CSE3502 – INFORMATION SECURITY MANAGEMENT

Review 1

ANDROID MALWARE ANALYSIS

Prof. In-Charge:

Dr. Vimala Devi

Team Details:

Harshit Mishra (19BCE0799)

Alokam Nikhitha (19BCE2555)

Shreeyam Sharma (19BCE2700)

Abstract:

Android is an open-source Operating System with more than a billion users. The amount of sensitive information produced be these technologies are rapidly increasing, which attracts a large number of audiences to develop tools and techniques to acquire that information or to disrupt the device's smooth operation. Despite several solutions being able to guarantee an adequate level of security, day by day the hacker's skills continues to grow, so it remains a permanent challenge for security tools developers to ensure the security of an android powered device.

As a response, several members of the research community are using artificial intelligence tools for android security, particularly machine learning techniques to classify between healthy or malicious android application.

In this project, we will implement a static framework and machine learning to do this classification.

Problem Statement and Objective:

Android is an open-source operating system for mobile devices, televisions automobiles and smart watches with more than a billion users. Therefore, it opens a wide array of attack vectors targeting the user information.

For the protection of the information and devices, android has several security mechanisms; the most relevant are: a sandbox environment at the kernel level to prevent access to the file system and other resources; an API of permissions that controls the privileges of the applications in the device; security mechanisms at the applications development level; and a digital distribution platform (Google play store), where the processes are implemented to limit the dissemination of malicious code.

Each application is compiled in an Android Application Package [APK] file, which includes the code of the application in ". dex" files, resources and the AndroidManifest.xml file. This latter is an important element, since it provides most of the information of the security features and configuration of each application. It also includes the information of the API regarding permissions, activities, services, content providers and the receiving broadcasts.

There are several tools and techniques for the analysis of threats for this operating system. Between the most representative, we have static analysis and dynamic analysis.

Static Analysis:

Static analysis is a technique that assesses behaviour in the source code, the data, or the binary files without the direct execution of the application. Its complexity has increased due to the experience that cybercriminals have gained in the development of applications. However, it has been demonstrated that it is possible to avoid this using obfuscation technique.

Dynamic Analysis:

Dynamic analysis is a set of methods that studies the behaviour of the malware in execution through gesture simulations. In this technique, the process in execution, the user interface, the network connections and sockets opening are analysed. Alternatively, there already exist some technique to avoid the processes performed by dynamic analysis, where the malware has the capacity to detect sandbox-like environments and to stop its malicious behaviour.

Literature Survey:

Paper	Problem and Objective	Proposed Methodology	Limitations
Machine Learning with Dynamic Analysis **Wen-Chieh Wu and Shih-Hao Hung** October-2014 Association for Computing Machinery	They had same problem statement as of ours. They wanted to study the malicious applications in an android environment through sandbox technique. Smartphones are getting more and more popular nowadays with various kinds of applications to make one's life more convenient. The malwares in use currently steals users' information. Some of them might send SMS or make telephonic calls which can results in additional charges employed by network provider without one actually knowing the reason. Thus, the detection of a malware is very critical to ensure ones' safety and privacy.	In this work, they have proposed a DrodDolphin named dynamic malware analysis framework which leverages the technologies of GUI-based testing, big data analysis, and machine learning to detect malicious Android Applications. The automatics testing tools were able to extract useful static and dynamic features from their training datasets and were able to classify an application as benign, that is safe, or malicious, that is unsafe.	This process is highly time consuming because it works only when the malicious execution takes place. The process can be bypassed by knowing when a sandbox environment is being set-up, which detects if a dynamic analysis is currently being executed. The APK can take up to 5 minutes to detect its execution which further delays the execution process.
A Machine Learning Approach to Android Malware Detection **Jusitn Sahs and Latifur Khan** April-2014	With the recent emergence of mobile platforms capable of executing increasingly complex software and the rising ubiquity of using mobile platforms in sensitive applications such as banking, there is a rising danger	They used an open-source project, named Androgaurd to extract features from packaged Android Applications (APKs). They then used these extracted features to train a One-Class Support Vector Machine	Their model is limited to just the permissions (built-in and non-standard), and configurations of the input applications. There are many potential sources of information rich features which their system lacked.

European Intelligence and Security Informatics Conference	associated with malware targeted at mobile devices. The problem of detecting such malware presents unique challenge due to the limited resources available and limited privileges granted to the user.	(SVM) using Scikit Learn framework which provides a convenient interface to LIBSVM. The classifier then classified the applications into being safe or not based on their permissions that they seek on the android platform. The main idea was to classify most of the data as positive and classify a few negative only if it is sufficiently different from the training data, making it ideal for their purpose.	Further, the malicious applications can easily by-pass this process of testing by obfuscation process. The process although is very fast, it is not the most reliable one to be worked with.
An Android Malicious Code Detection Method Based on Improved DCA algorithm **Chundong Wang, Zhiyuan Li, Liangyi Gong, Xiulian Mo, Hong Yang and Yi Zhao** February-2017 Entropy-Based Applied Cryptography and Enhanced Security for Future IT Environments	Android Malicious Code has increased dramatically and the technology of reinforcement is increasingly powerful. Due to the development of code obfuscation and polymorphic deformation technology, the current android malicious code static detection method whose feature selected is the semantic of application sources code cannot completely extract malware's code features. The Android malware static detection methods whose features used are only obtained from	This paper proposed a Dendritic Cell Algorithm (DCA), which is an Android malware algorithm that has a higher detection rate, does not need to modify the system, and reduces the impact of code obfuscation to a certain degree. This algorithm is applied to an Android malware detection method based on oriented Dalvik disassembly sequence and application interface (API) calling sequence. Through the designed experiments, the effectiveness of this method is verified for the detection of Android Malware.	The working of proposed system works only if they can access the Android Packaging (APKs) of a particular application. It requires a large number of APKs to deliver optimal working. With less than 400 APKs the accuracy of model stood at a mere 92%. With around 750 APKs it went to a count of 97%. Hence we need large number of APKs for it to work accurately.

	the AndroidManifest.xml files are easily affected by useless permissions.	This is a dynamic implementation approach which studies the execution phase of an application and classifies them as safe or not based on it.	
Machine Learning Aided Android Malware Classification **N Milosevic, A Dehghantanha and k Choo** Februray-2017 White Rose university Consortium	Malware have been used as a means for conducting cyber attacks for decades. With adoption of smartphones, which stores lots of private and confidential information, made them an important target for malware developers. Android as the dominant mobile operating system has always been an interesting platform for malware developers and lots of Android malware species are infecting vulnerable users everyday which make manual malware forensics would assist cyber forensics investigators in their fight against malicious programs.	In the work, they have proposed two Machine Learning based approaches for static analysis of the mobile applications: one based on permissions, while the other based on source code analysis that utilizes a bag of word representation model. Their source-code based classification achieved F-score of 95.1%, while the approach that used permission names only performed with F-measure of 89%. Their approach provides a method for automated static code analysis and malware detection with high accuracy and reduces smartphone malware analysis time.	Their approach had few limitations. For permission-based approach they reported an F-score of 87% for single machine learning algorithm. This means there are chances that some malware loaded applications were not classified as such and some benign applications are classified as malicious. Also, the False Negative rates in their case is high, which means that the detection rate of malware applications was low and this possessed a potential vulnerability in classification due to this alarming false negative rates.
Droid permission Miner: Mining Prominent permissions for Android Malware Analysis **AM Aswini and P Vinod**	Android is the most popular operating system that has held its root stronger than any other OS in the global smart phone market. The genuine Android applications are available at the android market	In this work, they have proposed a static analysis of android malware files by mining prominent permissions. The proposed technique is implemented by extracting	The dataset that they used was biased towards benign labels, the ratio being 2:3, that is 40% malicious and 60% benign. This suggests that the results of theirs is not totally trust

November-2014 The fifth international Conference on the Applications of Digital Information and Web Technologies (ICADIWT)	whereas third party applications are also developed progressively each day. Due to its ease of modifiability and open-source nature, there is a greater chance for the malicious code to be injected to the android applications. Also, the third-party applications have led to the proliferation of the malicious software. Android devices are vulnerable to threats and some of the noticeable vulnerabilities are Denial of Service Attacks, execution of code by the attackers using Android debugger bridge (adb), stack-based buffer overflow resulting in arbitrary code execution, memory corruption to gain root privileges, SQL injection to retrieve useful information, etc.	permissions from 436 .apk files. Feature pruning is carried out to investigate the impact of feature length on accuracy. The prominent features that give way to lesser misclassification are determined using Bi-normal separation (BNS) and Mutual Information (MI) feature selection techniques. Results suggested that Droid permission miner can be used to preliminary classification of Android Package files.	worthy because of label biasness and attribute disorder. Although the biased label ratio is not very high and it can still be considered for research purposes. They have employed various feature selection processes which makes it difficult for anyone to keep track of mathematical explanation for a newbie. Despite deploying these many features selection processes, the accuracy of their best proposed model stands at a mere 81.56% which is only good for study purposes but can't be used commercially or implemented functionally.