Cloud based malware detection with on-device network traffic analysis for enhancing Android user's Privacy

Security and Privacy in Android Smartphones

[1] Ashok Kumar .S, [2] Kaviarasu. V, [3] Kathirvel. K, [4] Lakshmi Narasimha Rao .K

[1] Assistant Professor, M.E , [2,3,4] B. Tech Graduate,
[1,2,3,4] Computer Science & Engineering,
[1,2,3,4] Manakula Vinayagar Institute of Technology, Puducherry, India.

Abstract: As we all know that the Android Platform is developed by Google for mobile devices with powerful processing capabilities. Its kernel is based on Linux OS. The android applications run in a sandbox. Nonetheless, many organizations such as Kaspersky, McAfee, and AVG Technologies, etc., have released the android version of their antivirus software products. Even though the antivirus application runs under the sandbox, it has a limit to scan the environment. This Project represents the creation of an Application for android platform users to protect their privacy and safeguard their data. To achieve this, we have created a firewall-like app which is a perfect solution if users would like to preserve their privacy. A cloud-based malware scanning can be implemented to prevent malware infections provided with a network traffic analysis mechanism that can be used to monitor the traffic flow through a Local VPN Tunnel to identify the outgoing packets per-app running on the device by using the Antmonitor Library. By implementing a local VPN, the traffic can be analyzed without the need for superuser permission to the device.

Index Terms **- Android app, Privacy, Network Traffic Analysis, Ant-Monitor Library, Malware scanning, etc.**

I. INTRODUCTION

The calculated number of mobile devices is around 5.8 billion, which is thought to have grown exponentially within five years and is supposed to reach nearly 12 billion within four years. Hence, it will be an average of two mobile devices per person along with the planet. This makes us fully dependent on mobile devices with our sensitive data being carried all over. As an effect, mobile security is one of the most important concepts to take into consideration. Mobile Device Security implies that the protection of the mobile device from various potential security breaches to the sensitive data stored in the mobile device. On that point are various levels in which several mechanisms to secure the data are implemented properly to thwart various attacks that arise concerns related to the privacy of the sensitive data stored in the device which are accessed by various applications. It is the same for Android-based devices especially smartphones. The written reports and surveys conducted by various researchers show that problems related to leakage of data became a major concern for both organizations and application developers for the android platform. Even though in that respect are several security mechanisms implemented and are revamped over the years, they are unable to properly secure the data used by the android applications, so a simple and secure system needs to be created for securing data.

Our project proposes a new enhanced security model to protect the information shared by multiple applications, SecDroid, can be called as a simply android firewall which uses the android local VPN service which forces the android device to re-route the network traffic to flow through this application and then connect to the internet for analyzing the data flow used by every app in the android device thus providing transparency over the data sent and received with their precise location where the data is being sent and received with a timestamp for proper identification. This project uses open-source tools and libraries for the overall execution. On some other note, it also provides a means to efficiently identify the malware with the help of the infamous cloud-based analyzer known as Virustotal. It has 70+ engines for malware scanning. Since the scanning process takes place in the cloud, resource use is relatively low, thus it does not impact the device performance.

II. OVERVIEW OF INFORMATION SECURITY

Information Security also simply called infosec is the practice of protecting the information by mitigating the information risks that are present. It involves preventing the possibility of the information being leaked or accessed without any authorization. The core of information security is confidentiality, integrity, availability. Every organization needs measures to protect their data, so as the android smartphone users. Information Security provides various practices that will help to secure the information stored. A computer is any device with a processor and some memory, such devices can range from no network-based standalone devices such as calculators to networked mobile computing systems such as smartphones and tablet computers. Hence, Information Security plays a critical role in protecting the user’s privacy. Android Device Security also one of the categories in the information security.

III. ANDROID SECURITY MECHANISMS

Before elaborating on this project, we should provide a brief introduction to the Android platform and its incorporated security functions mechanisms. The fundamental information will help to understand the challenges and threats in the Android platform. It’s vital for solving new problems raised by Android applications by using various security analysis methods and technologies.

1. Android Platform

Android is known as the open-source mobile operating system based on Linux Kernel and designed primarily for smart devices. Android system has a hierarchical structure that consists of a Linux kernel layer, library layer, application framework layer, and application layer. Linux kernel layer provides some basic such as memory management, process management, and network protocols. This layer contains the core drivers for all underlying devices of the hardware components. The Library layer provides the core library that includes the original library and third-party library for apps to assist the application framework layer. The application framework layer is equivalent to an intermediate layer that intelligently coordinates the components, which enhances the flexibility of the entire system. To complete this work, the application framework contains many system services such as Activity Manager, Window Manager, Resource Manager, Location Manager, Content Provider, and so on. The application layer, the only layer that can interact with users, consists of all apps running on Android devices. There exist solutions with secure protocols for IoT security. However, in this work, we focus on application security in the Android system deployed in smart devices.

2. Android Applications

An Android app is written in Java programming language using APIs provided by the Android Software Development Kit (SDK). Besides the Java code, an app may also contain some native libraries that are provided by the Android system or implemented by developers. Android Application Package (APK) is the package file format used by the Android operating system for distributing and installing mobile apps, mobile games, and middleware. If an APK is installed on an Android device, it runs by using the Android runtime environment. Android apps contain four main components: Activity, Broadcast Receivers, Service, and Content Provider. Activity indicates the User Interface and handles the user interaction with the smartphone screen. Broadcast Receivers deal with communication between operating systems and apps. Service manages the background processing of an app to perform long-running operations. Content Provider provides data sharing across apps.

3. Built-in Security Mechanisms

Developers are constantly introducing various security mechanisms when they design the Android platform. Android system has a hierarchical structure, and each layer has its security mechanism, namely, a traditional access control mechanism, a mechanism based on permissions management, a sandboxing security mechanism, a digital signature verification mechanism, and an encryption-based security mechanism.

3.1 Access Control

The traditional access control mechanism is equivalent to Android's Linux kernel security mechanism. Access control restricts the subject (such as users or services) to access the object (such as resources). It is a primary approach to protect the confidentiality and integrity of data. Access control includes two kinds of methods, mandatory access control (MAC) and discretionary access control (DAC). MAC is implemented by the Linux security module. DAC is implemented by file access control. These are some built-in access control mechanisms.

3.2 Permissions Management

Android uses a permission-based security model to restrict apps to access some resources. If apps want to use restricted resources, they have to apply for permission through XML files. Android permissions have four levels, namely Normal, Dangerous, Signature, and System. Low-level permissions, including normal and dangerous levels, are authorized as soon as an app applies. Signature level and signature/system-level permissions are known as advanced permissions. Before an app can apply for these permissions, it needs to achieve platform-level authentication. However, there are many shortcomings to this mechanism. Users need to decide if the permissions that an app applies should be authorized, yet some users may not have enough knowledge to judge it. Moreover, if the device is running on Android 5.1.1 or lower, or the app's target SDK Version is lower than 22, the system automatically asks the user to grant all dangerous permissions for the app at install-time. once the permissions are authorized, they will remain valid until the app is removed from their device or unless the users change.

3.3 Sandbox Based Security Mechanism

A sandbox is simply an isolated space that is used to run apps in the Android system. A sandbox provides very firm control over the set of resources for apps to run. During the execution, Android uses the Dalvik virtual machine to execute the apps, for which each app uses its own process space and resources. Therefore, different apps cannot interact with each other and cannot access each other's' resources and memory space.

4.) The Mechanism in Digital Signature

The digital signature mechanism plays a very important role in the security of the application layer. Android app developers have to sign their apps with digital signatures since the apps without the digital signatures the apps are not allowed to be installed. If an attacker deliberately changes the internal file of APK using some reverse engineering tactics, he/she has to re-sign the app. The signature should be the same as the developer's original signature. Also, the signatures of apps will be checked when apps need to be updated. The digital signature ensures the integrity and reliability of android apps.

IV. Malware Detection Methods

Existing analysis methods for detecting Android apps mainly consist of static, dynamic, hybrid, and meta-data analysis. We introduce these analysis methods briefly and classify the surveyed papers according to the taxonomies of their employed features. Experts from Kaspersky, Bitdefender, and Malwarebytes Labs discussed the increasing complexity of malware detection methods when dealing with threat factors of variable skill levels, and where detection trends need to be pursued in the future. There was agreement among experts that hash-based or signature-based malware detection methods should mostly be a thing of the past.

1. Static Malware Analysis

Since the Android app has surged dramatically, the Android platform becomes the target of attackers. In response, much research work aims to detect malaprops via static analysis. In static analysis, apps are firstly unpacked and decompiled into files that present essential information about the apps. Then these files are examined that if there are malicious codes. Static analysis is well known in traditional malaprop detection, gaining popularity as efficient mechanisms for market protection. It is useful for resource-restricted Android devices as the analysis is performed without running the app. Static analysis consumes much fewer resources and time. It is thus a relatively fast method. However, this approach can be hindered by malaprops that use reverse engineering techniques, such as obfuscation and repackaging.

2. Dynamic Malware Analysis

In contrast to static analysis, the dynamic analysis is used to find the malicious behaviors after deploying and running the apps on the emulators or real devices. It generates snapshots of processor execution, network activity, system calls, SMS, phone calls, etc. to differentiate malaprops from good ones. This technique requires some human or automated interaction with apps, as malicious behaviors are sometimes triggered only after certain events. The information observed through dynamic analysis reflects the app's actual execution. However, the dynamic analysis execution leads to excessive use of resources.

V. Literature Survey

1. A group of researchers had created a permission recommendation system which is designed based on crowdsourcing can help the android user to make safe decisions based on the expert opinions collected by the transitional Bayesian inference model to apply permissions to the android applications using the expertise opinion system created using the inference algorithm (Bahman Rashidi, 2017). Even though this model provides the users a permission recommendation system, some applications may have malware code in it which may look like a legitimate but force the user to give all the permissions to execute, in this case, the Bayesian algorithm fails to function properly, this flaw can be eliminated by our new application.

2. An application called WallDroid: Firewall for the Android OS was developed by Abhijeet Awade, Amir Talwar, Bhushan Khopade, and Vishal Nande of B.E Computer Engineering, Navsahyadri Education Society's Group of Institutions, Pune. This project proposes and evaluates an enhanced security model and architecture, WallDroid, enabling virtualized application-specific Firewalls. The WallDroid application can be considered as an Android Firewall Application but with some extra functionality. Key components used by the solution include VPN technologies like the Point to Point Tunneling Protocol (P2P) and the android Cloud to Device Messaging Framework (C2DM). Their project is based on the cloud, keeping track of millions of applications and their reputation (good, bad, or unknown) and while comparing network traffic flows of applications with a list of known malicious IP servers (Abhijeet Awade, 2017). This is similar to our application but they are using the data usage monitoring system to catch the harmful apps which take advantage of the application permissions to do malicious activities such as sending data to a private IP address or some third-party domains. But this lacks in providing the user the necessary information about the detected apps' source and destination IPs' and a separate malware detection engine which enforces more security for the android device.

3. In the journal Constructing Features for Detecting Android Malicious Applications: Issues, Taxonomy, and Directions published by IEEE, the authors such as Wei Wang, Meichen Zhao, Zhenzhen Gao, Guangquan Xu, Hequn Xian, Yuanyuan Li, and Xiangliang Zhang have provided a state-of-art survey on the various security mechanisms present in the android smartphones and the types of malware with the different types of malware analysis techniques including various algorithms used for malware detections and analysis techniques (Wei Wang, 2019). But they didn't specify a particular technique for android malware detection.

VI. Existing System

Many android apps are capable of protecting the data in an android device from malware infection. Some apps are available in the market that was developed for the analysis of the data traffic, they are proprietary and they lack the malware detection feature. The Built-in security features which come with the device require manual scanning and do not offer real-time protection to keep the data safe. This the main reason many users are forced to use a third-party security solution. Occasionally the built-in engines may show true-negatives and do not detect the malware properly. The apps available in the android market such as AF Wall, Netguard, etc., have the features that help to protect the data but they aren't capable of protecting the privacy of android users completely.

VII. Proposed System

Our proposed system has all the necessary features which can help the Android smartphone users to protect their privacy. By using the Android Local VPN service with the help of the Ant monitor Library which in turn emulates a VPN service through which all the data traffic in the device flows through to access the internet. With this in mind, we created a new way of monitoring the network traffic of all the applications in the device using the antmonitor library. The IP address where the data is sent or received are displayed with date and timestamp. We also added the malware detection feature with the help of Virus total API. It has a 50+ antivirus scanning engine that works in the cloud to detect the malware ignoring the possibility of true-negatives. The malware scanning requires internet permission for scanning on the cloud. The real-time traffic interception works only in the device and no data will be sent to the external servers. The Antmonitor library has many classes and methods for analyzing internet traffic data generated by various applications in the android device. A new TUN interface will be created for intercepting networking traffic using the mentioned library.

VIII. Architecture Diagram

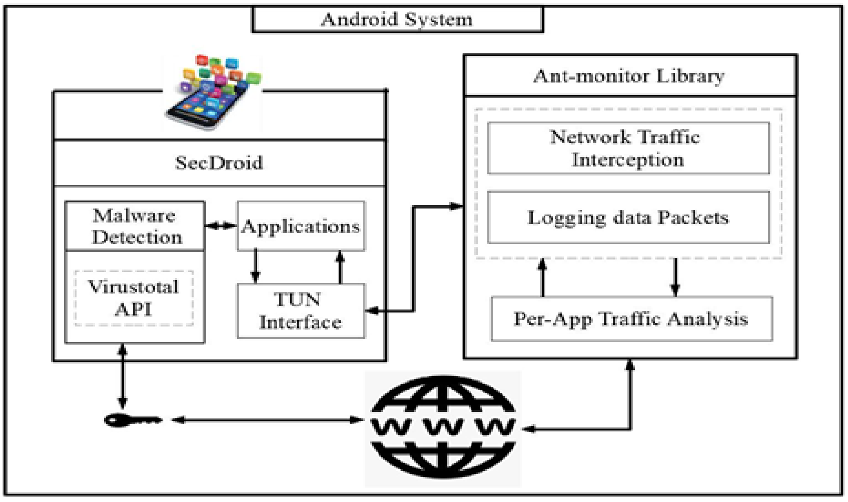


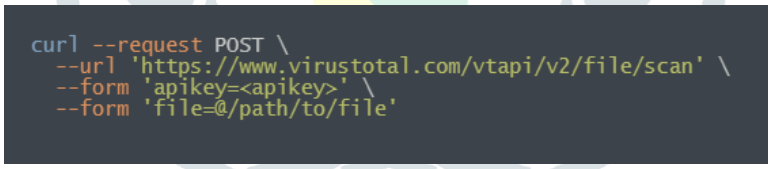
Figure 1: This image represents the architecture design of our application.

IX. Implementation

Ant-monitor Library: Ant-monitor (Shuba, 2016) is a library that runs as a service and passively monitors all the incoming and outgoing packets of the network interface. We used the antmonitor library which uses a local VPN service that creates a tunnel interface to intercept the entire device's traffic. It uses antpcaplib, a packet capturing library for android to intercept data traffic of the device. Ant-monitor provides various features which include dumping the traffic in pcap format, uploading it to a remote server, logging internet traffic and filtering out packets. The monitor can be used as a tool to support many passive monitoring applications, including real-time detection and prevention of private information leakage from the device to the network, ad blocking and also the network performance measurements.

Virustotal API: The Virustotal API lets you upload and scan files or URLs, access finished scan reports, and make automatic comments without the need for using the website interface. In other words, it allows you to build simple scripts to access the information generated by Virustotal. Public and Premium API keys are provided by virustotal for developers. Depending upon the public or premium API keys, various endpoints can be accessed by the developer. Scanning files, URLs, getting the report, analyzing the behavior of the file, scanning the domains and IP's are provided by the virustotal. File upload and report endpoints are illustrated below with curl command.

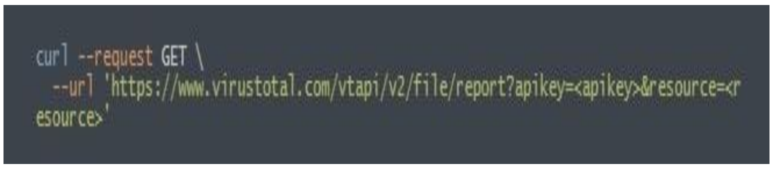
File scan-endpoint: This endpoint is used for scanning files in the device by uploading it to the virustotal cloud.



```
curl --request POST \
  --url 'https://www.virustotal.com/vtapi/v2/file/scan' \
  --form 'apikey=<apikey>' \
  --form 'file=@/path/to/file'
```

Figure 2: This is image of the script used in scan endpoint using curl.

Report-endpoint: This endpoint is used to get the report of the file uploaded for scanning in the virustotal cloud.

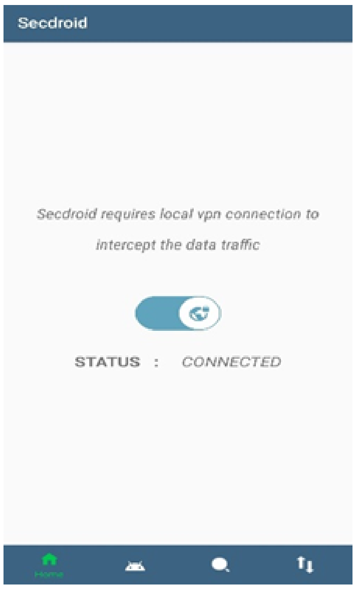


```
curl --request GET \
  --url 'https://www.virustotal.com/vtapi/v2/file/report?apikey=<apikey>&resource=<r
esource>'
```

Figure 3: This is image of the script used in report endpoint using curl.

X. Working of the Application

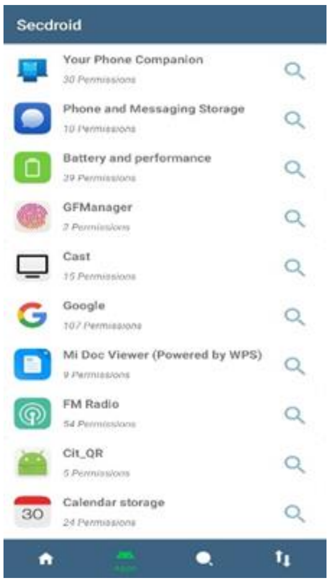
1. On Launching the application, the user is presented with a switchcompat button, when toggled on it triggers the local VPN service to start intercepting the outgoing traffic. All the traffic is made to flow through our app which helps in intercepting traffic. The status of the VPN is notified to the user on the screen. This Local VPN service does not send or receive data from/to a third-party resource. It is needed to eliminate the root permission in android phones to intercept the traffic in real-time (Refer Figure-4).


Secdroid

Secdroid requires local vpn connection to intercept the data traffic

STATUS : CONNECTED

Figure 4: VPN Toggle Button – Initiates local VPN service.


Secdroid

Your Phone Companion
30 Permissions

Phone and Messaging Storage
10 Permissions

Battery and performance
29 Permissions

GFManager
2 Permissions

Cast
15 Permissions

Google
107 Permissions

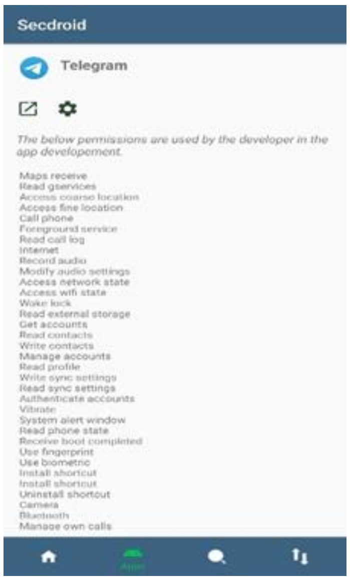
Mi Doc Viewer (Powered by WPS)
9 Permissions

FM Radio
54 Permissions

Cit_QR
5 Permissions

Calendar storage
24 Permissions

Figure 5: List Apps - Displays all applications present in the device.


Secdroid

Telegram

The below permissions are used by the developer in the app developement.

Maps receive
Read gservices
Access coarse location
Access fine location
Call phone
Foreground service
Read call log
Internet
Record audio
Modify audio settings
Access network state
Access wifi state
Wake lock
Read external storage
Get accounts
Read contacts
Write contacts
Manage accounts
Read profile
Write sync settings
Read sync settings
Authenticate accounts
Vibrate
System alert window
Read phone state
Receive boot completed
Use fingerprint
Use biometric
Install shortcut
Install shortcut
Uninstall shortcut
Camera
Bluetooth
Manage own calls

Figure 6; Permission List – Lists permission used in manifest file.

2. We used the bottom navigation view to simplify the features for easy access to the app's functions. This activity displays the list of applications present in the device using the recyclerview layout. The list of installed applications is provided by the Package Manager class in android (Refer Figure-5).

3. The Permissions that are mentioned in the manifest file of the respective applications are shown to the user. A launch shortcut and settings shortcut button in this activity are used to launch the corresponding application and settings of the same respectively (Refer Figure-6).

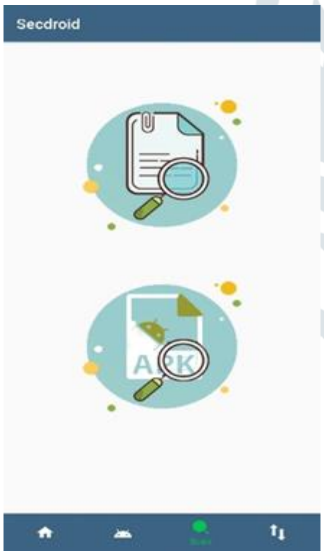
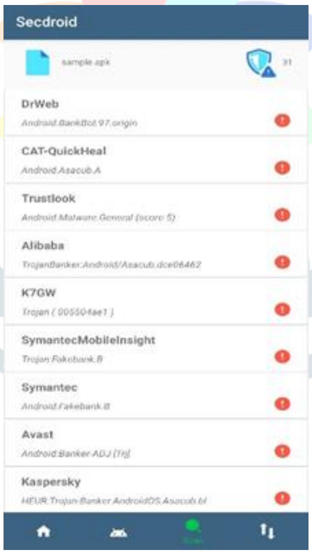


Figure 7: Scan – To scan files or installed applications.


Secdroid

sample.apk 31

DrWeb
Android.BankBot.97.origin

CAT-QuickHeal
Android.Asacub.A

Trustlook
Android.Malware.General (score: 5)

Alibaba
TrojanBanker:Android/Asacub.dce06462

K7GW
Trojan (005504ae1)

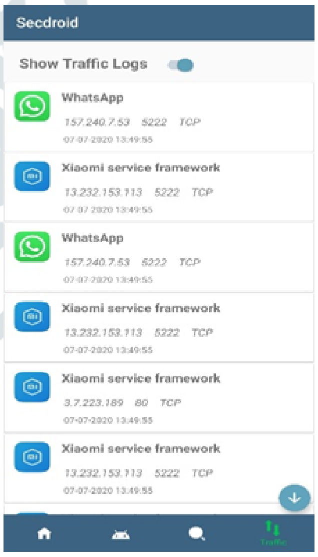
SymantecMobileInsight
Trojan:Fakebank.B

Symantec
Android.Fakebank.B

Avast
Android:Banker-ADJ [Trj]

Kaspersky
HEUR:Trojan-Banker.AndroidOS.Asacub.bl

Figure 8: Scan report – Detected as malware by many AV engines.


Secdroid

Show Traffic Logs

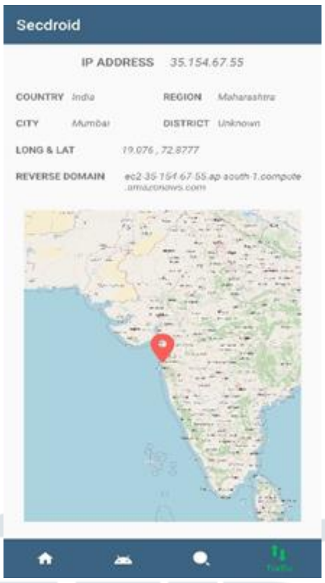
```
WhatsApp
157.240.7.53   5222   TCP
07-07-2020 13:49:55

Xiaomi service framework
13.232.153.113   5222   TCP
07-07-2020 13:49:55

WhatsApp
157.240.7.53   5222   TCP
07-07-2020 13:49:55

Xiaomi service framework
13.232.153.113   5222   TCP
07-07-2020 13:49:55

Xiaomi service framework
3.7.223.189   80   TCP
07-07-2020 13:49:55

Xiaomi service framework
13.232.153.113   5222   TCP
07-07-2020 13:49:55
```

Figure 9: Traffic Logs – Outgoing traffic of all applications.

4. Scan activity prompts the user an alert dialog to enter their API key at the start then only other options are shown to the user. User can select either a file scan or an apps scan. On selecting the file scan option, it triggers the file provider intent to choose the file for scanning. On the contrary, choosing the APK scan option forces the user to select the installed application in the previous activity. The POST request can be made to the **file/scan** endpoint along with the file chosen. Then we get a response with resource id which can be used to retrieve the scan report (Refer Figure-7).

5. Resource id returned from the scan activity is the key. The report of the file scanned can be obtained by making GET request to **file/report** endpoint along with the resource id as a param. The response from the report endpoint is in the form of JSON object which can be processed with the help of the android volley library. The result will be displayed to the end-user neatly in the recycler view, *green* indication means the file is *not malicious*, in-turn *red* indicates that the file is *malicious* (Refer Figure-8).

6. While the local VPN service is active, the intercepted traffic will be displayed to the user only when the show traffic log switch is toggled on. The IP address along with the Port number of the app that tries to send data to the internet are displayed with the timestamp. This feature helps in identifying applications which send malicious traffic. The outgoing traffic is displayed until local VPN is connected (Refer Figure-9).


Secdroid

IP ADDRESS 35.154.67.55

COUNTRY India
REGION Maharashtra
CITY Mumbai
DISTRICT Unknown
LONG & LAT 19.076 , 72.8777
REVERSE DOMAIN ec2-35-154-67-55.ap-south-1.compute.amazonaws.com

Figure 10: IP Lookup – The detailed info about IP address.

7. Additional info about the IP Address can be queried using IP-API (Refer Figure-10).
   - The GET request can be made to the IP-API with IP as param to get the response. Response includes country, city, region, latitude and longitude as JSON object. The location is resolved as Latitude & Longitude then displayed to the user using the osmdroid library, an alternative to the mapview in android.
   - osmdroid uses OpenStreetMap API to display the location and it is free and community maintained.

XI. Future Work

Some features of the application can be upgraded. For instance, by default, any Virustotal registered user is entitled to a public API key that allows them to interact with a limited set of endpoints. Some request calls are available only via the premium API using that m features can be made. The traffic analysis could be further improved by implementing some new features such as packet filters to block/filter unnecessary packets, sending the log file to the remote server for further analysis and allowing applications request to connect to the internet on demand for each app can also be added as a new trait. Bug fixes and code optimizations can increase the performance a lot.

References

[1] Wei Wang, Meichen Zhao, Zhenzhen Gao, Guangquan Xu, Hequn Xian, Yuanyuan Li, Xiangliang Zhang. May 2019. Constructing Features for Detecting Android Malicious Applications: Issues, Taxonomy, and Directions. IEEE Access. Special Section of Security and Privacy for Cloud and IoT, vol. 7, pp. 67602 - 67631.

[2] Abhijeet Awade, Amir Talwar, Bhushan Khopade and Vishal Nande, Jan 2014. WallDroid: Firewall for the Android OS. International Journal of Engineering Research & Technology (IJERT)., vol. 3, no. 1, pp. 1677–1682

[3] Bahman Rashidi, Member, IEEE, Carol Fung, Member, IEEE, Anh Nguyen, Tam Vu, Member, IEEE, and Elisa Bertino, Fellow, IEEE. Mar 2018. Android User Privacy Preserving through Crowdsourcing. IEEE THz Sci. Technol., vol. 13, no. 3, pp. 773-787.

[4] Antmonitor example, used as a reference to create this project's traffic interception mechanism. Available as a GitHub project under UCI-Networking-Group at https://github.com/UCI-Networking-Group/AntMonitorExample.

[5] The guide on how to use the Virustotal API for detecting various malwares which we have mentioned in this research paper is available at https://developers.virustotal.com/reference.