Android Malware A Oversight on Malware Detection Using Machine Learning

Yash Kanchhal
Dept. of Networking and Communication
SRM Institute of Science and Technology
Kattankulathur, Chennai, India 603203

Dr. S Murugaanandam*
Dept. of Networking and Communication
SRM Institute of Science and Technology
Kattankulathur, Chennai, India 603203

2022 International Conference on Computer Communication and Informatics (ICCCI) | 978-1-6654-8035-2/22/$31.00 ©2022 IEEE | DOI: 10.1109/ICCCI54379.2022.9741025

***Abstract*—The android operating system has been the most popular mobile operating system for more than a decade. Android has also caught the attention of cybercriminals and malware developers, which has increased security threats too. Malware is one of the severe issues for all operating systems counting Android also. Since Android supports the installation of applications from non-Google Play Store services, this can lead to the installation of malware applications along with benign applications. In this paper, we will be creating an Android malware application that we will inject into an Android device or emulator where the malware is out of sight from the victim user, and we will be collecting critical data from the victim system. Along with that, we will be detecting the malware using Random Forest machine learning.**

***Keywords*—Android Malware, Detection, Machine Learning, Random Forest**

I. Introduction

The most essential tool that we use in our day-to-day life is smartphones, on which we perform our crucial activities. Android, the most popular mobile operating system, has a 72% market share [1], due to its being free and open-source. Applications available on the Google Play Store are easy to download. It additionally permits the application to be downloaded from a third party, being open-source. There were approximately 2.89 million Android applications available in the first quarter of 2021 on the Google Play Store to choose from, making the Google Play Store with the vast number of available apps [2] these numbers are estimated to only increase in the coming years. Mobile application growth on Google Play Store has also increased by 10.60% compared to the previous quarter [3]. Users worldwide have downloaded approximately 108.5 billion mobile apps on the Google Play Store in 2020, up from 76 billion apps in 2018 [4]. Android's popularity has caught the attention of cybercriminals, which has increased the security threats such as malware and rootkits. Thus, the problem of accurately detecting Android malware has also increased.

Malware is any software that's intended to produce damage and destroy computers systems and computer networks. It is an acronym for malicious software such as viruses, worms, Trojan viruses, spyware, adware, and ransomware [5]. Cybercriminals generally use malware to obtain any information over their victims for monetary gain. This information can vary from monetary, healthcare records, private emails, and passwords. The possibility of what varieties of information can be compromised has become infinite. Malware developers now have become skilled in creating powerful malware that will be enough to infect any kind of computer system, and therefore studies are going on for the prevention of malware for computer systems.

The process of scanning the computer system and files to detect anomalies is called malware detection [6]. Sometimes this process may involve the use of multiple tools and software, which increases the complexity of this process. Malware detection techniques for smartphones are straggling far off in contrast to the growing population of malware found using several methods of attack.

Malware can be injected into a system in many ways such as email attachments, malvertising, fake software, infected USB, infected applications, phishing emails, and spam text messages.

To distinguish between a harmless application and a malicious application we can utilize a machine learning procedure based on numerous features that can be obtained from distinctive applications. In this research, we aim to propose a robust model using static malware features of android applications which will help in distinguishing novel and unknown android malware.

The reaming part of the paper is structured in the following sequence in section 2, is brief background, section 3, a synopsis of past similar work, section 4, the advised system is condensed, section 5, the expected outcome of the research is provided, and conclusion of the paper with suggestions for future work.

II. Background

Developed by Google the Android is an open-source Linux-based mobile operating system. It was revealed in November 2007, and it had its first commercial device release, the HTC Dream, in September 2008. Android was created essentially for touchscreen mobile devices [7]. Android has many features such as media support, messaging, multitouch, multitasking, web browsing, and many more. The most important feature of Android allows us to freely modify, invent and implement our device drivers and attribute. That has become its most significant concern also.

Authorized licensed use limited to: Biruni Universitesi. Downloaded on November 13,2022 at 18:13:02 UTC from IEEE Xplore. Restrictions apply.

The Android operating system design contains different numbers of elements from an open-source Linux Kernel to a group of different C/C++ libraries, which are uncovered through application framework services. The main functionality of operating system functions is present by the Linux Kernel and the stage for running the android application is presented by Dalvik Virtual Machine. Major elements of the Android design can be seen in the different software layers in figure 1.

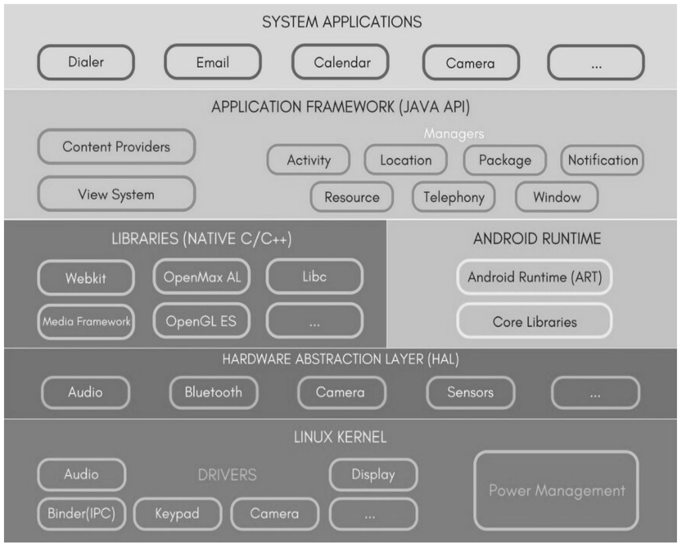

Figure 1. Android Operating System Architecture

Applications - The place where all the applications are installed and run is on this top layer. Inter-Process Communication (IPC) Endpoints help this layer to communicate with other layers.
Application Framework - Basic functionalities such as location management, voice call management, and resource management of the Android device are provided on this layer since this layer interacts with android applications directly. Android application developers can use these services in their applications.
Libraries - These are core Java libraries built specifically for the Android operating system to provide support for Android application development.
Core Java Libraries - These libraries provide the functionality defined in Java SE libraries.
Dalvik Virtual Machine - This runs on a device having its own operating system like a Java Virtual Machine it is a register-based virtual machine. Virtual machine features such as simultaneously, providing isolation, security, memory management, and threading support are provided by this layer
Android Runtime - Core Java Libraries and Dalvik Virtual Machine are included in this. It helps Android application programmers use the Java programming language for developing applications.
Linux kernel - Basic system functions such as process management, memory management, device management, and device driver are provided on this layer.

Before we detect malware we should know how malware can be spread in a system [8].
Repacking - Official mobile applications are repackaged with malicious application and it is distributed on third-party application stores.
Drive-by Download - In this technique, the malicious application is downloaded into the device when the user visits a malicious website.
Dynamic Payload - Malware application embedded onto an encrypted source in an application when installed it decrypts the malicious payload.
Stealth Malware Technique - This technique is referred to exploit hardware vulnerabilities to obfuscate malicious code.
Malware detection techniques can be categorized into two categories that are Statics and Dynamic techniques as shown in the Figure 2.

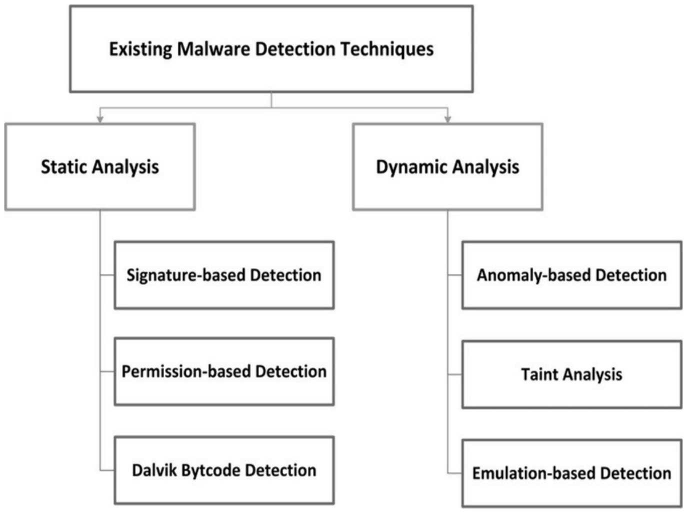
Existing Malware Detection Techniques
Static Analysis
Dynamic Analysis
Signature-based Detection
Permission-based Detection
Dalvik Bytcode Detection
Anomaly-based Detection
Taint Analysis
Emulation-based Detection

Figure 2. Existing Malware Detection Techniques

Static techniques - In this technique, the application is analyzed without actually executing it. The application is reverse-engineered by extracting the files from the application which is loaded into a disassembler. Then we analyze it for further understanding of how the application is functioning and what it is capable of doing.It can be further classified into the following classes

- **Signature Based Detection -** In this method, a unique digital signature is extracted from the application and compared against the existing signature database if the signature matches with the existing signature, the application is classified as malware.
- **Permission-Based Detection -** In this method, the application is overseen based on access rights. Any permission such as user data or system security is not available to applications by default. Usually, users themselves give access to the required application when the application is first installed. Analyzing permission can be an effective way since we don’t need to analyze the code of the whole application.

Authorized licensed use limited to: Biruni Universitesi. Downloaded on November 13,2022 at 18:13:02 UTC from IEEE Xplore. Restrictions apply.

- **Dalvik Bytecode Detection -** In this method, applications are converted into java bytecode. Then the bytecode is analyzed in a virtual machine. App behavior, control, and data flow are analyzed with the help of bytecode for detecting malicious functions in the application.

Dynamic techniques - In this technique, the application is analyzed during its execution. It can be further classified into the following classes

- **Anomaly Based Detection -** In this method, the application's behavior is monitored and all the changes in the system are recorded. To classify an application as malware, various features such as battery level, CPU usage, network traffic, etc are monitored. When all the data are recorded then it is passed to an algorithm for the classifier which is based on heuristics or rules.
- **Taint Analysis -** In this method, we use Taindroid which helps us identify data leaking and track multiple sources of delicate data in an application. All the recorded data are labeled by the tool.
- **Emulation-Based Detection -** In this method, we use different dynamic analysis tools such as DroidScope which helps us dynamically analyze the android application based on Virtual Machine Introspection. In this way, the malware is unable to detect any anti-malware services installed on the system and it executes as intended but we are still able to monitor the whole system. There are more tools available that can help us achieve the same goal.

III. Related Work

Android's popularity is the reason behind the enormous growth of malware development for the operating system. That has motivated researchers to look for resolutions to identify and prevent dangerous applications from getting injected into the operating system.

Joany [9] aimed to explore the development of Android spyware using reverse engineering techniques in which he found two security flaws the first, the Android Market where no specific control is applied to applications submitted by developers, and the Android security chain is the slow patching process. The limitation of this paper is that since the static analysis was done to review the permissions, it can become a time taking process and automated tools can provide false-positive results because they are only as good as the rules defined.

Rajinder [10] discussed the different features and architecture of the Android operating system, as well security features such as Dalvik VM and the signing mechanism provided by the Android operating system. The disadvantage of this paper is that the results are not precise anymore, since modern malware can be able to bypass security features provided by the operating system.

In [11] they talked about the ways to tell whether a user might be infected with spyware applications, the limitations of anti-spyware, and software controls that may become necessary for the operation system. They also explained some ways to prevent information leakage to spyware applications. They were able to only give the general idea about the spyware application based on pre-compiled data from different sources but didn't propose any work to overcome it.

Wei-Ling Chang and other [12] offered a program in Android sandbox to ascertain whether the unknown application is a malicious application or not, and it shows its Confidence value, which is the class probability distribution for an unknown APK. They performed their research on a very old dataset that was based on permissions, which can give false-positive results because not every reported permission in the details of the application is published by the application market or the developer of the application has disclosed it and the permission are necessarily the required permission of the application.

Authors of [13] discussed the behavior of different android malware and threats by addressing different types of malware with the help of static and dynamic analysis. The disadvantage of this paper is that both static and dynamic analysis can give false-positive results based on the rules. Furthermore, the dynamic analysis also has limitations due to limited access to the network.

Belal [14] studied and analyzed various malware detection techniques and gave the idea of a framework for developing a user profiling-based mobile malware detection tool. The limitation of this paper is that the techniques provided were not efficient enough compared to machine learning-based detection. And both static and dynamic analyses have limitations as mentioned before.

In [15] with the help of machine learning techniques, they developed a framework for analyzing Android applications to examine whether the application is malicious or not. The limitation is that performance evaluation and feature reduction of the data set can be optimized. Moreover, the dataset that was used was out-of-date.

In [16] they accomplished 84.14% precision using machine learning based on the feature generated from apk samples with the help of malicious and benign apk files. The technique they used for malware samples to generate images was not efficient as they were not able to extract the images from all the applications, or they didn't have classes.dex file. Their result was much less in all the algorithms they worked on.

J. D. Koli [17] offered the android malware detection system, RanDroid, help of permissions, APIs, and also the presence of various key apps data such as the dynamic code, reaction code, native code, cryptographic code, database, etc. to distinguish malicious applications from legitimate ones. The suggested method is constructed on the notion of static analysis and lacks dynamic inspection. The outcome may also diversify with the increasing size of the training and testing data sample set.

In [18] they researched large-scale fake applications. They were the first to introduce the concept of fake apps. And gave three viewpoints, namely fake sample characteristics, quantitative analysis, and fake authors. A technique they used can be classified as signature-based detection, and there is a limitation of signature-based detection that they cannot

Authorized licensed use limited to: Biruni Universitesi. Downloaded on November 13,2022 at 18:13:02 UTC from IEEE Xplore. Restrictions apply.

detect unknown or new types of attacks. Since signature-based detection is a static technique, it can also be circumvented by code obfuscation.

IV. Recommended System

The proposed solution for detecting and classifying benign or malware applications is discussed in this section.

We have discussed, Android operating system, how various malware detection techniques work and based on the survey we concluded that all the techniques provided used an out-of-date dataset further, the outcome of machine learning results can be improved. The solution is to use a most recent dataset with more pertinent features to enhance the classification outcomes and provide effective prediction results for the malicious applications.

The suggested solution consists of the following states

A. Creating Malware

To show a modern malware application, we will be creating our malware android application.

There exist several programming languages such as Kotlin, Java, and C++ which can be applied to code an Android application [19]. The contents that an Android app requires at runtime are included in an archive package with .apk extension, these packages are used in all the android devices to install apps. This file basically contains app code, resources, and meta information.

Since Android malware is no different from an Android application, so we can use the same procedure to develop malware, with the major difference being the malware application will be designed to harm a user's device.

The malware that we created can collect all the sensitive user data with the help of android system permissions and API calls in the Android operating system. The malware will be hidden from the user because it is embedded in another application.

B. Injecting Malware

As discussed earlier, there are various ways of injecting malware applications into an Android device or emulator.

We will be using spam email where the malware application that we created will be attached to an email and that email will be sent to the victim user on their email. Once the victim user has downloaded and installed the malware application and tries to open the installed application, the malware will be successfully injected into their system, giving the attacker access to the victim's Android device.

C. Collecting Data

Once we have established a connection between the victim's Android device and the remote system we have gained complete access to the victim's Android device meaning we can collect any sensitive data from the victim's Android device.

Since our malware was designed with all the android system permissions and API calls in the Android system, we can collect the victim's applications installed, files, contacts, messages, calls, accounts, network information, and other personal information.

D. Detecting Malware

This section will be divided into two parts dataset and machine learning algorithm. The detection model is shown in Figure 3.

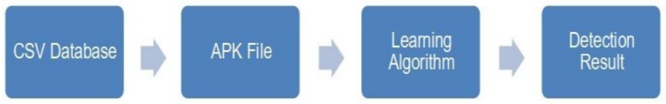

Figure 3. Detection Model

- **Dataset**
  We required a dataset that is reliable and comprehensive to help with our machine learning model and will be able to validate the android malware.
  Hence, we will be using CIC-InvesAndMal2019 [20] which contains the combination of 426 malware samples of android applications and benign samples, and it is classified into four categorizations namely Adware, Ransomware, Scareware, SMS Malware. This dataset accommodates various features such as battery states, log states, packages, process logs, etc.
- **Machine Learning**
  Random Forest is a supervised learning technique in the machine learning algorithm [21]. We can use this algorithm to resolve Classification and Regression problems. Ensemble learning concepts are utilized in this. Figure 4 illustrates the working of the random forest algorithm.

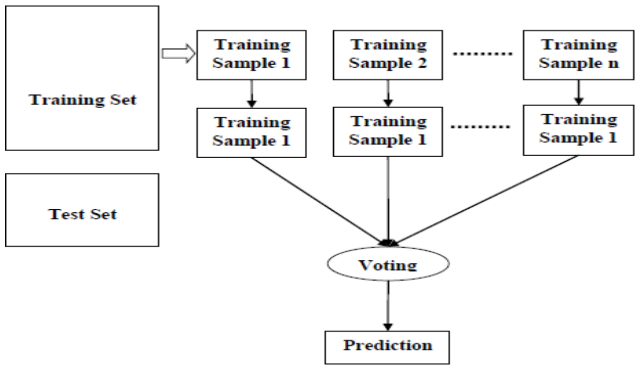

Figure 4. Random Forest Algorithm

From the given figure 4 we can see that initially when a dataset is supplied to the algorithm, random samples will be picked from the supplied dataset. Every sample will be used by the algorithm to make decision tree. Once the trees are constructed, prediction results will be obtained from every decision tree. For every predicted result, voting will be carried out. Finally, the prediction having the maximum number of votes will be considered as the final result

Similarly, we will pass the CIC-InvesAndMal2019 dataset along with the malware application we created to get the final prediction result. The accuracy rate is expected to have more than 95%.

Authorized licensed use limited to: Biruni Universitesi. Downloaded on November 13,2022 at 18:13:02 UTC from IEEE Xplore. Restrictions apply.

CONCLUSION

In this paper, we saw what is Android malware, and how mobile development has evolved in recent years. We also got an understanding of the android operating system, and what are the different types of malware spreading and detection techniques.

The main aim of the paper was to demonstrate the working of an Android malware application and to propose a detection model which is more efficient compared to previous models. For that, we created an Android malware application that was injected into an Android emulator using spam email then explained why the data sensitive data can be collected, then we used the CIC-InvesAndMal2019 dataset along with the random forest machine-learning algorithm to provide an efficient malware detection model.

Furthermore, in the future work will focus on enhancing the prediction result outcome of the given model.

REFERENCES

[1] Statcounter, ”Mobile Operating System Market Share Worldwide”, https://gs.statcounter.com [Online] Available: https://gs.statcounter.com/os-market-share/mobile/worldwide

[2] Statista, ”Number of available applications in the Google Play Store from December 2009 to September 2021”, https://www.statista.com/ [Online] Available: https://www.statista.com/statistics/266210/number-of-available-applications-in-the-google-play-store/

[3] Statista, ”Growth of available mobile apps in the Google Play Store worldwide from 2nd quarter 2015 to 1st quarter 2021”, https://www.statista.com/ [Online] Available: https://www.statista.com/statistics/185729/google-play-quarterly-growth-of-available-apps/

[4] Statista, ”Growth of available mobile apps in the Google Play Store worldwide from 2nd quarter 2015 to 1st quarter 2021”, https://www.statista.com/ [Online] Available: https://www.statista.com/statistics/185729/google-play-quarterly-growth-of-available-apps/

[5] Mcafee, ”What Is Malware?”, https://www.mcafee.com/ [Online] Avail- able: https://www.mcafee.com/en-in/antivirus/malware.html

[6] Comodo CyberSecurity, ”What is Malware Detection - is it important?”, https://enterprise.comodo.com/ [Online] Available: https://enterprise.comodo.com/what-is-malware-detection.php

[7] Wikipedia, ”Android (operating system)”, https://en.wikipedia.org [On- line]

[8] Belal Amro,” Malware Detection Techniques For Mobile Devices”,International Journal of Mobile Network Communications & Telematics(IJMNCT) Vol.7, No.4/5/6, December 2017.

[9] Joany Boutet ,”Malicious Android Applications: Risks and Exploitation”, SANS Institute Information Security Reading Room paper, The SANS Institute 2010.

[10] Rajinder Sing int. journal of Engineering Research and Applications Vol.4 (Version1) February 2014, pp.519-521. An overview of Android operating system and its security features.

[11] Mahesh v, and Dr. Sumatra Devi K.A., ” Analysis of Prediction and Detection of Spyware for User Applications”, Roy et al, ISSN 2349-3917, AJCSIT,2016

[12] Chang,Wei-Ling, Hung-Min Sun,and Wei Wu.”An Android Behaviour-Based Malware Detection Method using Machine Learning.” 2016 IEEE International Conference on Signal Processing, Communications and Computing (ICSPCC).IEEE, 2016.

[13] Lokesh Vaishanav1, Shanu Chauhan1, and others, ” Behavioural Analysis of Android Malware and Detection”, International Journal of Computer Trends and Technology (IJCTT) – Volume 47 Number 3 May 2017.

[14] Belal Amro,” Malware Detection Techniques For Mobile Devices”, International Journal of Mobile Network Communications & Telematics (IJMNCT) Vol.7, No.4/5/6, December 2017.

[15] Vrama,P.Ravi Kiran,Kotari Prudvi raj,and KV Subba Raju.”Android mobile security by detecting and classification of malware based on permissions using machine learning algorithms.”2017 InternationalConference on I-SMAC (IoT in Social,Mobile,Analytics and Cloud)(I-SMAC).IEEE2017

[16] Darus,Fauzi Mohd,Salleh Noor Azurati Ahmad,and Aswami Fadillah Mohd Ariffin.”Android Malware Detection Using Machine Learning on Image Patterns”2018 Cyber Resilience Conference(CRC).IEEE,2018.

[17] Koli, J. D. “RanDroid: Android malware detection using random machine learning classifiers.” 2018 Technologies for Smart-City Energy Security and Power (ICSESP).IEEE,2018.

[18] Chongbin Tang, Sen Chen, and others. ”Large-Scale Empirical Study on Industrial Fake Apps”, arxiv:1902.00647v2[cs.CR] 10Feb2019.

[19] Tutorialspoint, ”Android-Overview”, https://www.tutorialspoint.com [Online] Available: https://www.tutorialspoint.com/android/android_over view.htm

[20] Laya Taheri, Andi Fitriah Abdulkadir, Arash Habibi Lashkari; Extensible Android Malware Detection and Family Classification Using Network- Flows and API-Calls, The IEEE (53rd) International Carnahan Confer- ence on Security Technology, India, 2019

[21] Javatpoint, ”Random Forest Algorithm”, https://www.javatpoint.com/ [Online] Available: https://www.javatpoint.com/machine-learning-random-forest-algorithm

Authorized licensed use limited to: Biruni Universitesi. Downloaded on November 13,2022 at 18:13:02 UTC from IEEE Xplore. Restrictions apply.