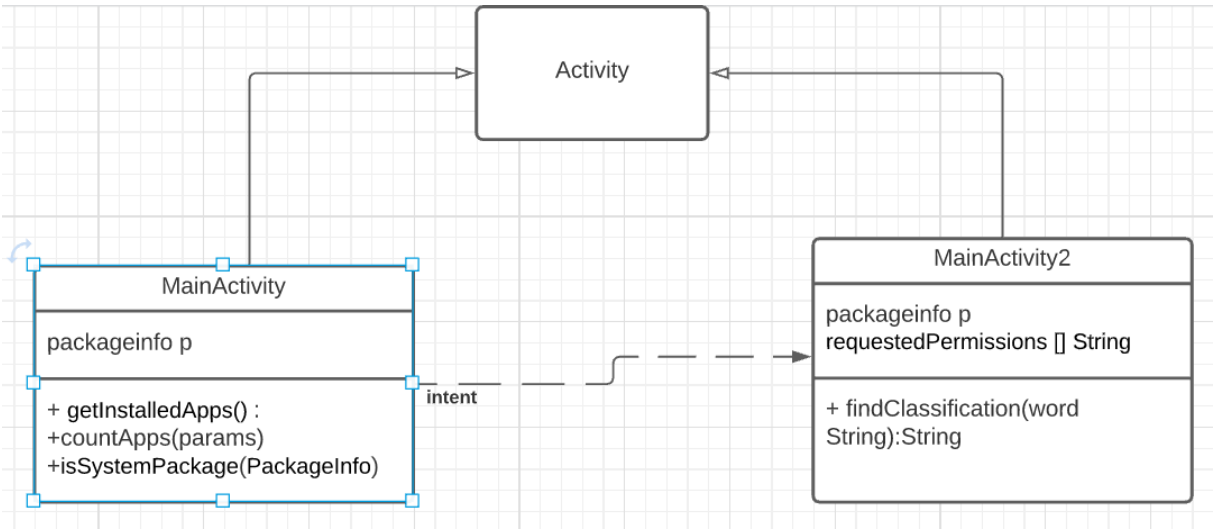

Figure 3. 3: Permission Checker Simplified Class Diagram

3.4.4. Step 4: identify the interface and implementation

In this section, we describe our mobile application named "***Permission Checker***" illustrated by its icon in Figure 3.4. We continue in the following explaining the multiple activities included in the Android Studio project of this application.

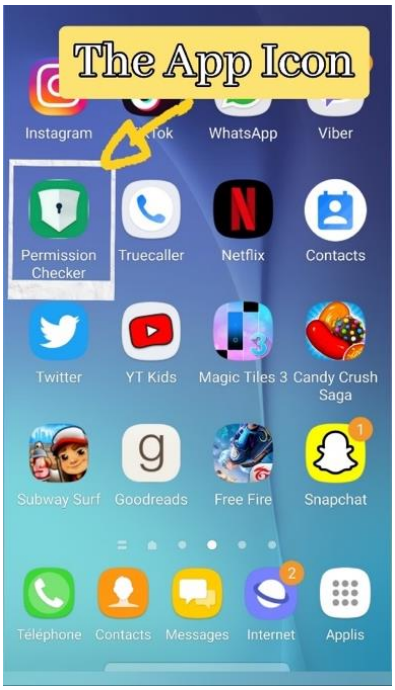

Figure 3. 4: Our "Permission Checker" icon upon installing it on a smartphone

- **The launcher activity**

The launcher activity of this application is called the main activity. Once clicked, our application displays the list of installed application on device. The displayed list illustrates each application's icon, its short name and its package name. On top of the list the total number installed applications is written. Figure 3.5 illustrates our permission checker main view.

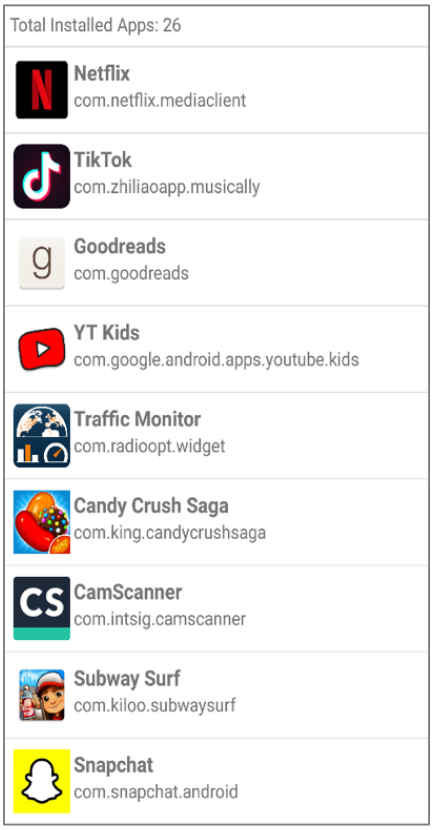

Figure 3. 5: Permission Checker Main View

When selecting an application and clicking on it. A new window is prompted offering a menu composed of three items (see Figure 3.6 for illustration)

- *App details:* Lists the application permission classification.
- *Open the app:* Starts running the selected application.
- *App info:* Displays the application resource consumptions.

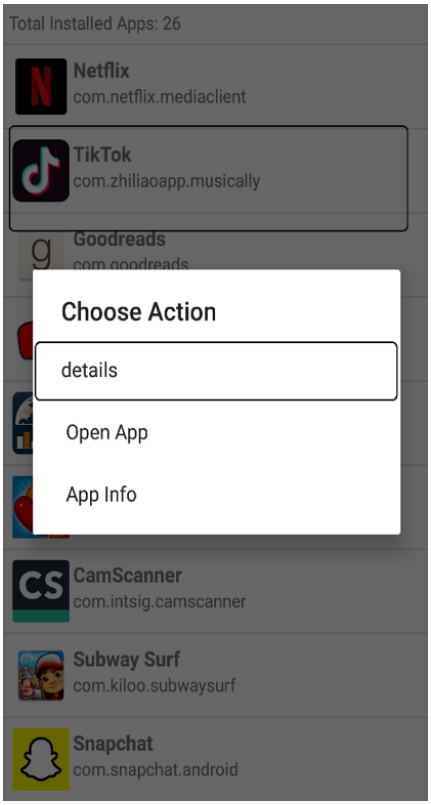

Figure 3. 6: Item listener options in our application

Noting that once one of the three options is selected, the activity is changed from the main activity to the chosen activity.

- **The second activity**

This activity is executed if the ***details*** option is chosen from the item listener from the main activity. In this activity, the potential risky accesses to phone resources are listed for the analyzed application. These potential risky accesses are mainly accessing to these sensitive resources: SD CARD, CAMERA, AUDIO, and CONTACTS. We concentrated on analyzing permissions related to these resources due to its direct impact and risk on the phone user's privacy. If there is a permission that the installed app uses, that is related to one of the 4 categories, an alert is illustrated to the users precising the potential risky access to notify him/her of the possibility of his/her phone resources and details being misused.

Figure 3.7 shows an example of one of the applications we analyzed in chapter 3 and found it potentially risky. When we checked it using our Permission Checker, we notice that it is found to be potentially risky as well. Our checker specifies also the resource accesses causing this alert.

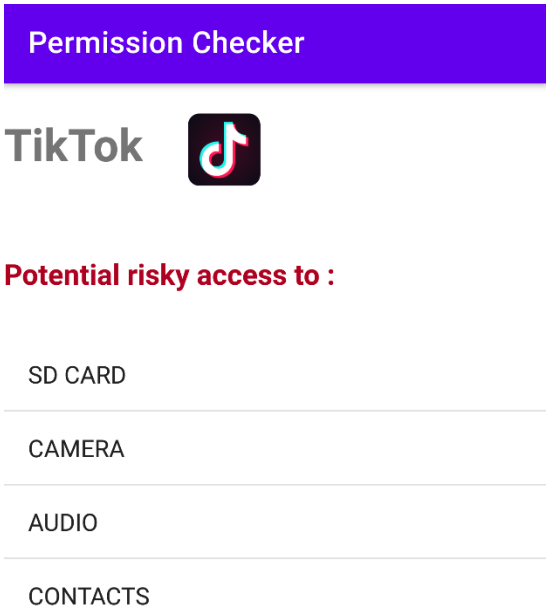

Figure 3. 7: An example of Application Permission Checking (TikTok detail view)

- **The installed app interface**

If the option *open app* is chosen from the item listener in the first activity, the selected app will be opened. We added this option to allow the user run the selected application from within our application after checking its permission.

- **The app info interface**

If the option *app info* is chosen from the item listener in the first activity. The application information are displayed which are: the application consumptions of resources (Figure 3.8.A), its settings (Figure 3.8.B) and its granted permissions (Figure 3.8.C). This view also allows the uninstall of the application if the user is convinced about its risks as well as a button to force quit it. When clicking on the permissions, our application offers the user the ability to select the permissions to grant access and the ability to withdraw access to resources judged risky by our application.

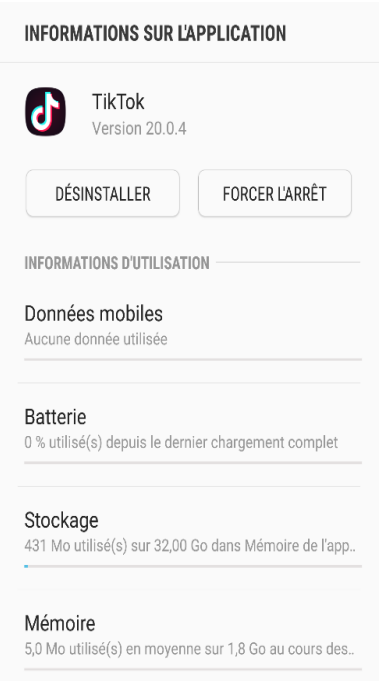

Figure 3. 8. A: Application Information (Resource Consumption View)

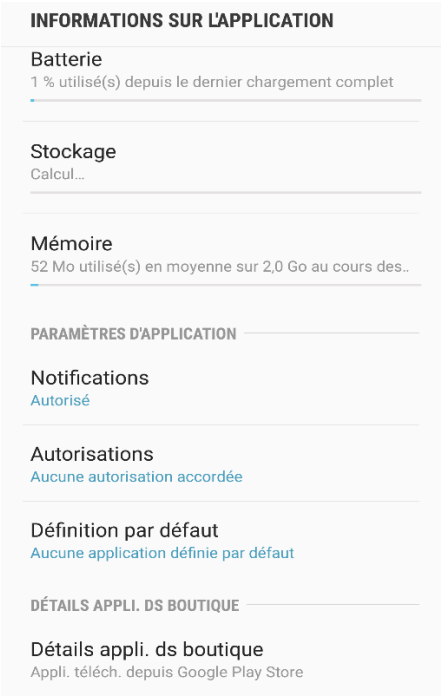

Figure 3. 8. B: Setting View

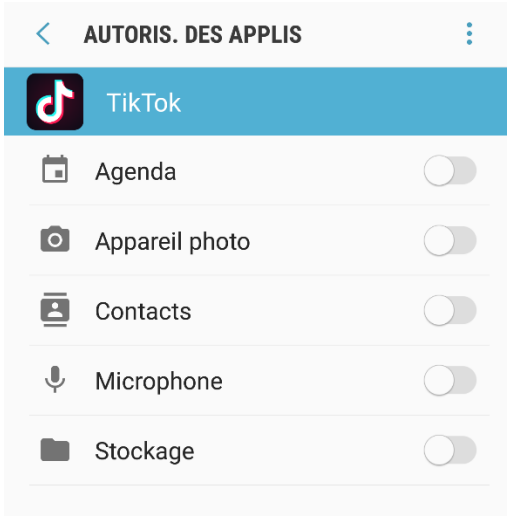

Figure 3. 8. C: *Granted Permissions* Configuration *View*

3.5. Evaluation of Our "Permission Checker" App

The evaluation of our application's performance and correct functionality is one of the most important phases before deploying it on Google store. This phase includes the verification of the application's compatibility with Android OS, checking its views interoperability on various devices sizes and lastly ensuring that it achieves the purpose it was developed for which is to detect risky permission accesses and to enable the user to disable them. An emulator and two mobile devices have been used to test our application's interoperability. Table 3.1 contains the characteristics of these devices:

Table 3. 1: Characteristics of devices used for our applications' compatibility testing

Device	Type of device	OS version	Screen Resolution	CPU	Disk size
Pixel 4 API 30	Emulator	Android 11	1080 x 2280 pixels	X86	10GB
Samsung S6	Physical	Android 7.0	1440 x 2560 pixels	octa-core	32GB
LG Stylus 2	Physical	Android 6.0.1	720 x 1280 pixels	Quad-core 1.2 GHz	16GB

To check the detection, we first used our checker on the applications we analyzed in the previous chapter (chapter 3). The obtained results were identical to those obtained in chapter 3 which means that the application is functioning correctly.

Figures 3.9, 3.10 and 3.11 show the permission checking results given by our application when verifying Instagram, GoodRead and Subway Surfers apps respectively.

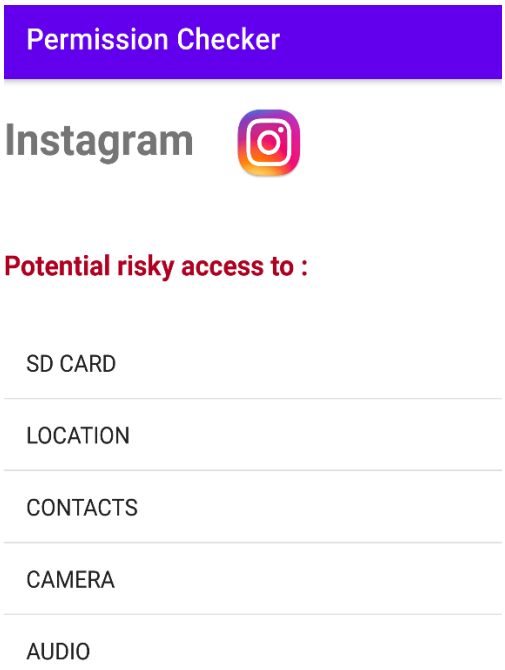

Figure 3. 9: Risky Permissions Granted to Instagram

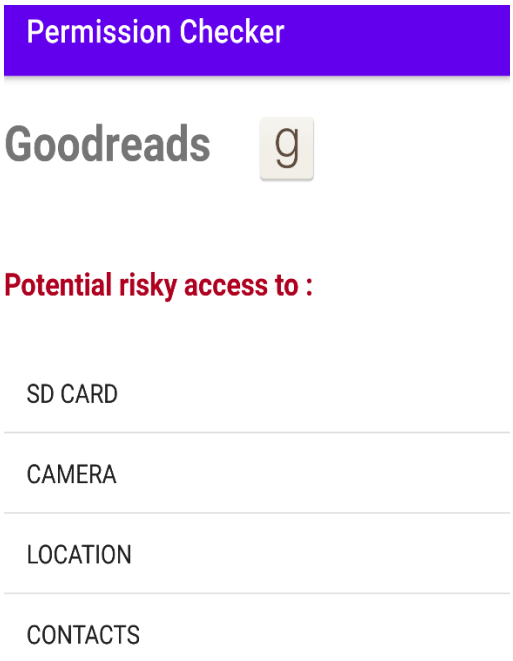

Figure 3. 10: Risky Permissions Granted to GoodReads

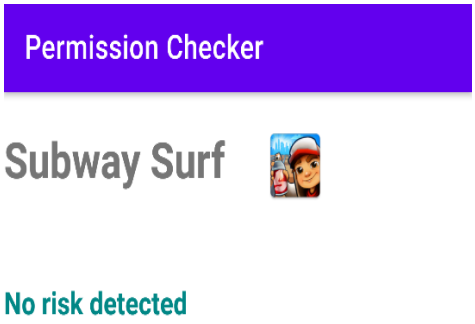

Figure 3. 11: Permission Checking for Subway Surfer Game

To do avoid limiting our functionality correctness testing to the application chosen in chapter 3, we decided to add few more tests on new Algerian developed applications that are commonly used by our citizens within Algeria which are **Jumia** that is used for online Shopping. **Yassir** is Algerian alternative of Uber to reserve Taxis. Figures 3.12 and 3.13 illustrate the results obtained by our checker when analyzing Jumia and Uber respectively. We found that Jumia

access to the Phones SD-Card, it is app to the user benefitting from this application to eliminate this access or to leave it. The user may grant this access only when s/he needs to save data into his/her phone storage and prevent this access otherwise. Yassir app also needed this permission grant besides the location access. The location is indeed needed by this application for its correct functioning. However, it is always desirable to enable this option only when using this application and disable it otherwise. Generally speaking, Algerian application accessed to the least of permissions, i.e. only those needed by the application for its intended functionality.

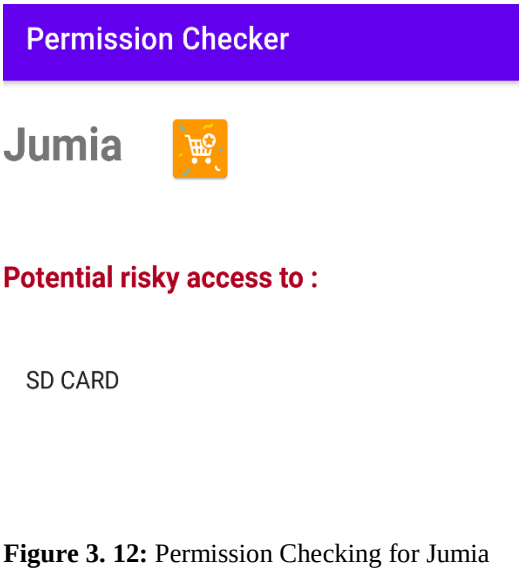

Figure 3. 12: Permission Checking for Jumia

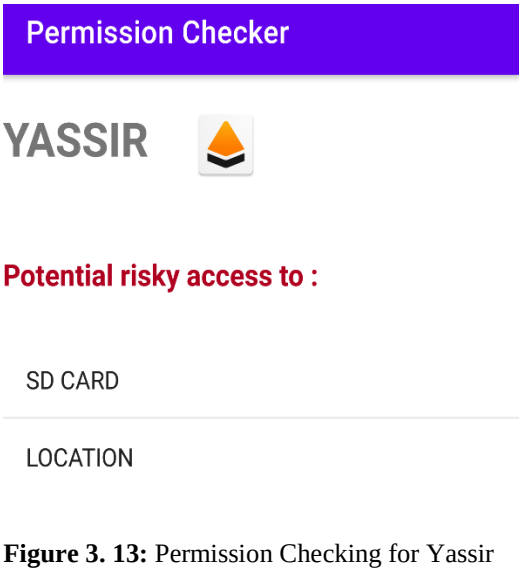

Figure 3. 13: Permission Checking for Yassir

Now that we ensured the correct functionality of our Checker, we need to measure its resource consumption in terms of CPU, memory, and network which is known as the performance evaluation. To monitor our application and measure its performance, we used the Android profiler within Android Studio that provides a real time data on the application resource consumption while running in a testing mode. To do so, we create a new profiling session by selecting a physical device, and then select our application. During the first phase, we started the application and left the phone on idle mode, the aim was to check whether our application is being used as background service or if it is using the phone resources when idle. Noting that we added this test as proof that our application is not abusing the phone resources, we made sure while developing the application to prevent its usage to resources when the phone is idle and the obtained results from the profiler illustrated in figure 3.14 further proves this claim. Figure 15 illustrates an overall view of our application's resource conception in terms of CPU, memory and network. The screen-shot is obtained by the profiler while the application is running.

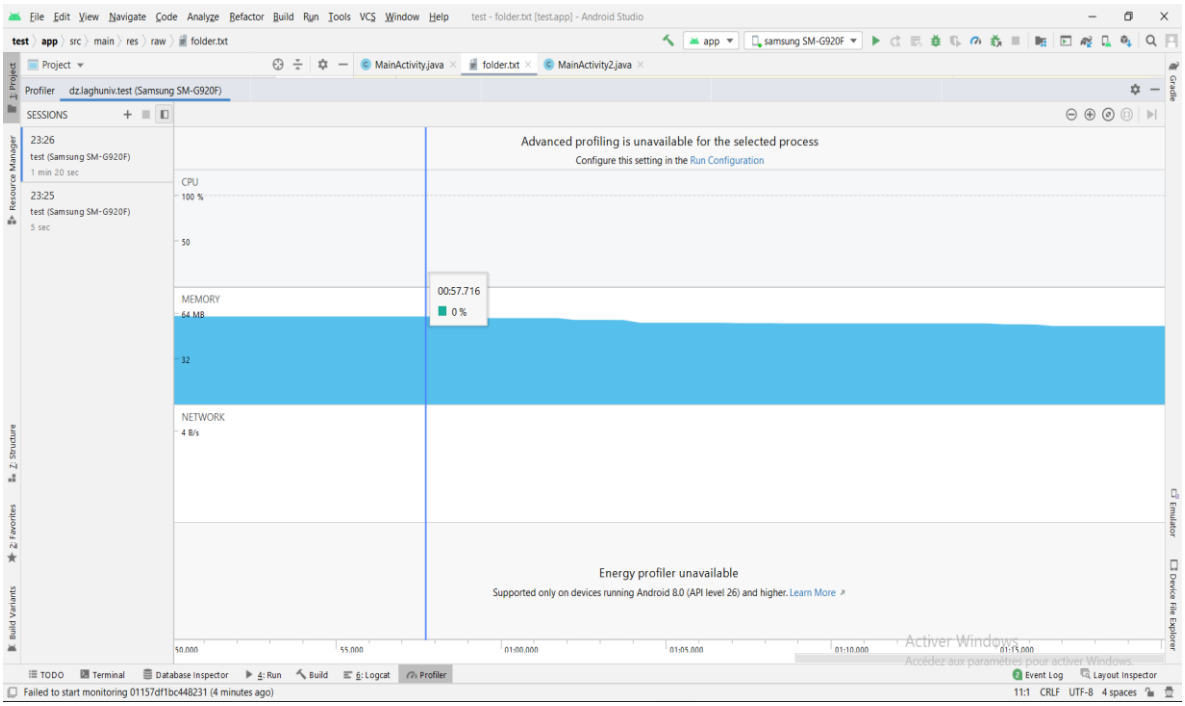

Figure 3. 14: Resource Consumption of our App while it is started and the phone is idle

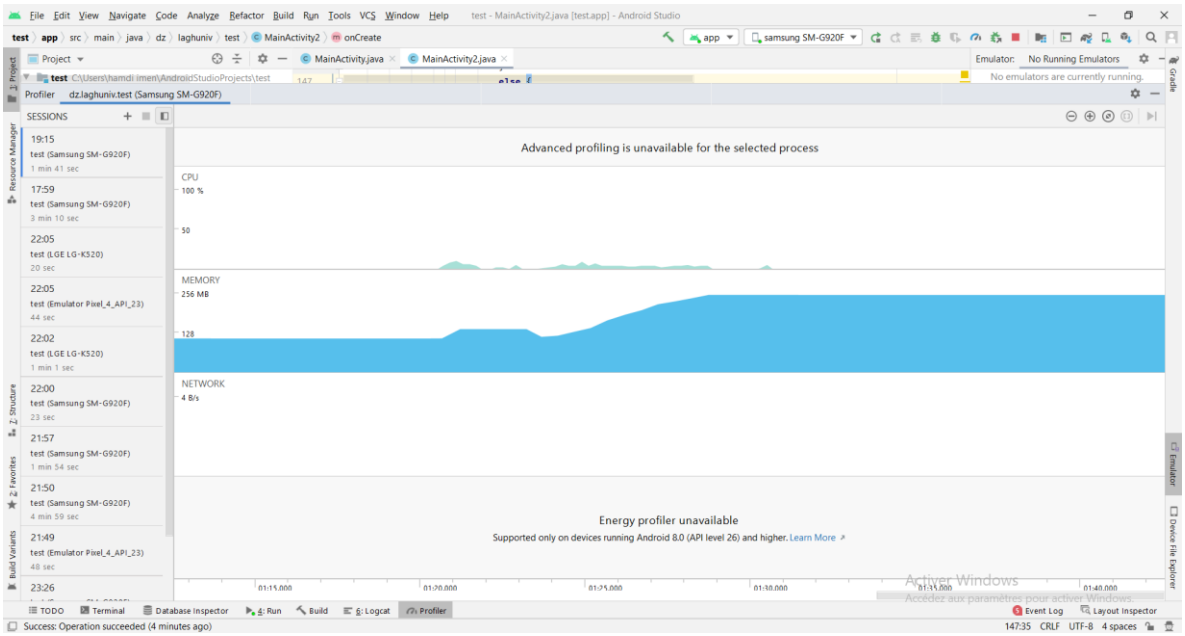

Figure 3. 15: Resource Consumption of our App while it is running

The figure 3.15.1 gives a detailed view on the CPU consumption of our app while it is running, the highest number of running threads is 31. Our app consumed 1% of total available CPU while system processes and other apps have consumed over 73%. This illustrated also that our application is efficient and does not consume much of the device's battery life.

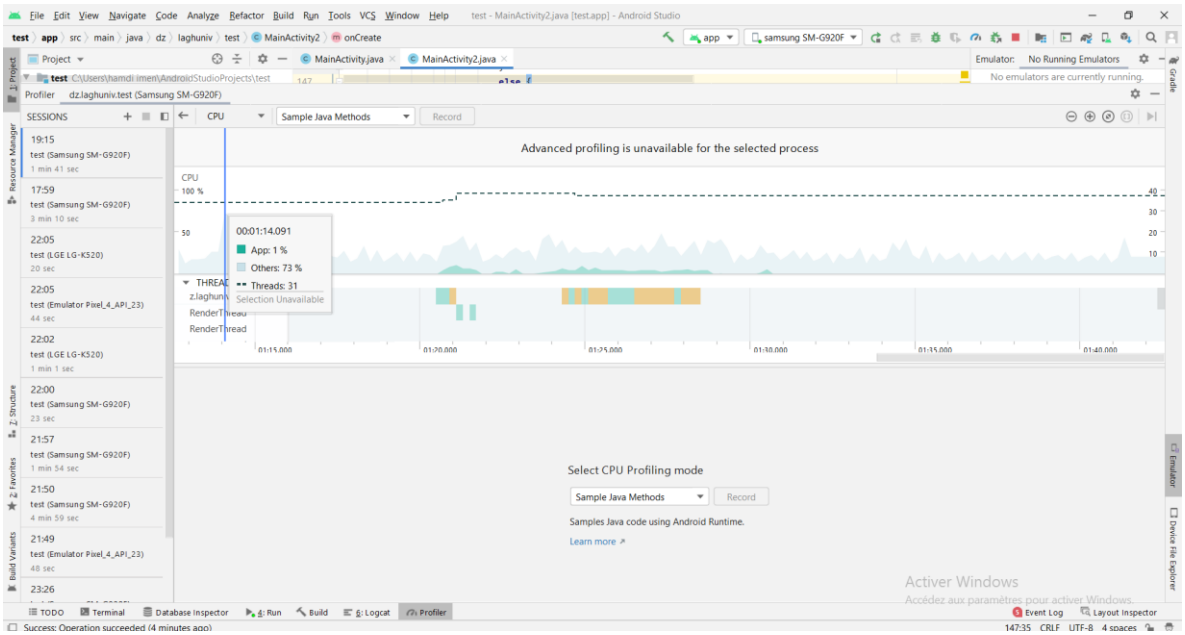

Figure 3. 15.1: CPU Consumption of our app while it is running

In figure 3.15.2, we inspect the use of network by our application, results shows that while the app is running we have 0% of sent and received data during the traffic monitoring, which matches our conception as our application was developed to run offline and does not require internet connection access or use.

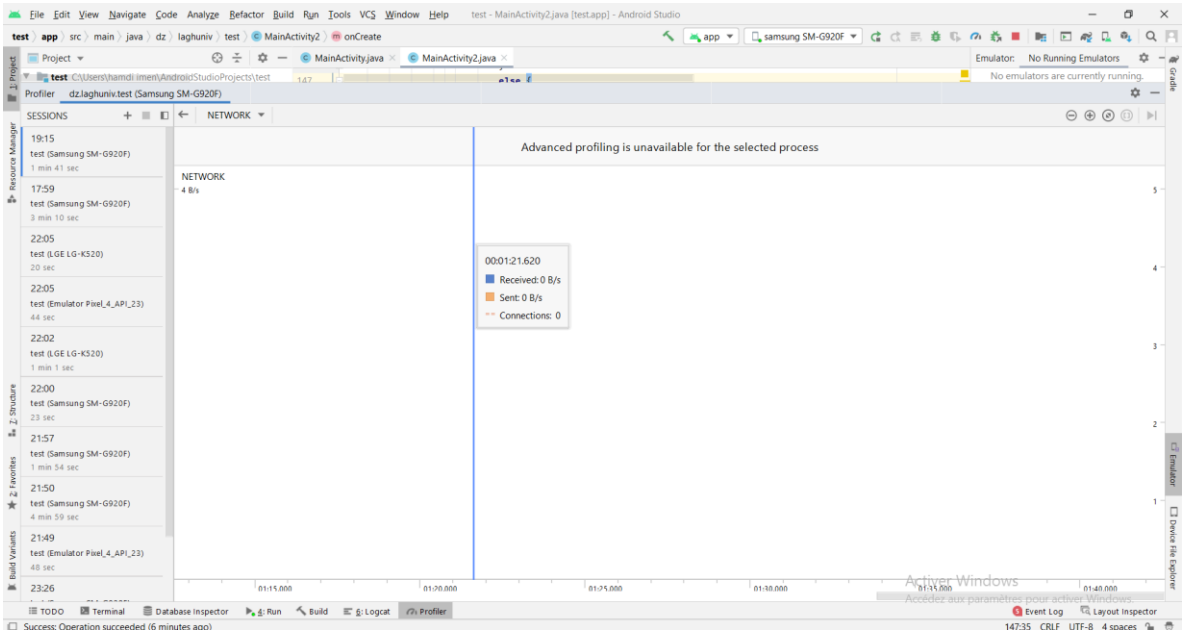

Figure 3. 15.2: Network Consumption of our app while it is running

3.6 Conclusion

To automate the permission risk analysis process that has been performed in the chapter 3, we implemented an android application named “Permission Checker”. This chapter explained its building process and provided the reader with a practical knowledge background of android mobile development steps, it also described the essentially needed software to implement an android application. Our Permission Checker app not only allows the users to check the permission-related risks of installed applications but also enables his/her to disable and enable the access to phone resources as per his/her necessities. The application is tested on analyzed and non-analyzed applications to ensure its correct functionality and its performance was further studied to confirm to the users that our application does not over-consume the phone resources, drain it battery-life or damage his/her device.

Conclusion and Future Perspectives

In this thesis, we introduced the reader to the mobile applications on different mobile operating systems, their evolution history and their growing market value. We also provided a guide on how to build an android mobile phone application. Then, we highlighted the security and privacy risks in mobile phones by referencing to some recent scandals. To raise awareness about the potential risks related to permissions grants and to emphasize to the users the importance to manage them instead of blindly trusting the app developers and randomly grant permission accesses, we first explained to the engendered risks, then, we shed light on the efforts made by the community to detect and protect against permission misuse and abuse. Lastly, we followed their footsteps in investigating commonly used applications and developed an application that would automate the detection process and enables the user to select which permissions to grant to the app and which to not.

The current investigation is not final, another phase is needed to precise in irrefutable manner the risks of the selected applications and the type of exchanged data. For this, more robust full-features monitoring tools are needed which are to be used for longer observation periods. Moreover, we intend to fully automate the detection process basing on the rules we set in chapter 03 and add real-time detection feature to the application as future perspectives to work on.

Bibliography

[1] R. J. Seungyeop Han, "Collaborative Verification of Information Flow for a High-Assurance App Store," *EEE,* pp. 1-14, 2014.

[2] H. K. M. N. A. E. H. Emad Shihab, "What Do Mobile App Users Complain About?," 2014.

[3] "39+ Smartphone Statistics You Should Know in 2020," Review42, 17 11 2020. [Online]. Available: https://review42.com/resources/smartphone-statistics/.

[4] "Endpoint Protection," Juin 2021. [Online]. Available: https://community.broadcom.com/symantecenterprise/network/members/profile?UserKey=909a8e41-f1e7-45af-914a-628128e3819f.

[5] H. C.-Y. W. M. K. G. Lin Ying-Dar, "Mobile Application Security," *Computer,* vol. 47, no. 6, pp. 21-23, 2014.

[6] T. K. W., "Mobile Application Development Experiences on Apple's iOS and Android OS," *IEEE Potentials,* vol. 31 , no. 4, pp. 30-34, 2012.

[7] "Data Privacy and Security: Why Mobile Apps are the New Weak Link," [Online]. Available: ps://www.infosecurity-magazine.com:443/next-gen-infos.

[8] "Intertrust Technologies," [Online]. Available: https://www.intertrust.com/blog/six-of-the-largest-app-related-data-breaches/.

[9] "Android vs. iOS: Which is more secure?," [Online]. Available: https://us.norton.com/internetsecurity-mobile-android-vs-ios-which-is-more-secure.html.

[10] "www.kaspersky.com," [Online]. Available: https://www.kaspersky.com/resource-center/threats/botnet-attacks.

[11] F. Z. H. W. L. Yingjiu., "Permission based Android security: Issues and countermeasures," *Computers & Security ,* vol. 43, pp. 205-2018, 2014.

[12] Z. A. a. M. M. Saudi, "RAPID-Risk Assessment of Android Permission and Application Programming Interface (API) Call for Android Botnet," pp. 1-7, October 2018.

[13] Y. M. S. M. M. R. Farida., "A new mobile botnet classification based on permission and API calls," *IEEE,* vol. 2017 Seventh International Conference on Emerging Security Technologies (EST), pp. 122-127, 2017.

[14] "google play twitter," 27 juin 2021. [Online]. Available: https://play.google.com/store/apps/details?id=com.twitter.android&hl=en&gl=US.

[15] "google play tiktok," [Online]. Available: https://play.google.com/store/apps/details?id=com.zhiliaoapp.musically&hl=en&gl=US. [Accessed 27 juin 2021].

[16] "google play snapchat," [Online]. Available: https://play.google.com/store/apps/details?id=com.snapchat.android&hl=en&gl=US. [Accessed 27 juin 2021].

[17] "google play instagram," [Online]. Available: https://play.google.com/store/apps/details?id=com.instagram.android&hl=en&gl=US. [Accessed 27 juin 2021].

[18] "google play facbook," [Online]. Available: https://play.google.com/store/apps/details?id=com.facebook.katana&hl=en&gl=US. [Accessed 27 juin 2021].

[19] "google play whatsapp," [Online]. Available: https://play.google.com/store/apps/details?id=com.whatsapp&hl=en&gl=US. [Accessed 27 juin 2021].

[20] "google play viber," [Online]. Available: https://play.google.com/store/apps/details?id=com.viber.voip&hl=en&gl=US. [Accessed 27 juin 2021].

[21] "google play true caller," [Online]. Available: https://play.google.com/store/apps/details?id=com.truecaller&hl=en&gl=US. [Accessed 27 juin 2021].

[22] "google play netflix," [Online]. Available: https://play.google.com/store/apps/details?id=com.netflix.mediaclient&hl=en&gl=US. [Accessed 27 juin 2021].

[23] "Google play youtube," [Online]. Available: https://play.google.com/store/apps/details?id=com.google.android.youtube&hl=en&gl=US. [Accessed 27 juin 2021].

[24] "google play youtube kids," [Online]. Available: https://play.google.com/store/apps/details?id=com.google.android.apps.youtube.kids&hl=en&gl=US. [Accessed 27 juin 2021].

[25] "google play goodreads," [Online]. Available: https://play.google.com/store/apps/details?id=com.goodreads&hl=en&gl=US. [Accessed 27 juin 2021].

[26] "google play kalimat," [Online]. Available: https://play.google.com/store/apps/details?id=com.zytoona.wordscrush&hl=en&gl=US. [Accessed 27 juin 2021].

[27] "google play subwaysrfers," [Online]. Available: https://play.google.com/store/apps/details?id=com.kiloo.subwaysurf&hl=en&gl=US. [Accessed 27 juin 2021].

[28] "google play candy crush," [Online]. Available: https://play.google.com/store/apps/details?id=com.king.candycrushsaga&hl=en&gl=US. [Accessed 27 juin 2021].

[29] "google play magic tiles3," [Online]. Available: https://play.google.com/store/apps/details?id=com.youmusic.magictiles&hl=en&gl=US. [Accessed 27 juin 2021].

[30] "google play garena free fire," [Online]. Available: https://play.google.com/store/apps/details?id=com.dts.freefireth&hl=en&gl=US. [Accessed 27 juin 2021].

[31] "kaspersky.com," 2021. [Online]. Available: https://www.kaspersky.com/resource-center/threats/botnet-attacks. [Accessed 24 JUIN 2021].

[32] "Permissions on Android," 2021. [Online]. Available: https://developer.android.com/. [Accessed 25 june 2021].

[33] "developer.android.com Project compenent," Juin 2021. [Online]. Available: https://developer.android.com/studio/projects.

[34] "developer.android.com Guide," Juin 2021. [Online]. Available: https://developer.android.com/guide/components/activities/activity-lifecycle.

[35] "Google Play," 2021. [Online]. Available: https://play.google.com/store/apps. [Accessed 24 juin 2021].