Malware Detection Android App

Akshay Gaikwad*1, Ajay Kakade*2, Arati Deshmukh*3

*1,2Student, Department of Computer Engineering, MM Polytechnic, Pune, Maharashtra, India

*3Co-Ordinator, Department of Computer Engineering, MM Polytechnic, Pune, Maharashtra, India

Abstract

Android has become the most popular mobile OS in time , Due to the widespread use of Android, malware developers mostly target Android devices and users. this reason for Malware detection systems to be developed for Android devices are important. Android mobile users can download free applications from Android Application store. But, they applications were not certified by legitimate organizations and they may contain malware applications that can steal privacy information for users. Android smart phone security is built upon a permission-based mechanism, which restricts access of third party Android applications to critical resources on an Android device. The android device user accept the set of permissions that an application requires, before the installation proceeds .In this project , a framework that can detect android malware applications. our project to develop permission and a machine learning -based malware detection system on Android to detect malware applications. this application to monitors various permission based features and Source code based analysis from the android application, analysis these features by using machine learning classifiers to classify whether the application is good ware or malware.

KEYWORDS: Android, malware applications, android security

I. INTRODUCTION

Android operating system has become the most popular open source operating system for android device with an estimated market share of 70% to 80% smartphones are evolving every day and with this evolution, security issue increase in day by day . Security is an important everyone in a world, with inadequate security, it becomes an issue for the safety of the smartphone users. One of the biggest security threats to smartphones is the issue of malware., Most people are using these devices to access important services such as accessing banking applications or making e-shopping. Research to wards fighting malware and securing Android has focused on three main directions. The first one consists of static and dynamic analysis of the application code to detect malicious activities before the application is loaded onto the device. The second directions consist of modifying the Android system to insert monitoring modules to allow the interception of malicious activities occurring on the device. The third approach consists of using virtualization to implement the separation of domains ranging from lightweight isolation of an application on the device to running multiple instances of Android on the same device Attacker techniques evolved rapidly with sophistication, changing dynamically the behaviour of applications during their runtime. This poses significant challenges for their detection. Such unofficial markets have repeatedly proven to be a fertile ground for malware, particularly in the form of popular applications modified (repackaged) to include malicious code This has been proven with limitations so far. the application asks for at installation time for Permission this Permission may be dangerous user . example, applications requesting a permission to access the accelerometer of a smartphone are rather common. However, it has been demonstrated that it is possible to infer the keys pressed by the user on a touch-screen from just vibrations and motion data Therefore, using such a permission collaborate with Internet access, which is another common permission, can lead to a serious risk of user sensitive information. Additionally, Android malware is often not limited to an application's sandbox, but to the collaboration of many applications. The permission escalation attack defined by belongs to this objective. It allows malicious application to collaborate with other applications in order to access critical resources without the requesting for corresponding permissions Explicitly. it is based on the fact that the malicious unprivileged applications can take advantage of benign privileged applications to perform malicious actions. our propose Android Malware Detection System to identify malware android application with efficiency and effectiveness. To develop a Permission-based analysis and machine learning based malware detection system on Android to detect malware applications and to enhance security and privacy of android device users.

II. LITERATURE SURVEY

This Section provides a brief survey on the existing Android Malware Detection proposed in the literature by the researchers in the past. For instance [1]. Rajesh Kumar ,has presented Android Malware Detection Techniques [2]. Modupe Odusami has presented the Android Malware Detection: A Survey [3]. Nikola Milosevic has presented the Machine learning aided malware classification of Android applications [4]. Kaijun Liu has presented Android Malware Detection Approaches Based on Machine Learning [5]. Franklin Tchakounte has presented Malware Detection based on Android Permissions

III. Existing Android Malware Detection Methods

In this paper, we utilized two static malware analysis approaches: in the first one we analyzed permissions the application is requesting while in the second we analyzed the whole source code of the application. In order to automate analysis process, we experimented with two machine learning approaches, namely classification .

a)Static Analysis Approach

The approach for automated analysis is static analysis. Static analysis method considers software properties that can be investigated by inspecting the downloaded application and its source code only. this analysis tools often decompile or disassemble application packages and search for specified patterns like permissions, function calls and hard-coded variables throughout the source-code. All they techniques have in common that the suspicious application is not being executed. Signature-based detection of applications. proposed a method that detects and classifies android malware using static analysis with the combination of the attacker information.

b) Dynamic Analysis Approach

Dynamic analysis can identify application behavior at run time and it is mostly done in a sandbox environment It executes the suspicious application for that within a controlled environment often called sandbox.

C)Permission-based analysis:

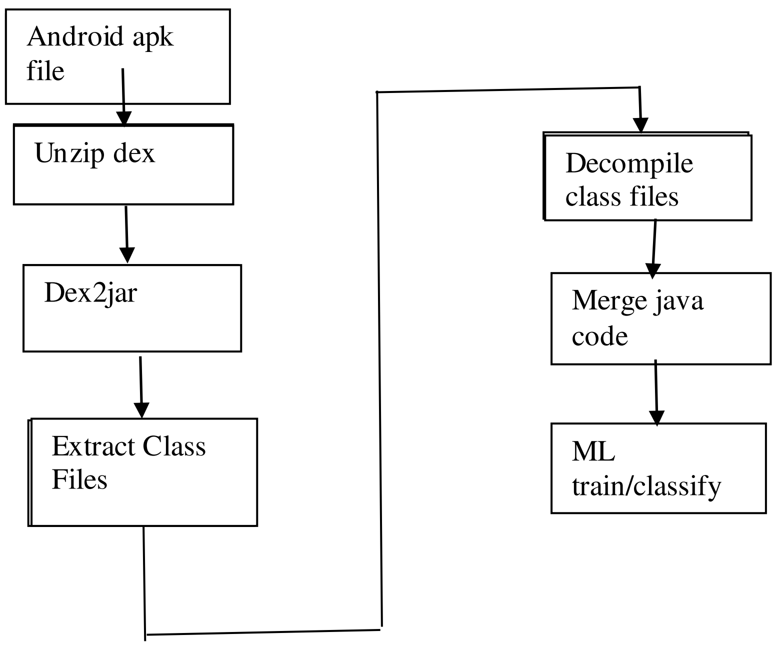
In this approach, we analyze permissions that the android applications are using and we build a machine learning model that uses permission names as features. Android security model is based on applications permissions. Every application has to acquire different privileges to access a variety of phone features. During the installation, a user is notified about the permissions requested by applications and has the option to either allow access or to cancel installation of the application. However, malicious applications usually require certain permissions i.e. in order to access and exfiltrate sensitive information from the SD card, a malicious app would require access to both SD card and the internet. Our approach is modeling the combinations of Android permissions that malicious applications are using. We propose an approach to use the appearance of specific permissions as features for machine learning algorithm. In this approach, we first extracted permissions from our dataset and create a model. For training we used Weka toolkitand tried several machine learning algorithms, including SVM. The advantage of permission-based analysis is that it is computationally inexpensive and it can be integrated on the mobile device. We used modified Weka 3.6.6 library for Android to make Android application that utilizes this model.,built using support vector machines with sequential minimal optimization, became a part of permission scanner in malware detection application. We also applied several clustering techniques in order to examine and compare the performance of unsupervised learning algorithms with the supervised ones. Training, testing and evaluation were as well done using Weka toolkit.

D) Source code based analysis:

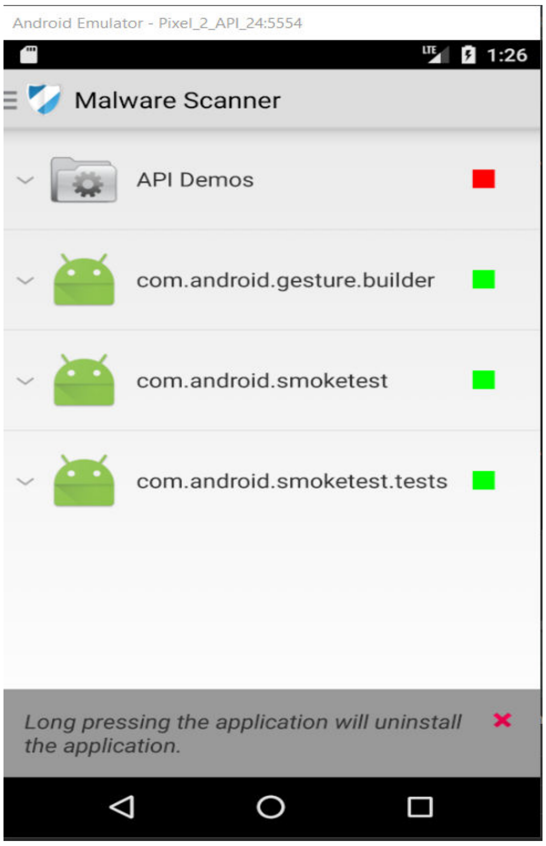
The second approach is based on static analysis of the application's source code. Our assumption was that malicious codes are using a combination of services, methods and API calls in a way that is not usual for benign applications. Machine learning algorithms are able to learn combinations of methods, API and system calls that are common for malware and distinguish them from the patterns that are usual for benign applications. In this approach, android apps are first decompiled and then using decompiled code and text mining classification approach based on a bag of words to train a model. Decompiling Android applications to conduct static analysis involves several steps. It is first necessary to extract Dalvik Executable file (dex file) from the Android application package (APK file) by unzipping Android application package. The second step is to transform Dalvik

Executable file to Java archive using dex2jar tool (Pan 2014). Afterwards, we extract .class files from the Java archive and utilize Procyon Java decompiler (version 0.5.29) to decompile .class files and create .java files. Then we merge all Java source code files of the same application into one large source file for further processing. Since Java and natural language text do have a certain amount of similarity, we applied the technique that is used in classification of natural language processing called a bag of words. In this technique, the text, or code in our case, is represented as a bag or set of words, disregarding grammar or word order. The model is taking into account all the appearing words (Joachims 1998; McCallum et al. 1998). Our approach considered whole code (including import statements, method calls, arguments, instructions, etc.). The source code obtained in previous step was tokenized into unigrams that are used as bag of words. We used several machine learning algorithms for classifications namely C4.5 decision trees (in Weka toolkit called J48), Naive Bayes, support vector machines with sequential minimal optimization, random forests, JRIP, logistic regression and AdaBoostM1 with SVM base. We performed our training, testing and evaluation using Weka toolkit. For source code analysis we also applied ensemble learning with combinations of three and five algorithms and majority voting decision system. Again, the number of algorithms were chosen, so system is able to unambiguously choose the output class based on majority of votes.

Fig. 1 Workflow of android file decompiling and machine learning based malware detection method



IV. GUI OF THE APPLCATION


Android Emulator - Pixel_2_API_24:5554
1:26
Malware Scanner
API Demos
com.android.gesture.builder
com.android.smoketest
com.android.smoketest.tests

Long pressing the application will uninstall the application.

V. RESOURCES USED

Hardware for pc/laptop :

RAM : 4GB RAM minimum, 8 GB RAM recommended (plus 1 GB for the Android Emulator)
Hard Disk Space : 2 GB of available disk space minimum, 4 GB recommended (500 MB for IDE plus 1.5 GB for Android SDK and emulator system image)
Input device :Standard Keyboard and Mouse
Output device :1280 x 800 minimum screen resolution of laptop or monitor for pc's

Software for pc/laptop :

Operating System : Windows 7/8/10
Language : java
Tool : jdk, Android Studio

Hardware for mobile device :

Sound : speakers
Camera :any camera present in the device
Screen : Touch screen
Connectivity : internet support

Software for mobile device :

Operating System : Android operating system minimum version 4.0 (Ice Cream Sandwich)

VI. CONCLUSION

In this paper, we have developed a application for classifying android application using and permissions machine learning techniques .we implement an Android malware detection system based on SVM.it can detect the malware and good ware Android application based on the machine learning. We extract various features with the method of static analysis and dynamic analysis features with the method of static analysis and dynamic analysis .we proposed to use permissions, API calls of Android applications to detect malware and malicious application in Android based mobile platform. we presented two machine learning approaches based on app permissions and source code analysis to detect malware on Android devices. The permission-based method was able to classify malware from good ware in 89% of cases while source code analysis classification performance was over 95%. used dynamic analysis of the android applications in combination with machine learning .the only automated static malware analysis method for android applications that uses machine learning. static and dynamic application analysis in which multiple machine learning classifier would be applied to analyze both source code and dynamic features of application in run-time.

ACKNOWLEDGEMENTS

We got this opportunity to thank all the individuals connected with this project for their useful direction, help and timely support which helped us to complete the project in specify amount of time. We would like to express great gratitude to our HOD Mr. V. S. Solanke and our project guide Mrs. Arati Deshmukh for their all- important support, motivation, guidance and helpful suggestions all over the project work. Lastly but not least our sincere credit goes to our family for their key support since we begin our education and also to our group person.

VII.REFERENCES

[1]. Wang, W., X. Wang, D. Feng, J. Liu, Z. Han, and X. Zhang: the Exploring Permission-Induced Risk in Android Applications for Malicious Application Detection.

[2]. Nikola Milosevic has presented the Machine learning aided malware classification of Android applications

[3]. orregrosa, B.: A framework for detection of malicious software in Android handheld systems using machine learning techniques. Master thesis, Universitat Oberta de Catalunya,Barcelona 2015

[4].The Permission-based Malware Detection Mechanisms on Android: Analysis and Perspectives (researchgate.net)

[5]. https://www.sciencedirect.com/science/article/abs/pii/S0045790617303087 (2010)

[6].Brodeur, P.: Zero-permission Android applications.http://leviathansecurity.com/blog/archives/17-Zero-Permission-Android-Applications.html.

[7].Ehringer, D.: The Dalvik Virtual Machine Architecture, Technical Report, March 2010. [Enck/Ongtang/McDaniel 2009]

[8]. Singh, R.: A overview of the android operating system and its security. Int. J. Eng. Res. Appl. 4 (2), 519–521 (2014)

[9] https://media.kaspersky.com/en/enterprise-security/Kaspersky-Lab-Whitepaper-Machine-Learning

[10].Android Manifest Permissions.http://developer.android.com/reference/android/Manifest.permission.html. Visited: January 2015.