International Journal of Applied Engineering Research ISSN 0973-4562 Volume 13, Number 7 (2018) pp. 4753-4763

Machine Learning Techniques for Malware Detection on Android Mobile Platform

[1]Dr. S.Sivasathya and [2] Kush Manohar

[1]Associate Professor, Department of Computer Science, Pondicherry University, Pudducherry-605014, India.
[2] Department of Computer Science, Pondicherry University, Pudducherry-605014, India.

Abstract

Nowadays android has become the most popular mobile operating system for mobile users. It is an open source mobile operating system with good user interface that provide various applications making the user smarter. Review made in the year 2017 explicitly states that more than 2.0 billion gadgets which can be summed up to 87.5 %, contributes to the android version in the global market of mobile operating systems. As the numbers do the talking, 2 billion people every month are found active on Android devices according to Google report. Hence, the problems regarding malware is soon to be expected and it has introduced itself in due time. Android threats are increasing everyday that prompts digital security threat. Malware apps are commonly distributed in markets that are operated by third parties. Even Google can't give assurance that all of the listed apps are eligible for safe and secure use. In the global arena, many anti-viruses are available for portable devices but most of them are not capable to identify new and advance malicious threats. Today, a number of analytical techniques have been developed, but precision of detecting new android malware apps are still under supervision and remains to be a questionable issue. This paper is directed to focus on a systematic study of the design overview of android platform, applications, malwares and the techniques used for the detection of android malware. We aim to illustrate and summarize the techniques of static and dynamic code analysis for Android and some existing work in this field, using machine learning perspective.

Keywords: Android threads, Machine Learning, Ransomware , Banking-Trojans, Advance thread.

INTRODUCTION

Smartphone has become a crucial part of the human race as mobile devices conclusively encompass our interests and needs. The speedy growth in Android OS based mobile devices and amazing advances in technologies today, like 4G/5G has lead to a tough competition with the systems such as BlackBerry and iOS, etc. But amongst these mobile OS, Android has turned into a deserved option as it seeks to provide portability and trouble-free access to the user anywhere, anytime. In this digitalized era, smartphones play a key role. Used as a portable, comprehending device for online shopping, watching movies, internet banking, filling forms, for all kinds of bill payments, online order etc., these devices thrive to control the heavy handedness of mundane tasks in life. But apart from its 'happy-go-lucky' usage, most people are unaware of the mobile threats and how to use it in a secure way along with its technical limitations.

Deficiency of knowledge in this regard leads to serious types of threats like sms leak, personal information leak, password stealth, Phishing, Banking-Trojans, Spyware, Bots, Data stealing, Root Exploits, ransom ware, Premium Dialers and Fake Installers. These types of troubles are faced by people when they download apps and data without knowledge about which one is a malicious app or a good one. These malicious apps or programs harm the device upon its arrival. There are anti viruses available to increase the security level of android devices and prevent any kind of safety breach. But sometimes, rogue software's can replicate sites or apps. One would normally trust them- being tricked and giving into the temptations, eventually compromising password or card details which will simply be sent to remote users of mobile devices, which is annoying but not exactly hazardous. But for last few years, Android Smartphones have been progressively targeted more than usual by cyber attackers and being infected with different types of malevolent apps. Once installed, they send one's vital information in the background, costing one's money [1]. So if one installs third-party apps outside of Google play store, it will be unsafe. Also, in May (2017), ransom ware attacked almost all devices in the world. So, one needs an extra shield from these type of malware. Today, Machine learning technique is more renowned for malware investigation. In this technique, features are extracted from manifest.xml (definition file) and .dex (code based) files using reverse engineering. These techniques extract features (e.g., permission based, API based, etc.) from known Android benign application and malware application, then use different machine learning algorithms (e.g., SVM,NN, decision tree, etc.) to learn these extracted features in order to distinguish between benign apps and malware apps. This technique is fully automated and allows the static and dynamic detection of android malware application.

OVERVIEW OF ANDROID OPERATING SYSTEM

Android OS is one of the most frequently used operating system .Android OS is developed on top of Linux kernel because of its robust driver model, networking support, efficient memory management and process management for the its core services.

Android operating system is separated into four major layers, the kernel, libraries, an application framework, and applications. Android app developers write code for apps in Java language and control their operation using Java Libraries.

Java Libraries are designed by Google. Android software stack has a number of diverse elements, shown in Figure-1[2].

Android Software stack is a compilation of Linux kernel and libraries of C/C++ that are uncovered through the app framework. This android app framework provides special services and management at specific runtime. All function and Core services are monitored by Linux kernel. It also provides an abstraction link between device hardware and the remaining of the stack. Libraries include a number of dissimilar C/C++ core libraries such as SSL and libc. Android run time includes Dalvik virtual machine and core libraries. There are many classes available for creating android apps and these classes are provided by App Framework. Android apps are built on app layer. Both third party and native apps use the identical type of API libraries.

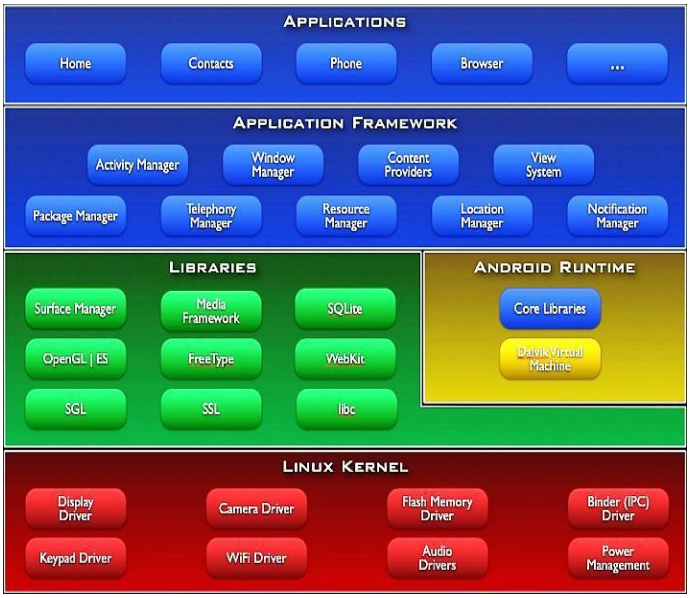

Figure 1. Andriod operating system structure

ARCHITECTURE OF ANDROID APPS

In this segment, the idea is to give the outline of the structure of android apps in closest proximity possible. Android apps come as Android Package files (APK). The android APK files are packed in ZIP files .This compact file is a synchronized grouping of the apps and their bytecodes along with the essential data, intermediary libraries, resources, and a manifest file that are required for apps to work. Figure 2. [3] shows java source code and how it's translated into APK file in a step by step manner . The building processes of apps are illustrated as below.

Resource code creation - Android Asset Packaging Tool (AAPT) takes the application resource files, compiles it and produces R.java.

Interface code generation – This tool converts .aidl interfaces into Java interfaces.

Java compilation – It compiles the entire Java native code, as well as R.java and .aidl files and converts into .class files.

Bytecode conversion - The dex tool converts the .class files to Dalvik bytecode and packages into the ultimate .apk file.

Packaging – It packages the entire .dex file and compresses resources into a .apk file.

Signing the package- Before installation in devices android apps needs to be digitally signed with a certificate. This certificate is usesed for identifying the author of an app . It uses self-signed certificates. Developer holds the certificate's private key. One the apk file signed ,it is ready to the used.

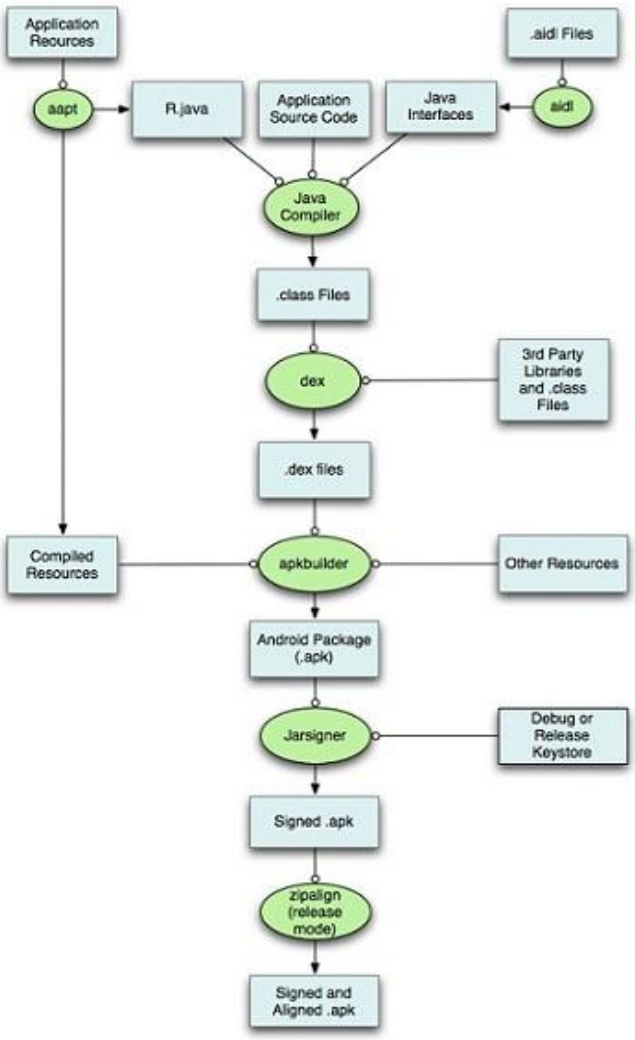

Figure 2. Android application builds process

STRUCTURE OF ANDROID APPS

Android application files (APK) written in Java programming language is a package that contains along with data , many resource files in a single container as shown in the following Fig-3.

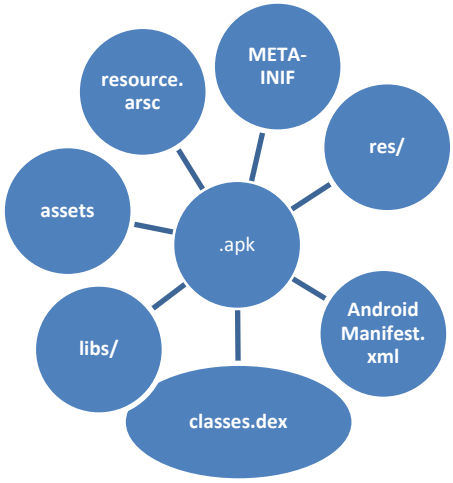

Figure 3. Structure of .APK file

resource. arsc: It contains the binary XML resource file after compilation.

META-INIF: It contains Meta file or explanation from Java jar file.

res/: It contains the resource document.

AndroidManifest.xml: It contains the configuration file about the authorization and services.

classes.dex: Contains the Dalvik byte-code or Android execution file.

libs/: It is a optional folder that contains Android Native Development (NDK).

assets: It contains the raw file that are needed for an application like music, font, image, etc.

ANDROID MALWARE

A malware is a piece of program designed for crooked orientation and to perform the function of 'stealth' or stealing user's personal information, email credentials, Internet banking information , etc. Malware that affects the android mobile or android based device are termed as android malwares. This section provides a description of the mobile malware that exist to perform malicious activities and create future threats. According to a leading computer security company G Data, a latest Android malware is exposed every 15 seconds. The company also predicts, that it will produce 3,500,000 malicious Android apps during 2017 as shown in fig-4 [4].It is also seen that the occurrence of Android malware is due to many users using older versions of the android OS which is easier to target.

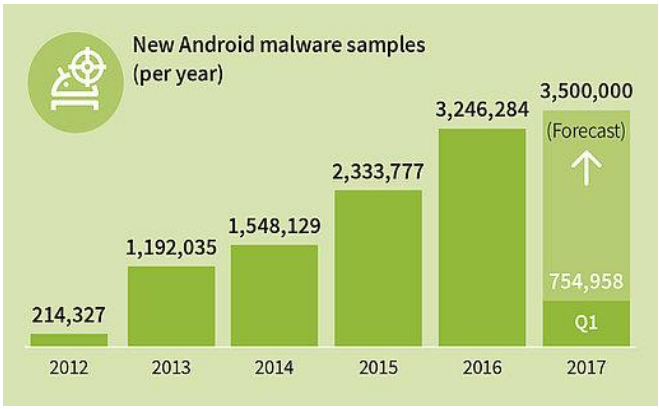

Figure 4. Latest Android malware growth

It can be seen in the given Figure-5 that there are 30.1% of Android users still working with older edition of an OS from 2014 , below 0.1% user are not shown in this figure. This is the main reason to target the android OS.

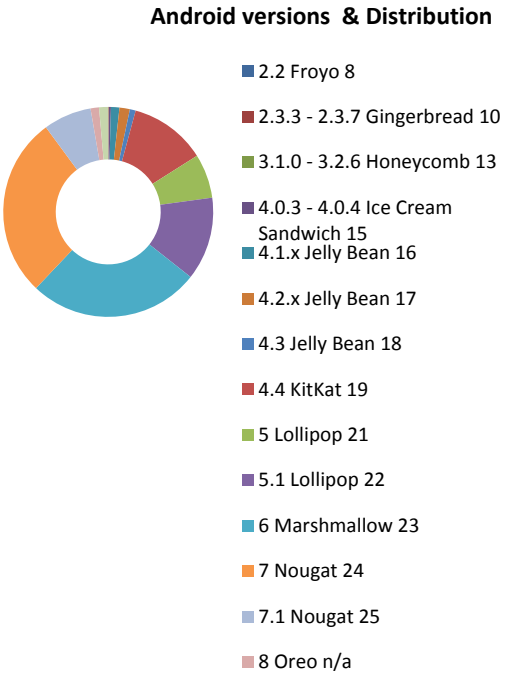

Figure 5. Different Android versions and market distribution

A. Popular Types of Mobile Malware

This section discusses some of popular mobile malware and also classified them based on functional characteristics. Also it discusses some popular malware specific to android os.

Banking Malware: This malware exploits user's phone and performs stealth on private information. It looks identical to financial apps, hence it becomes easy to fake its real face and

deceive people with ease, in accessing services. Nowadays banking-based malware is on the go , These apps are available on Google play store. Even Google play store traces the hostile apps and tries to be efficient in removing these types of apps but until recently, backbot apps were not found. Hacker keeps those users' under observation who like it better to conduct all of their banking work online as well as transfers and payments from their smart phones. Banking malwares are one of the fastest increasing malware.

Mobile Ransom ware: - It was first made popular on PCs, but mobile ransom ware threats are increasing since last 3 years. It spreads through third party apps, games and also as system update. Once it enters the system, it locks the user device, encrypts the files and lock the files. Cyber criminal demands a ransom which users have to pay. If user doesn't pay the demanded ransom they can delete the document, lock the device permanently. Mobile ransom ware increased in 2017 by 250%. Ransomware is one of the most dangerous threats faced today.

Bot: - It is a type of malware that grants its dispatcher or author or bot master to access the system remotely and also control remotely. It carries out spyware activities on the infected system.

Mobile Spyware:-This type of malware is loaded as a form of program into devices. Once it enters the device, the spyware monitors all the activities, location and steals key information such as usernames and passwords of user accounts, credit card details and other credential information. In certain cases it has been observed that spyware is a package that runs with other benign application and collects credential data in the background and sends back. But the user may not still realize the existence of any spyware until device performance degrades to a considerable level.

MMS Malware: Malware makers are too intelligent. They always search for ways to take advantage of the devices. They yearn to exploit devices through text-based communication as a way to distribute malware. The vast uses of Android phones can be hacked by distribution to the user a specially made multimedia message (MMS) to the users. The weakness in Android devices like media library, Stage device, makes it comfortable for attackers to send an embedded text message with malware to any user mobile number. Even if users doesn't open or acknowledge the text, the malware could still install in the device, allowing hackers the root admittance to Smartphone device.

Mobile Adware: Adware has become the universal app based Smartphone threat around the world. Mobile ads act as an important part of today business but behind the sense these ads hold malware that can place user privacy at risk by capturing personal information without user authorization. Even though most ads are safe, 6.5 percent of free apps in Google Play store enclose adware.

SMS Trojans: SMS Trojans was founded since the time when mobile devices have turned into user interface true to its word. When malicious code is being installed in mobiles and Trojan malware infectivity occurs user can't realize about activity. SMS Trojans vary from good apps by looking like the good app with a fake app name, and with fake download links.

B. Functional Classifications of Mobile Malware

Mobile Device Data Stealers –This type of malware application steals private information such as contacts, phone logs, browsing history, SMS and GPS data, etc.

Rooting Capable Malware- Generally this type of malware controls a device by obtaining root access. Once this type of malware gets root admittance to devices it becomes very difficult to remove it from mobile.

Premium Service Abuser– These malware apps take advantage of secret and Dial premium services or send messages that could cost the device owner. It is so stealthy as the device owners don't identify that his gadget is sending out messages to the premium numbers or doing other activities in the background of the devices that could cost the device owner.

C. Popular Android Malware

Generally, four types of the most widespread malevolent programs those are well-known to affect the Android mobile devices and operating systems are available.

Worm: This is a stand-alone type of malware with no ending. Replicates itself constantly in the device and extends to other device. Worm apps make identical copies of itself .It spreads through text, internet, messages, SMS, MMS or removable media.

Spyware: Generally, this category of malware application poses a threat to mobile devices by diffusioning into a user's private or some vital information without the user's knowledge. Usually, this kind of malware utility poses an opportunity to cellular devices via spreading a user's private or some crucial data without the user's knowledge.

Trojan: It typically enters into devices in a different look like attractive and non-malicious app. Once it enters the devices it executes a spy or steals user information and also performs some activities like deleting data, modifying and blocking data. This type of malware application constantly requires user contact for activation. Once it gets activated, it forms the foundation of a serious damage to the applications or the device itself. It synchronizes with the user's accounts and any supplementary source of information like password and other secret information are sent to a server. It comes in many variants like Backdoor Trojan, Downloader Trojan, Information stealer Trojan, Remote Access and DDoS attack Trojan.

Ghost Push: Generally, this type of malwares infects the Android OS by gaining root admittance to the Android operating system. After entering inside the android operating system it converts into the system app and loose root access which makes it virtually impossible to remove by the user.

MALWARE DETECTION USING MACHINE LEARNING TECHNIQUES

The present mobile communication and digital system infrastructures are extremely vulnerable to a variety of attacks which can cause severe damage to mobile users. The current expansion of high-speed Internet digital infrastructure has led to the increase in the produce of new mobile malware.

Although maximum existing anti-malware systems are only capable of detecting known signature malware, they cannot show an equal action against new and unknown malware application which has strange signatures. Only few android anti-malware systems are based on machine learning algorithms. Generally, Machine learning algorithms provide an efficient way to deal with this kind of problem. It provides the ability to systems that learns from past experience or through experience without being explicitly programmed [34]. The basic concept of machine learning algorithm is to train the model, based on some algorithm like Naive Bayes, genetic algorithm, neural network, SVM, MLP , Apriori and so on to perform a particular task like regression, classification,etc. Fig-6 gives an overview of the framework of android malware detection or classification model using machine learning algorithm.

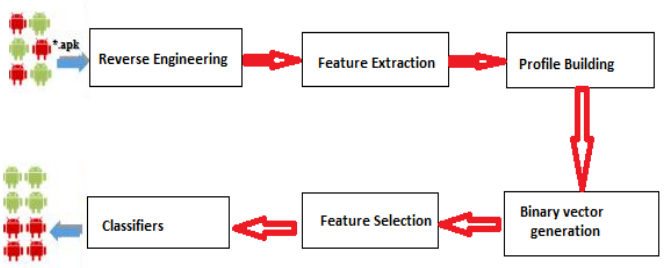

Figure 6. The framework of machine learning model

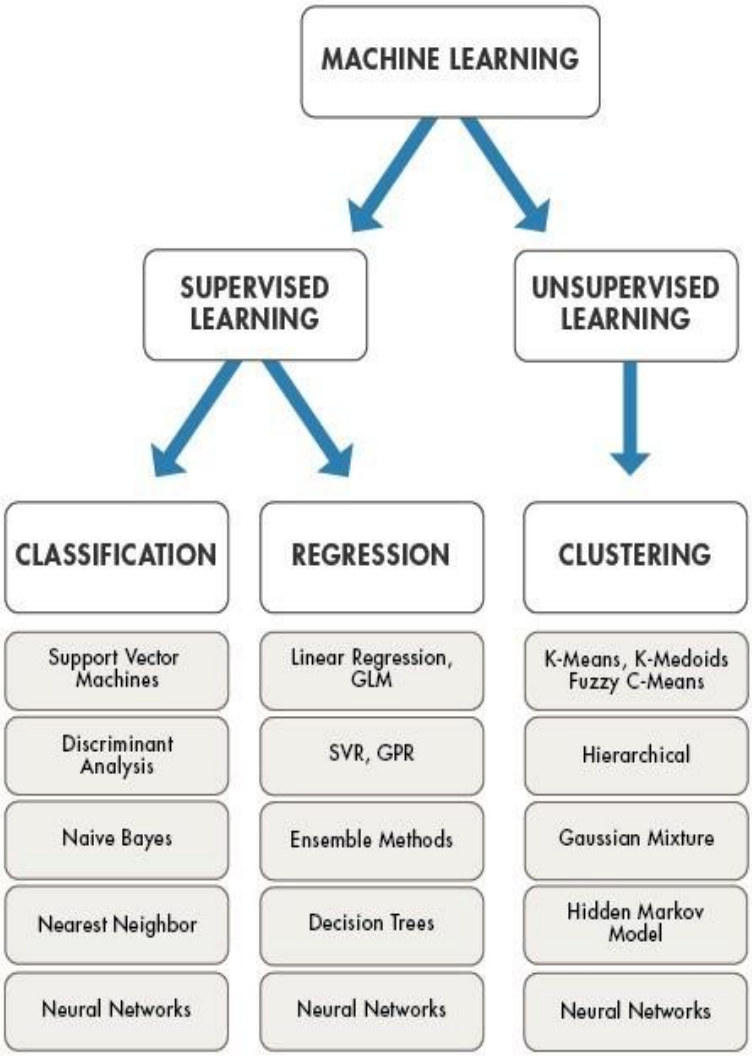

Figure 7. Different types machine learning algorithm

ANDROID MALWARE ANALYSIS MECHANISMS AND FEATURE EXTRACTION.

There are two methods of security management for android mobile.

- *Network based approach.*
- *Host based approach*

A. Network based approach.

A network based approach monitors network traffic for any suspicious, abnormal or unauthorized exploit which could lead to a cyber attack. This method is based on "acquired knowledge". Method used to identify these activities is either Behavior based (Anomaly Detection) or Signatures based. [6]. Only few researches have worked in the field of android Network based malware detection using machine learning.

Table 1. Research On Network based feature Android Malware Detection

[Reference No.,] Author`s, Name [Year	Technique used	Approach	Detection Method
[7] , Feizollah et al. [2013]	Naïve Bayes, K-nearest Neighbor, Decision Tree, Multi-Layer Perception and Support Vector Machine (SVM	NIDS	Network traffic connection length, TCP length, GET/POST request.
[8] Narudin et al. [2014]	Bays network classifier, Random forest classifier	NIDS	TCP/TP packet

Feizollah et al. [7] proposed a method for Android Mobile Botnet Detection using machine learning. In this study, three features were extracted from the network traffic i.e. connection length, TCP length, and GET/post request. Five classifiers have been used in this study. Naïve bayes, nearest Neighbor, decision Tree, Multi layer belief perception and Support Vector Machine (SVM). In this proposed method, total 100 malware apps that belong to 10 malware families and normal apps from Google apps were used for feature extraction. K-fold method was used for validation. The author claimed a true positive rate (TPR) of 99.94% and FPR (False Positive Rate) of 0.06%.

Narudin et al. [8] have associated anomaly based feature detection model for Android malware detection.Network traffic based feature selected for this model, which is based on networks information like basic connection, time and content. For feature selection work, the author first filtered the TCP/IP traffic then extracted data from that TCP/IP packet. Wireshark tool is used for traffic generation and TCP/IP features extraction. After feature selection labeling the selected feature was done. In this model 11 network features are used. Bayes network classifiers are and random forest classifiers used for classification. Here two types dataset are used. MalGenome dataset and private data set. K-cross fold method was used for validation. The proposed method gave 74.15% accuracy with random forest classifier.

B. Host based approach

In host based approach, a system that monitors is installed on a particular mobile for identifying malware apps, malicious action and alerts to the user. There are two approaches used in host-based machine learning method for malware detection.

- **Static approach**
- **Dynamic approach**

A. Static approach

Most of the antivirus companies used static approaches to detect malware.

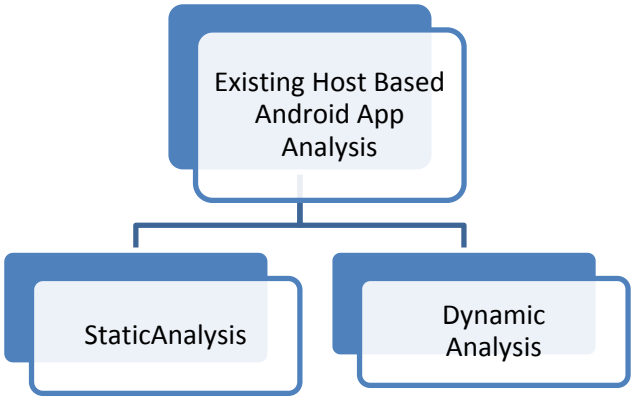

Figure 8. Different approaches of analysis technique.

In this approach the android apps have to be decompiled to extract the static feature. After decompilation the apps are converted into Manifest.xml, classes.dex and these files keep lot of information regarding the app that are useful for static analysis like API call, permission-based, component-based etc. Also it keeps the information of the packages that are used in an application. Static analysis does not need to run the application on emulator.

There are many tools available for reverse engineering and static feature extraction of Android apps Eg- Androguard ApkTool, dex2jar tools, Monkey and Android View Client etc. Dynamic code loading and code obfuscation have a big disadvantage of this method but the advantage of this method

is the low-cost computation, less time and low resource utilization.

Table 3. List of static analysis and their feature.

Static Analysis	Features
API	sensitive APIs calls (Activities, services, etc.)
Code Feature	native code, java reflection, dynamic loader
Component Analysis	Components, broadcast receivers
Permission	personalized permissions
Category Analysis	categorization the APK
Basic Information	file hash, file size, package name, etc.

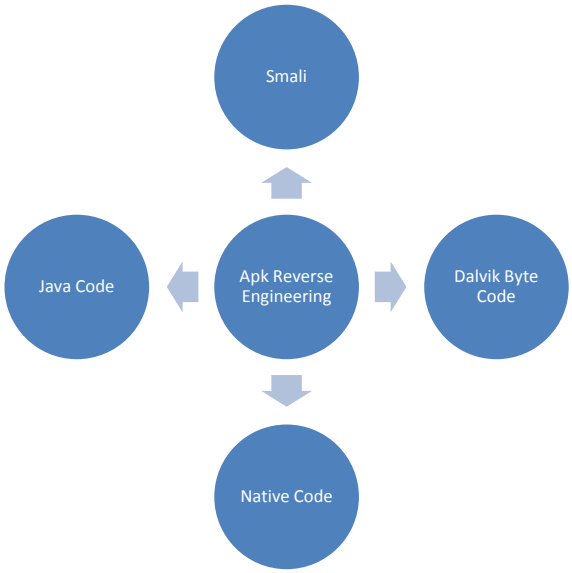

Figure 9. Decompile process.

```xml
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.greatfeat.greatfeat">

    <uses-permission android:name="android.permission.INTERNET"/>

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity android:name=".startup.LaunchScreenActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
```

Figure 10. Manifest.xml with declared components.

Table 4. Code and their feature.

Manifest.xml	. dex code
Hardware based components	API calls
Application components	Permission used
Requested permissions	Network addresses
Filtered intents	Suspicious API calls

Most of the work done in the area of Android malware detection using machine learning Techniques is based on static features. Only a few of researches have focused on the dynamic feature based technique for android device malware detection.

Table 5. Research on Static Feature Based Android Malware Detection System

[Reference No.,] Author`s, Name [Year	Technique used	Approach	Detection Method
[9] D.Arp et al., [2014]	Support Vector Machine (SVM)	HIDS	Permission and API call.
[10]Xin Su, et al [2016],	Deep brief netwok(DBN) with SVM	HIDS	Permission and API call.
[11]Du et al.[2015]	of SVM, Decision, BayesNet Tree (J48)	HIDS	Permission based
[12] Sanz Borja,et al[2013],	Random forest	HIDS	Permission based
[13] Aung et.al [2014]	SVM	HIDS	Permission based
[14] Aafer et al. [2013]	KNN model	HIDS	API-based
[15]Wu et al. [2012]	Clustering algorithms. Bayesian classification Detection rate	HIDS	manifest properties and API calls
[16] Sahs et al. [2012]	One-Class SVM classification algorithm	HIDS	Permission and control flow graph
[17]Ghorbanzadeh et al[2013]	Neural network(NN)	HIDS	Permission based
[18]Sezer et al.[2015]	Bayesian classifer	HIDS	Permission and API call.
[19] Zhou et al.[2012]	fuzzy hashing technique	HIDS	Dalvik opcodes

D.Arp et al. [9] presented a broad static feature-based machine learning model. In this model 123,453 good applications and 5,560 malware application samples were used from Malware Genome dataset. All malicious apps were verified with Virus Total service and antivirus scanner.SVM model is used for classification of apps. In this model, a feature was based on permissions and API call. Extracted features are stored in a binary vector. For the evaluation purpose the author divided the data into two-parts of 66% and 33% for training and testing purpose respectively. This model was able to detect approx 94% of the malware samples.

Xin Su, et al. [10], presented a static feature based malware detection system .This model is based on deep learning model for malware classification. The author used apktool for features extraction.A total of 32247 features were extracted from android apps. Deep brief model (DBN) was used for learning features from feature vector and SVM was used for classification purpose. For the experiment purpose, 3,500 benign apps and 3,500 malware apps were used. This model achieves an outstanding runtime effectiveness which made it very easy to a larger scale of real-world Android malware detection. The average accuracy under different number of neurons is higher than 96%.

Du et al. [11] presented his idea that focused on a static analysis. This research is based on the static features for malware recognition. Static features were extracted individually from apps. The Dempster Shafer algorithm was used for collating these features into a single vector. Here the author used different classification algorithms like SVM, Decision tree, BayesNet and the performance was compared .A total of 2400 benign samples and 1580 malware samples were used to examine the performance of the proposed method. This method achieved 97% accurateness and 1.9% of (FPR) as a result.

Sanz Borja et al. [12] presented a model called PUMA. This model is based on permission based features. The author extracted 'Permissions' from AndroidManifest**.**xml file. Android Asset Packaging Tool was used for extracting the feature from apps. In this study, the author used 1148 apps, both malware and benign that was collected from the different source and verified. WEKA tool was used for mining and classification and k-fold method was used for validation. Different classifier method was used in this experiment. Bayesian-based classifiers obtained accuracy but Random Forest was with 50 trees obtained 86.41% among the classifier. In this study, Random Forest was good with 0.92 of accuracy.

Aung et al. [13] research was based on permission-based features and K-mean clustering classification model . In this experiment total 700 apps were used for features extraction. The dataset was built in (.arff) file format and WEKA was tool used for analysis.

Aafer et al. [14] introduced a robust and lightweight classifier called Droid API Miner that distinguishes between malware apps and benign apps. This model used API based features and Androgaurd tool for features extraction .In this research the author used RapidMiner tool to generate many classification models like ID5 DT , C4.5 DT, KNN and SVM Classifier but KNN classifier gave good results. A total of 32,000 benign apps and malign apps were used in this experiment. For testing purpose 3,000 benign apps and 1,000 malicious apps were used. Accuracy showed in this experiment was 99% using KNN classifier.

Wu et al. [15] introduced a static malware detection system that was based on manifest properties and API calls. They called their framework Droid Mat. In this model features were extracted based on permissions, components used, intents and inter-component communication which are extracted from the manifest file. API requires every segment which is extricated from .dex records. In this approach, clustering algorithms were used for evaluating the result. The best accuracy they obtained with this model was 97%.

Sahs et al. [16] proposed a model based on machine learning. The One-Class SVM classification algorithm was used for classifying the apps. This method is based on the static features. The Androguard tool was used for features extraction from apps. Permission and Control Flow Graphs based features were extracted. Both features were stored in a binary vector. The test set was made with 2,081 clean applications and 91 malicious applications, with the data divided into 5 portions based on binary vectors, strings, diagrams, sets, and uncommon permissions and another one for each application. For training purposes, a random subset of data was collected.

Ghorbanzadeh et al. [17] proposes feed-forward neural networks (NNs) that classified the android apps based on permission. This model is too reliable and capable of efficient prediction. Apktool tool is used for de-assembling of an APK and 124 permissions were extracted from the manifest.xml file and stored in a binary vector. The preparation, approval, and testing stages took 70%, 10%, and 20% of the dataset information, individually. The NN has a two-layered structure. This model got the data from a binary vector that represents the requested permissions for each application. They accomplished an exactness of 65% through 10 tests.

Sezer et al. [18] proposed a Bayesian classifier based classification model. Total 2000 apps (1000 benign and 1000 malware that belonged to 49 android malware families) were analyzed for features extraction. Features were based on API calls, permission commands, encrypted code, and presence of secondary applications or .jar files. For feature ranking purpose MI(mutual information) was calculated for each and every feature and ranked. Only top 25 features were collected in the form of binary vector. To train the model, 1600 samples (800 each) were used. For validation 5-fold cross method was used in this model. The training phase used 400 apps 200 benign, 200 malevolent. The most outstanding outcomes were got from top 20 features set and 1,600 applications (training set had 50%Benign and 50% malware) and achieved a accuracy of 97.22%.

B. Dynamic approach

Dynamic analysis Compared to static analysis, dynamic analysis is principally based on examining malware by executing or emulating the application in a real platform or real environment. This method is fast like app execution. In

this approach, android application behaviors are accounted during runtime as a log file. The main disadvantage of this technique is if it fails at some during execution on the emulator. There is much risk that the applications can fail to execute the malicious code while extracting the features. One more complex problem with this procedure is that it is really hard to apply as compared to static analysis.

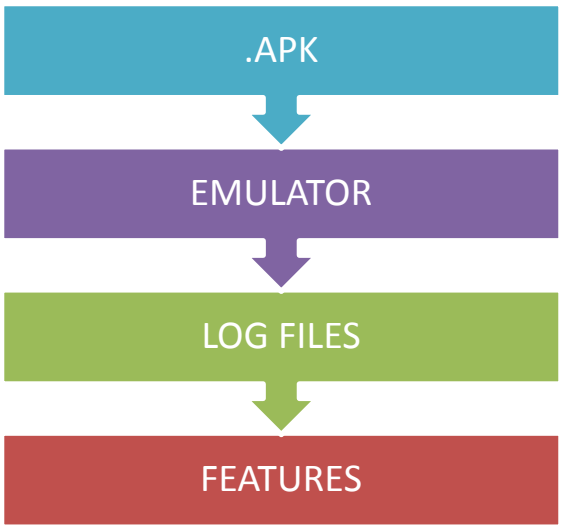

Figure 11. Dynamic feature extraction

Many dynamic analysis mechanism or tools are available for android apps like TaintDroid [25], CopperDroid [27], Andrubis [28], Apps Play-ground [29], DroidBox [26], etc. Also, there are so many tools available online for Android application analysis dynamically like NVISO ApkScan [33], CopperDroid [31], TraceDroid [32], and SandDroid [30]. Dynamic feature extraction is illustrated in table 7.

Table 7. List of dynamic analysis and their feature.

List of Dynamic Analysis	**Features**
IP Distribution Analysis	parse IP information based on the extracted URLs
File I/O	file path and data
Http Data Recovery	recover data from http
Cryptography Operation	record cryptography usage
Information Leakage Monitor	Information leakage
SMS	Record massage behavior
Network I/O	capture network data

Table 8. Research on Dynamic Feature Based Android Malware Detection System.

[Reference No.,] Author`s, Name [Year	**Technique used**	**Approach**	**Detection Method**
Aphonso et Aphonso et al. [19]	RandomForest J.48 NaiveBayes SimpleLogistic BayesNet	HIDS	API call and system call
Maline et al.[20]	random forest, SVMs, LASSO Ridge regression	HIDS	system call
Zhao et al. [21]	Support Vector Machine (SVM)	HIDS	behavioral signature
Toshiki Shibahara et al [22]	**Recursive Neural Network** (RNN)	HIDS	Network behavior based feature.
Davidt et al[23]	deep learning	HIDS	behavioral signature
Droid Dolphin et al [24]	Random forest	HIDS	Permission based

Aphonso et al. [19] proposed a model that was based on dynamic analysis. Monkey Runner tool was used for dynamic feature extraction .The API call, system call and frequency-based features were extracted. Malgenome Project dataset and some other data set were used in this experiment. Total of 7520 malicious and benign samples were used. For training purposes 3,780 and 3740 apps for test purposes were used. This system accuracy was 96.82%.

Simone Atzeni et al. [20] proposed a method based on dynamic features analysis .For dynamic features the author used sandbox and recorded the system calls in a log file and converted log the files into Feature vectors .Several classification algorithms: like random forest, SVMs, LASSO, and ridge regression were used by the author. The detection accuracy in this method was 96%.

Zhao et al. [21] proposed a model named AntiMalDroid that was based on behavioral signature .In this experiment they created a log and feature sequence that was used to detect the malwares.

Toshiki Shibahara et al [22] method is based on dynamic features analysis. Recursive Neural Network (RNN) model

was used in this experiment for classification. For dynamic features extraction Botnet Watcher tool was used by the author and network behavior based features was extracted. This method has two limitations: Long Sleep and Labeling. If malware samples sleep for a long time after the start of the dynamic analysis, this proposed method cannot be applied.

Davidt et al [23] proposed a model that was based on deep learning. Sand box application was used for dynamic features extraction. The Deep stack de-noising auto encoders deep learning model was used in this study for classification. The main advantage of this proposed model was it capability to identify new variations in malwares. This model was built with eight layers and trained with extracted features. A total of 1800 malware samples with 300 variants were used in this experiment. The Author obtained 98.6% accuracy in a moderate way.

Wu et al. [24] proposed a framework called DroidDolphin that analyzes the malwares dynamically. This framework used a combination of machine learning algorithm, a GUI based testing and big data analysis. A total of 32,000 benign apps and 32,000 malicious apps were used in this proposed model for useful feature extraction. Android Virtual Device (AVD) was used in this study for collecting logs file. Sandbox tool was used for collect the information about runtime activities .The author used many tools and a total of 13 types of activities were recorded. SVM machine-learning Algorithms were used for classification and six kinds of metrics was used to evaluate the performance of the given model. The best accuracy of this model using the cross validation was 92.5%.

CONCLUSION

There is a steady increase in android OS based devices worldwide. The escalating rate of android application development for such devices is becoming important to provide secure environment to users safe guarding from the malicious apps. The user requires such application that provides maximum security from different type of threats that are available in the android market. When the user connects to the internet they might feel insecure due to stealth and hijacking of private information. In this paper, many types of malwares and their penetrations techniques have been systematically analyzed and an overview on the modern mechanism of Android malware and their exposure techniques were presented. Also, discussed an outline of the pros and the cons in regards to the recent and past work on the machine learning algorithms in the field of Android malware detection.

REFERENCES

[1] https://www.bychico.net/does-your-android-phone-need-an-antivirus-app

[2] http://elinux.org/Android_Architecture.

[3] http://www.herongyang.com/Android/Project-Android-Application-Project-Build-Process.html

[4] https://www.gdatasoftware.com/blog/2017/04/29712-8-400-new-android-malware-samples-every-day

[5] https://developer.android.com/about/dashboards/index.html

[6] S. Young and D. Aitel, The hacker's handbook: the strategy behind breaking into and defending networks. CRC Press, 2003.

[7] A. Feizollah, N. B. Anuar, R. Salleh, F. Amalina, R. R. Maarof, and S. Shamshirband, "A study of machine learning classifiers for anomaly-based mobile botnet detection," Malaysian Journal of Computer Science, vol. 26, no. 4, 2014.

[8] Narudin, A. Feizollah, N. B. Anuar, and A. Gani,"Evaluation of machine learning classifiers for mobile malware detection," Soft Comput, nov 2014.

[9] D. Arp, M.Spreitzenbarth, M. Hubner, H.Gascon, K. Rieck, and C.Siemens, "Drebin: Effective and explainable detection of android malware in your pocket," in Proceedings of the Annual Symposium on Network and Distributed System Security (NDSS), 2014..

[10] A Deep Learning Approach to Android Malware Feature Learning and Detection,Hunan Provincial Key Laboratory of Network Investigational Technology, Hunan Police Academy, Changsha, China, 2016 IEEE

[11] Y. Du, X. Wang and J. Wang, "A static Android malicious code detection method based on multi-source fusion", in: SECURITY AND COMMUNICATION NETWORKS 2015

[12] B. Sanz, I. Santos, C. Laorden, X. Ugarte-Pedrero, P. G. Bringas, and G. Álvarez, "PUMA: Permission usage to detect malware in android," Adv. Intell. Syst. Comput., vol. 189 AISC, pp. 289–298, 2013..

[13] Aung, Z. and W. Zaw, Permission-Based Android Malware Detection.

[14] Aafer, Y., W. Du, and H. Yin. DroidAPIMiner: Mining API-level features for robust malware detection in android. in Proc. of International Conference on Security and Privacy in Communication Networks (SecureComm). 2013.

[15] Wu, D.-J., et al. DroidMat: Android Malware Detection through Manifest and API Calls Tracing. in Information Security (Asia JCIS), 2012 Seventh Asia Joint Conference on. 2012. IEEE

[16] Sahs, J., & Khan, L. (2012). A machine learning approach to android malware detection. In Intelligence and Security Informatics Conference (EISIC), 2012 European (pp. 141-147). Los Alamitos, CA: IEEE Computer Society.

[17] Ghorbanzadeh, M., Chen, Y. Ma Z., Clancy, T. C., & McGwier, R. (2013, January). A neural network approach to category validation of android applications. In Computing, Networking and Communications

(ICNC), 2013 International Conference on (pp. 740-744). Piscataway, NJ: IEEE.

[18] Yerima, S. Y., Sezer, S., McWilliams, G., & Muttik, I. (2013). A new android malware detection approach using bayesian classification. In Advanced Information Networking and Applications (AINA), 2013 IEEE 27th International Conference on (pp. 121-128). Los Alamitos, CA: IEEE Computer Society.

[19] V. M. Afonso, M. F. de Amorim, A. R. A. Gregio, G. B. Junquera and P. L. de Geus, Identifying Android malware using dynamically obtained features, Journal of Computer Virology and Hacking Techniques, 11(1):9–17, 2015.

[20] M. Dimjasevic, S. Atzeni,I. Ugrina and Z. Rakamari, Evaluation of AndroidMalware Detection Based on System Calls, Retrieved December11,2015,http://www.cs.utah.edu/docs/techreports/2015/pdf/UUCS-15-003.pdf.

[21] M. Zhao, F. Ge, T. Zhang, and Z. Yuan. Antimaldroid: An efficient svm-based malware detection frameworkfor android. In Information Computing and Applications. 2011.

[22] Efficient Dynamic Malware Analysis Based on Network Behavior Using Deep Learning Toshiki Shibahara, Takeshi Yagi, Mitsuaki Akiyama, Daiki Chiba and Takeshi Yada NTT Secure Platform Laboratories, Tokyo, Japan.

[23] DeepSign: Deep Learning for Automatic Malware Signature Generation and Classification 978-1-4799-1959-8/15 IEEE2015.

[24] DroidDolphin: a Dynamic Android Malware Detection Framework Using Big Data and Machine Learning.

[25] W. Enck, P. Gilbert, B.-G. Chun, L. P. Cox, J. Jung, P. McDaniel, and A. N. Sheth. TaintDroid: An Information-Flow Tracking System for Realtime Privacy Monitoring on Smartphones. Osdi '10, 49:16,2010.

[26] DroidBox,GoogleArchivehttps://code.google.com/archive/p/droidbox/#!

[27] K. Tam, S. J. Khan, A. Fattori, and L. Cavallaro.CopperDroid: Automatic Reconstruction of Android Malware Behaviors. Ndss, (February):8{11, 2015.

[28] U. Bayer, I. Habibi, D. Balzarotti, E. Kirda, andC. Kruegel. A view on current malware behaviors. Proceedings of the 2nd USENIX conference onLarge-scale exploits and emergent threats: botnets, spyware, worms, and more, page 8, 2009.

[29] V. Rastogi, Y. Chen, and W. Enck. Apps Playground: Automatic Security Analysis of Smartphone Applications. CODASPY '13 (3rd ACM conference on Data and Application Security and Privacy), pages 209{220, 2013.

[30] SandDroid, Hu.Wenjun, http://sanddroid.xjtu.edu.cn/.

[31] CopperDroid,http://copperdroid.isg.rhul.ac.uk/copperdroid/

[32] Tracedroid, http://tracedroid.few.vu.nl/.

[33] NVISO ApkScan - Scan Android applications for malware, https://apkscan.nviso.be/.

[34] ALPAYDIN, E. Introduction to Machine Learning,2nd ed. The MIT Press, 2010.