International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 06 Issue: 03 | Mar 2019 www.irjet.net p-ISSN: 2395-0072

Android Malware Detection using Machine Learning

Bhagyashri Chavan[1], Bhavika Tanna[2], Shivangani Jaiswal[3] ,Swati Nadkarni[4],Nida Jawre[5]

[1]*Student, Dept. of IT Engineering, Shah & Anchor Kutchhi engg. College, Maharashtra, India.*
[2]*Student, Dept. of IT Engineering, Shah & Anchor Kutchhi engg. College, Maharashtra, India.*
[3]*Student, Dept. of IT Engineering, Shah & Anchor Kutchhi engg. College, Maharashtra, India.*
[4]*Associate Professor, Dept. of IT Engineering, Shah & Anchor Kutchhi engg. College, Maharashtra, India.*
[5]*Assistant Professor, Dept. of IT Engineering, Shah & Anchor Kutchhi engg. College, Maharashtra, India.*

Abstract - *A major share in the mobile application market is occupied by Android. There is an exponential increase in the amount of malicious software which made it necessary to use machine learning algorithms for identifying malicious and genuine files through classification. Our Android application detects whether an application contains malware which can harm the mobile OS, based on APK (Android Application Package). For this, major challenges are to reduce the time consumption for detecting malware and increasing the rate of malware detection. The main aim is:*

1. *Extract features from application to detect malware.*
2. *Train the model by exploring various mechanisms of machine learning algorithm.*
3. *Notify the user whether the application is harmful or not.*

Key Words: Android, malware detection, machine learning, APK, extraction, Application.

1. INTRODUCTION

Android is a working framework essentially focusing portable gadgets, for example: cell phones and tablets. The working framework has held a vast offer of the market for a couple of long time now, and it's piece of the pie keeps on developing. Much the same as numerous other cell phone working frameworks, Android's vital moving point is countless applications-or applications for short -that are created by different gatherings. The web availability on Android-based gadgets gives these gadgets and on-going applications practically boundless routes for co-operation with other PC framework on the internet.

These certainties additionally make Android an engaging focus for vindictive programming (malware). A client may introduce a malware application on an Android gadget without realizing that the application is vindictive. A few instances of malignant activities are sending unapproved SMS, and sending delicate private data to a remote server without the client's learning and consent. So as to moderate dangers presented by a malware application, a totally programmed system to distinguish vindictive conduct is proposed.

The proposed system comprises of two parts: Testing and Machine realizing. The testing part is to test each application in a controlled situation altogether to learn how much as could be expected about it's conduct. The more intensive the testing of the application is done, better vector catching the genuine conduct of the application can be built. The machine learning segment serves the motivation behind separating these component vectors from Android applications, and afterwards utilizing the element vectors to learn parameters of a classifier that segregates amiable from vindictive applications.

1.1 AIM OF THE PROJECT

Android users face great threat like monetization through premium calls and SMS, privacy breaches, and stealing of private and sensitive information. The main aim is to provide security against such threats and classifying malicious and non-malicious apps using machine learning mechanisms to alert the user.

1.2 OBJECTIVE OF THE PROJECT

Developing an android application that detects the presence of malware contained in android application installed on the smartphone. Application provides real time analysis of the malware present and alerts the user and also provides the classification of the malware family.

1.2 SCOPE OF THE PROJECT

An application can be developed which can detect the malware in real time during the installation of malicious apps and notify the user in real time. As new and unknown malwares are created every day, so automation and dynamic analysis of malware is needed. Continuous detection increases consumption of battery and causes drainage. To overcome this issue, we can set sync frequency for different time intervals. A real time graphical representation of all the malware detected in the user's phone till date can also be provided.

2. REVIEW OF LITERATURE

Ravi et al. [1] used different machine learning algorithms such as J48, Naive Bayes, and Random Forest for detection of malware and evaluated the performance of each algorithm. They also implemented a framework for classifying android

applications that used machine learning techniques to check whether it is malware or not. Cristina et al. [2] used algorithms to illustrate different types of ensemble configurations that boosted detection rate. They concluded BDT4-ENS10-OSC-BC-3000 provided the best detection rate, but also had the highest false positive rate. So false positive filtering is required. Alireza et al. [3] presented a literature survey of different approaches of detecting malware using data mining. They classified the approaches into two categories; (1) Signature-based and (2) Behaviour-based. Different approaches were compared and analysed based on factors like accuracy, data analysis method, etc. Muazzam et al. [4] presented different approaches using file features. They provided a classification hierarchy based on features type, program analysis method, type of detection used. Kirti et al. [5] highlighted methodologies that were incorporated by security researchers to tackle different kind of attacks. They explained static, dynamic and hybrid detection techniques. In static analysis, feature is extracted without actual execution of code. In dynamic analysis code is executed actually, which makes it less safe. Schultz et al.

[6] used data mining for detecting malware using machine learning algorithm i.e. Naïve Bayes. There accuracy rate was 97.11%. They presented a method useful for detecting previously undetectable malicious executables. Their method is being implemented as a network mail filter which uses a algorithm to identify malicious executables prior to their reception through mail. They provided two options (1) wrap the potential malicious executables. (2) Block it. HA Alatwi et al. [7] They proposed category based machine learning classifier to improve the performance of classification models. Machine learning algorithms had been used to train classifiers according to features of malicious apps to build models capable of detecting malicious patterns.

M Siddiqui et al. [8] They presented a data mining framework to detect worms. Its main feature was detecting frequency of occurrence of variable length instruction sequences. Instruction sequences could be traced back for further analysis in addition to being used in classifier. It had 95.6% detection rate and a false positive rate of 3.8%. Mohammed et al. [9] They detected infections by malware on mobile devices by using signature-based and anomaly- based approaches. They tried to deduce if a machine learning classifier random forest can be used to detect malware. They began with gathering the datasets and running them on Android adb-monkey tool. Using 5-fold cross validation 99.9% of accuracy was achieved. Mengu Qiao et al. [10] They proposed a novel machine learning approach for detecting malware based on patterns of permissions of API function calls of Android applications. It began with combining features and experimenting with machine learning algorithms. It gave them good accuracy and great potential to differentiate between benign and malware Applications.

3. DESIGN AND IMPLEMENTATION

This system is based on detecting malware in Android application using APK (Android Application Package). It checks if any APK contains malware which can harm the Mobile Operating System. It uses Machine learning algorithms for malware detection.

In this application, APK will be uploaded which will be then sent to the server. The training model is present at the server where the uploaded file will be unpacked and decompile using apktool.jar.

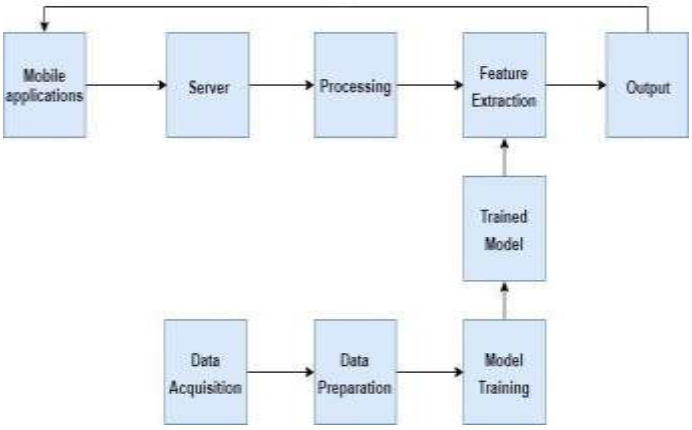

Fig-1: Block diagram

The main components of the project are:

1. Android application: This is used as interface to upload the test APK on the server and then show the result after running through trained model. Android application is developed on Android Studio which is compatible with Windows OS (7,8,10).

2. Server: This is the place where actual logic runs. It decompile the APK and processes it in the trained model and gives output if there is any malware present in the uploaded APK. The server used is Heroku Server.

3. Machine Learning Model: The model is trained with multiple APKs and contains list of features and learned parameters which is used for checking the uploaded test APK.

4. Test APK: APK is used to check whether the model is giving correct output. It contains two types of application: Malware and Benign.

3.1 APP DE-COMPILATION

In the first part, the model parses AndroidManifest.xml which contains information such as Requested Permissions, Activity names and intent filters. In particular the following

tags are extracted from this file: activity, permissions, feature, intent, and service receiver.

In the second part, the URLs accessed by the application are fetched using a regular expression, which matches every string starting with http:// or https:// . The call tag can be extracted by checking whether a line of decompiled code contains a line from apicalls_suspicious.txt. The following tags are extracted from the decompiled code: api_call, permission, url, call, real_permission.

After extraction of feature from the application we can predict if the application contains any malware. The output will be displayed in the Android application if the uploaded app is safe for use or not.

3.2 DATASET

There is requirement of large dataset containing good-ware and malware.

GOODWARE: Source: https://androozo.uni.lu /

Total APKs: 1796

MALWARE:

Source: http://amd.arguslab.org /

Total APKs: 1796 APKs

Following table shows different classes of malware family:

Sr.no	malware family	No. of APKs	Malware type
1.	DroidKungFu	340	Backdoor
2.	GoldDream	53	Backdoor
3.	Gumen	145	Trojan-SMS
4.	Ksapp	36	Trojan
5.	Latoor	312	Hacker tool
6.	Minimob	203	Adware
7.	Mseg	235	Trojan
8.	Sms Key	30	Trojan-SMS
9.	Youmi	442	Adware

Table -1 : Classes of malware family

3.3 APPROACH

TRAINING PHASE

For differentiating benign apps from malicious apps, first there is a need to upload the APK in the system after that decompilation of APK is done for further feature extraction. After feature extraction Random tree classifier Algorithm is used.

Dataset is input for the Random Forest Algorithm where the dataset is first divided into random subsets. Then a decision tree is created for each subset. On the basis of their results the trees are grouped. To classify the unknown sample of APK, as benign or malicious, group with maximum number of trees is used. For creating a malware detection model two types of classification is required i.e. binary classification and multi-class classification. Random Forest Algorithms is used to create two models- Binary model and Multi-class models. Binary classification is used to classify APK as benign or malware. On the other hand, multi-class classification helps to find category of malware present.

TESTING PHASE

This phase begins with converting the APK (.apk) into text (.txt) file. Conversion of APK into text is important for feature extraction. In order to extract valuable and relevant features a CFG is created. CFG is unique for each APK. Maximum number of features to be extracted is set in order to avoid out of memory error. The features extracted are then converted into a vector. Trained models are loaded.

This gives the probability of a given APK being a malware or benign. In order to reduce time consumption, multiple APKs are processed parallely.

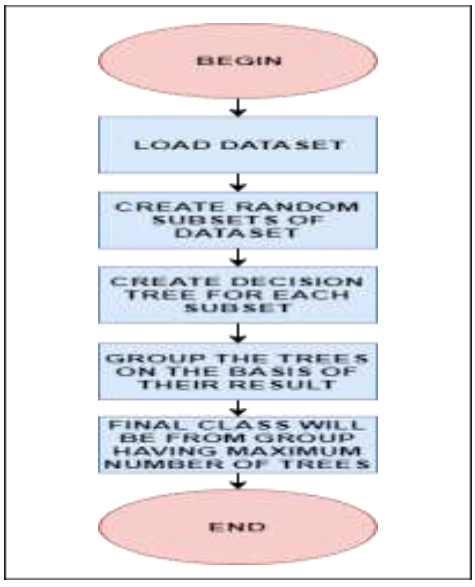

FIG -2: Block diagram of Random Forest algorithm

4. CONCLUSION

The model created uses binary as well as multiclass classification algorithm to classify an application using it's APK. The application is first identified whether it is a malware or not i.e. they are first classified into goodware and malware based on probability. If malicious, the malware is then classified into the type of malware family. The application thus alerts user about the malicious application and also what kind of unauthorized activity it performs based on the malware family it belongs to. The application detects malware in real time using the uploaded APK. It parallelly processes multiple APKs in order to reduce time consumption. The accuracy of binary classifier is 96% and that of multi-class classifier is 95.84%.

REFERENCES

[1] Ravi Kiran Varma P, Kotari Prudvi Raj, K.V Subba Raju , "Android Mobile Security by Detecting and Classification of Malware Based on Permissions Using Machine Learning Algorithms", International Conference on I-SMAC (Iot in Social, Mobile, Analytics and Cloud) (I-SMAC 2017).

[2] Cristina Vatamanu, Doina Cosovan, Dragos Gavrilut, Henri Luchian, "A Comparative Study of Malware Detection Techniques using Machine Learning Algorithms" , World Academy of Science, Engineering and Technology, International Journal of Computer, electrical, Automation, Control and Information Engineering Vol. 9, No. 5,2015.

[3] Alireza Souri, Rahil Hosseini, "A state-of-the-art Survey of Malware Detection Approaches using Data Mining Techniques", Souri and Hosseini Hum. Cent. Comput. Inf. Sci. (2018).

[4] Muazzam Siddiqui, Morgan C. Wang, Joohan Lee, "A Survey of Data Mining Techniques for Malware Detection using File Features".

[5] Kirti Mathur, Saroj Hiranwal, "A Survey on Techniques in Detection and Analyzing Malware Executables ", International Journal of Advanced Research in Computer Science and Software Engineering Vol. 3 Issue. 4, April 2013.

[6] Matthew G. Schultz and Eleazar Eskin, Erez Zadok, Salvatore J. Stolfo, "Data Mining Method for Detection of New Malicious Executables".

[7] Alatwi, Khuda Ali, "Android Malware Detection using Category-Based Machine Learning Classifiers", Thesis. Rochester Institute of Technology (2016).

[8] Muazzam Siddiqui, Morgan C. Wang, Joohan Lee, "Detecting Internet Worms Using Data Mining Techniques".

[9] Mohammed S. Alam, Son T. Vuong, "Random Forest Classification for Detecting Android Malware", International Conference on Green Computing and Communications and Internet of Things and Cyber, Physical and Social Computing IEEE 2013.

[10] Mengyu Qiao, Andrew H. Sung, Qingzhong Liu, "Merging Permissions and API Features for Android Malware Detection", 5th IIAI International Congress on Advanced Applied Informatics 2016.