ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 2, Issue 3, March 2014

Detection and Prevention for Malicious Attacks for Anonymous Apps

Saranya .T[1], Shalini .A.P.[2], Kanchana .A[3]

Assistant Professor, Dept of IT, Panimalar Engineering College, Chennai, India[1]

Assistant Professor, Dept of IT, Panimalar Engineering College, Chennai, India [2]

Assistant Professor, Dept of CSE, Panimalar Engineering College, Chennai, India[3]

ABSTRACT: In the present situation people wanted their work to be completed easily in this smart phones plays a vital role in their work. The forthcoming open operating system will focus not on desktops system, it will be on devices that we use every day (E.g. Mobile). The objective is to design and develop an effective Algorithm to detect malicious applications which posing threat to hand held devices (in Android platform). This mainly focus on the Vulnerability of Android application and its permissions, since critical data is manipulated using hand held devices for Banking etc., Since Android is an open source platform lot of third party applications are developed and uploaded on Android market and Third party application stores. Users download it via their devices or download bulks of apps from the torrent. The main problem is that users are unaware of the third party applications which works as a simple application but contains malicious application in it. The proposed algorithm realizes a malware detection system that continuously monitors various features and events obtained from the device, if the application is declared malware by the algorithm then it is prevented.

The proposed approach which detects malware based on the samples of known malware. The new environment will lead to new applications in their markets to enable greater integration. The proposed work not only checks permissions but it also involves feature selection method to find the best performance in detecting new malware. The results suggest that proposed work is effective in detecting malware on Android devices.

KEYWORDS: Android, Malware, Mobile devices

I. INTRODUCTION

Mobile phones in the past are simple devices capable of performing some basic phone functions, by the release of newer smartphone operating system, mobile phones began to include advanced features like desktop which caused naive users (and made application developers) to think differently about mobile devices. Nowadays, Smart phones - Mobile phones with advanced features such as always on full-featured web browsers, Internet Connectivity, and multimedia capabilities - have become extremely popular. Additional applications such as games, productivity and communications are developed by third-party developers for entertainment purpose. The developed third-party application can be placed directly on the Android market, and there won't be any review on the application. While using Android phone for the first time, it requires application to be signed once, Google uses these signature for bookkeeping. Users can download Android application [1] from anywhere, not only from Google play (Android Market). There have been a handful (plenty) of malicious applications in Android; most of these applications were available in Third party application store and in markets other than Android. The Android market is open for all developers for their easy use; this constraint also allows ease of entry to malware developers.

Android Developers utilize a comprehensive SDK (Software Development Kit), with ample tools for development of powerful, applications with rich feature. The basic architecture diagram is given in Figure 1. It follows layered architecture, the first layer Application involve applications we work. The Framework involves the communication of applications with activity. The Libraries SQL involves data storage, web kit involves viewing HTML page, and Android is built in Linux Kernel.

II. RELATED WORK

In [1] the author in their paper attempts to unmask the complexity of Android security. The technique involves the mitigation module which acts like a browser where the user can connect to a third party application store, and download their favorite application. Once downloaded the user can check the application by running in the mitigation module, where the application is decompiled to access the Andromaly Framework. If the malicious permissions are found in the decompiled app then that is a malware application, which particular application is unsafe to be installed in the android device and it is mitigated or prevented from installing onto the android device. In their work, Machine Learning anomaly detectors are applied to classify the collected datas as Benign or Malicious. Applications statically identify the permissions that govern the rights to their data and interfaces at installation time. In Machine Learning application, a large number of extracted features, some of which redundant or irrelevant is handled by fine feature selection in a preparatory stage enabled. In [2] the authors in their paper focused on the finer-grained security model for quantifying mobile application functionality. This work discusses on quantifying application functionality which views on vendors operational field and security flaws. This enforces a finer-grained security permission model that identifies and thus thwarts a wide range of malware. The Android manifest.xml also contains information about application's content providers and required features. The author in their paper attempts to automatically detect the privacy leaks in android applications when used on a large scale. Since Android operating system provides a permission-based security model that restricts an application's access to the user's private data. Each application statistically declares the sensitive data and functionality that it requires in a manifest. This work discussed to combat this problem framed an Android Leak a static analysis framework for finding potential leaks in sensitive information in [3]. In [4] author in his paper discussed on exploring vulnerabilities of attackers in Android. Since Android offer a public market place, the application store and the Android market take dramatically different approaches to limit malware on their devices. Developers can directly place their application on the Android Market and there's no review of the application before they arrive there. This work proposed an Android Sandbox model, which is application specific. In [5] author their paper discussed about the installation of software on smartphones in a secure manner. This work involves a security framework that includes a novel classification of third-party application installation models. Process and file system isolation is provided primarily by making each application run as its own user. Android makes no security claims or assumption that the custom virtual machine itself provides security. This can specify (in a manifest file) permission labels that protect their own interfaces, or labels to request access to another application protected interfaces. In [6] author in their paper discussed about the components communication in an android application.This work discusses about the components interaction with each other by a simple message and component address. The invocation of this method tells the android framework to begin executing code in the target application. The openness of these new environments will lead to new applications and markets will enable greater integration with existing online services. However, as the importance of the data and services our cell phones support increases, so too do the opportunities for vulnerability.

A. *Android Latest Version*

Android 5.0 (Key Lime Pie) is a progression of the platform that provides improved performance and enhanced user experience. It adds new features for users and application developers. This provides an introduction to the most notable and useful new APIs for application developers. As an Android application developer, Android 5.0 is available with an SDK platform. The new Mobile OS with feature, better multiple device support, performance profiles enhanced social network support, Line-drawing keyboard options, video chat app, multi-select in app and many more. Each version increases it supports with the Application Program Interface (API).

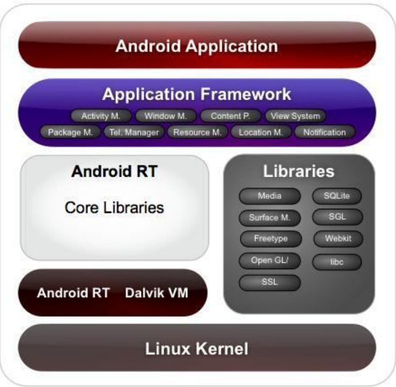

Fig. 1 Android Architecture**

B. *Android API*

API known as Application programming interface for android is a platform specification intended to be used as an interface by components communicate with each other. An API may include **Source: Internet (Default Architecture for Android Platform) specification for specific routines to structures, object classes, and variables.

III. Mobile Malware

The Mobile phones differ from conventional desktop, in mobile devices/handheld devices, the resources are limited in terms of power, consumption of energy and memory. A malicious (or) malware application targets mainly this weakness.

A. *Android*

Android is a widely anticipated open source operating system. Android developers can upload (or) place their application on the Android market. Android mobile requires signature for authentication purpose, Google will be using these signature as authentication of users and that's why android users able to download application not only from the Android market, but from any third party application store. An application is Android is said to be malicious on the basis of comments and rating from user on the application. They will have a note on how many users downloaded the application and reported in their comments. If more number of users suggests negative aspect on a particular application then Google will delete the application from Android market, it is also said that it has the ability to access devices remotely by removing it. For android users, the only way to protect devices from malicious application is to download application in Android market, not in any other third party app-store and also before downloading the application, the users need to check how many times the application have been downloaded, with positive comments. Since Android market's openness with pros allows easy for the developers organized around the world, and with cons allows entry to malware developers. It is in the hands of user to prevent the entry of malware.

B. *Techniques for Malware Detection*

The fast growing infrastructure in communication & handheld devices are susceptible to various vulnerability attacks. The usual way of injecting these attacks is by means of malware application (malicious) such as viruses-Genimi, GG Tracker, and Trojan horse, worms [2] which will spread among devices and may lead damage to users, their data's(which are confidential) can be hacked by the malicious developer. The growth in the Internet paves way for increase in malware application. There might be several techniques for detecting malware. Basically it is classified into two types of analysis static and Dynamic. In static analysis, it based on previous observation, since most of the android applications are signature based, they can be easily bypassed, when the application is fast and more effective this analysis is limited, the malicious application can evade into your device. The obfuscating code makes the scanner to believe that the particular application is Benign. In case of dynamic analysis, the information about a particular application is analyzed and detected at the time of execution (i.e. run time) it depends on parameters such as usage of CPU, modifications in memory, more consumption of battery power. The distinguish behavior of this particular feature is reported dynamically [3]. When a method has pros, it is evident that it will have some cons, first the time required to observe the activities of particular malicious application and second it is very difficult to simulate the applications based on some appropriate conditions in which the malware applications will be activated.

C. *Comparison with Existing System*

In Past, it normally states the provision for application and its reaction to the malicious attack. Now the system to be demonstrates on the malicious attack in handheld devices. In handheld devices, the system and we designed the mechanism to detect malicious attack and the enhancement to detect analysis with the proposed algorithm and existing with respect to time and space.

IV. Architecture Diagram

The Proposed architecture diagram follows the basis of Android Architecture in which naive users works in Android Mobile with the help of GUI (Graphical User Interface), he can download Android application not only from Android market, but also from others since it uses signature for authentication. While downloading the application, the user is unaware whether it is Benign (or) malicious.

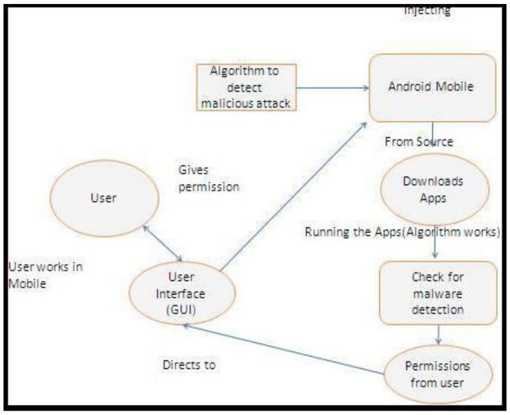

Fig. 2 Architecture Diagram for Proposed work

Here our purposed algorithm to check the changes in their features by monitoring permissions and parameters using (cross-validation). The changes in the system usage parameters for monitoring events can be collected using knowledge based temporal abstraction (patterns). To enhance the accuracy, we use cross-validation algorithm. If we collect n/10 data set, then it is divided into n/10 sets each set containing n/10 different values. In this ((n/10)-1) sets are used for training and ((n/10)-9) set is used for testing purpose, in the same way all the n/10 data sets are simultaneously tested. If the average value for features goes beyond the limit then by use of Alert Manager, it gives alarm to the user that this application is suspended to be malicious.

A. *Malicious Attack Illustration*

Developer takes legitimate application and repackages it with malware using Android manifest (AndroidManifest.xml). As discussed in Figure 3 Malicious Developer uploads Application to third party application store; Malicious Developer can control the phone remotely and access user's private information.
The Malicious Developer can view the location, contact information, send and read sms, place phone calls etc., Such an application is said to be malicious.

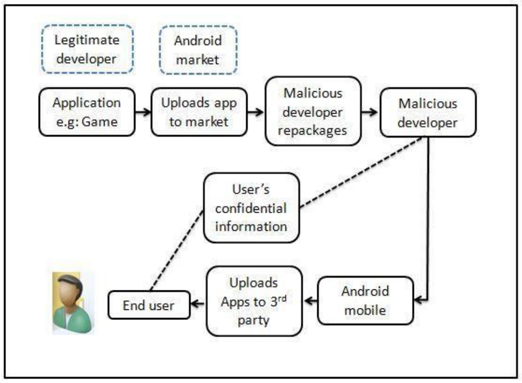

Fig. 3 Malicious Attack Illustration

B. *Importance of Security*

Modern operating systems and desktop computers have faced many challenges in integrating their applications in business, personal and other services on a unique platform. In a mobile platform, which an open source operating such as Android is it remains beside the point (complex) in which researchers states about the complications on their environment which software(legacy) can cause. Android works with applications that are executed on the java middle layer running in Linux kernel [6]. The Android uses (.apk) extension for application (Android Application Package File). It uses .dex (Dalvik Executable Format) as described in Figure 4 in which the application can be executed with high resource in a low environment.

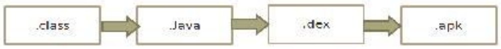

Fig. 4 Making of APK

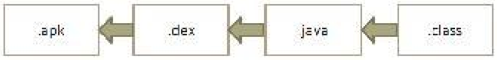

Fig. 5 Reverse Engineering

Android uses permission label model to restrict access for resources and communication with other applications, but for several reasons the developers have added potentially obfuscating refinements as given in Figure 5 which cause complexity in Android security.

C. *Types of Threats*

The Application which cause significant changes (or) sends confidential information about user is said to be malware. It is mandatory to note that Android platform does not have any significant viruses (or) malicious programs that may spread through handheld devices. The malicious developer will focus on tracking the information about the users. The Application that tends to do one thing but would perform another without the knowledge of user is said to be malware. This is commonly known to be Trojan horse. SMS Trojan are nearly half the malicious Android application, it will send text messages in the background without knowledge of user, which may have premium-rate number services that cost money. What will be the result? It will reflect in huge charge on monthly bill. Genimi is the most sophisticated malware till date which creates multiple variants on the same device. It identifies each device to the hacker with unique IMEI & IMSI, mainly for information stealing. It may call a number, send Email, sends all the sms'es to the server etc., the main theme is that it has been hosted at Android Market. There are more other malware applications such as snake, Dream Droid, GG Tracker, Android/Fake Token [4] etc. In recent years, Google transformed Android market to Google play which involves services such as music, apps, etc. There began cybercrimes/cybercriminals by creating fake domain in Google play designed to track users information by installing malicious application.

Another type of threat, that passes malicious application into the device, by means of websites. According to EU cyber security group, the European Network and Information Security Agency (ENISA), warned that drive-by-Exploits [7] which inject malicious code in form of HTML websites that exploit vulnerabilities in web browsers, targeting plugins such as Flash, Adobe Reader and Java. Depending on the mechanism, the malicious application may crash the device that may restart the device to execute a hasty payload. The Drive-by-Exploits technique, exclusively works as breaking the patches which disables code [9, 10] signature and breaks all the protection. It paves way as a platform for malware and disables execution by allowing them access information

V. Experimental Results

The rapid growth in internet and technology in communication will pave way for both good and bad to environment. The person who steals another's info is known to be intruder. There are not of mechanism involved to prevent this but the goals are high. The IDSs (intrusion Detection Systems) must maximize the security goals. In order to calculate the impression of algorithm to detect the malicious application, we measured the CPU consumption for each application from our datasets collected and also the memory consumption. It is not evident that the application which consumes more CPU are said to be malicious. In other way permissions are recognized as the most important security feature in Android. Android smartphones uses permission-based model [8] to decrease the Application's behavior and it will also provide details about potential behavior to user. It is declared in AndroidManifest.xml file. This presents list to users, and it is up to the user's interest to install the application. The permissions can't be selected selectively either all (or) none should be done. So, we also evaluate the capacity of permissions to detect malware/malicious application based on machine-learning with the data set collected. In order for validating, collected 200 malware samples of Android applications the features are monitored for each application and evaluated using the Area under ROC curve (AUC). The number of Permission in application are shown in Figure 6

To evaluate the capability of our algorithm, we measure the True Positive Ratio (TPR), since experts over reverse engineering tools are more in Android.

TABLE I Android Malware Detection

Algorithm	TPR	FPR	AUC	Accuracy
SimpleLogistic	0.90	0.20	0.87	84.00%
NaiveBayes	0.52	0.17	0.80	68.84%
DecisionTree	0.91	0.19	0.86	85.77%
RandomForest 25	0.92	0.20	0.90	85.72%
RandomForest 50	0.90	0.18	0.93	85.51%
Crossvalidation	0.91	0.20	0.93	86.45%

Table 1 shows the obtained results. The Bayesian-based classifers, gives an accuracy higher than 80%. The RandomForest is trained with 25, 50 trees which yields various accuracy. In AUC, the cross validation and RandomForest leads higher accuracy with TPR above 0.90. With reference to [1] the values TPR, FPR, Accuracy are calculated

$$TPR = \frac{TP}{TP+FN} \quad .... \quad (1)$$

Using (1) TPR gives the positive instances (in which Benign application are classified as Benign), TP is the number of malicious application classified correctly (True Positive) and FN is the number of malicious application misclassified as Benign (False Negative)

$$FPR = \frac{FP}{FP+TN} \quad .... \quad (2)$$

We can also compute, False Positive Ratio (FPR) using (2), in which FP (False Positive) is benign application misclassified as malicious and TN (True Negative) is the legitimate/ benign executed correctly.

$$\text{Total Accuracy} = \frac{TP+TN}{TP+TN+FP+FN} \quad ... \quad (3)$$

Further, we can also measure the accuracy by (3) under the positive instances divided by the total instances in dataset.

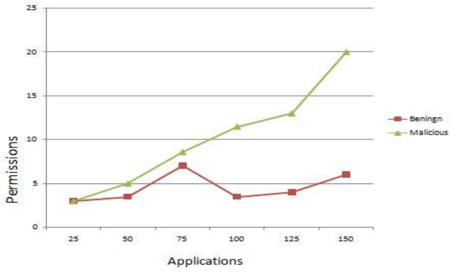

Fig. 6 Number of permissions of benign and malware apps

VI. CONCLUSION AND FUTURE WORK

Android is a widely anticipated open source operating system, since it has been introduced and still explored for its security mechanisms. There are many solutions where proposed to prevent malware applications invading in our handheld device. Many applications are developed for handheld devices, since critical data is manipulated using handheld devices in areas as E-Banking, E-billing etc. in which transactions are involved; there is a need for security in using these applications through handheld devices. In our proposed work to detect application that cause malicious threat in handheld devices using algorithm with prevents it, intend for posing threat to handheld devices. Our future work of Android malware detection tool is targeted in two directions; first, there are other features from which application can be detected as malware and it also increases the ratio of detection, more tools are to be developed to obtain new features to detect malware. Second, the dynamic analysis on the malware system, which includes detecting, changes in the device which persists for a long time. But, handheld devices resources are limited and analysis consumes resources that can be used for reporting suspicious behaviour of application to Android market.

REFERENCES

1. Asaf Shabtai, Uri Kanonov, Yuval Elovici, Chanan Gleze, and Yael Weiss, "Andromaly": A Behavioral Malware Detection Framework for Android Devices", Journal Intell Inf Syst Springer Science+Business Media, 2011.
2. Angelos Stavrou, A. Voas, J. Karygiannis, and T.Quirolgico, "Building Security into off-the Shelf smartphones", IEEE Journals and Magazines, IEEE Computer Society 0018-9162/12, pp. 82-84, 2012.
3. Clint Gibler, Jonathan Crussell, Jeremy Erickson and Hao Chen, "AndroidLeaks: Automatically Detecting Potential Privacy Leaks in Android Applications on a Large Scale", Android Developer Reference. Accessed March 30, 2012 http://d.android.com/, 2012.
4. Charlie Mille "Mobile Attacks and Defense", IEEE Journals and Magazines, IEEE Computer and Reliability Societies 1540-7993/11, pp. 68-70, 2011.
5. David Barrera and Paul Van Oorschot, "Secure Software Installation on Smartphones", Carleton University IEEE Journals and Magazines, IEEE Computer and Reliability Societies – Security and Privacy, pp. 42-48, 2011.
6. William Enck "Understanding Android Security", Pennsylvania State University IEEE Journals and Magazines, IEEE Computer Society 1540-7993/09, IEEE Security and Privacy pp. 10-17 computer.org/security/, 2009.
7. Yeongung Park, ChoongHyun Lee, Chanhee Lee, JiHyeog Lim, Sangchul Han, Minkyu Park and Seong-Je Cho, "RGBDroid: A Novel Response-Based Approach to Android Privilege Escalation Attacks", In Proceedings of ACM CCS, 2011.
8. Y. Zhou, X. Zhang, X. Jiang and V. Freeh,"Taming Information-Stealing Smart-Phone Applications (on Android)". IEEE Journals and Magazines Trust and Trustworthy Computing, pp. 93-107, 2011.
9. Borja Sanz, Igor Santos, Carlos Laorden, Xabier Ugarte-Pedrero, Pablo Garcia Bringas and Gonzalo Alvarez, "PUMA: Permission Usage to detect Malware in Android", International Joint Conference American International School (AISC) 189, pp. 289-298 Springer-Verlag Berlin Heidelberg 2013.
10. Asaf Shabtai, Robert Moskovitch, Clint Feher, Shlomi Dolev and Yuval Elovici, "Detecting unknown Malicious Code by Applying Classification Techniques on Opcode patterns", a SpringerOpen Journal on Security Informatics, 2012.
11. Asaf Shabtai, Robert Moskovitch, Yuval Elovici, and Chanan Glezer, "Detection of Malicious Code by Applying Machine Learning Classifiers on Static Features: A State-of-the-Art Survey", an Information Security Technical Report 14, pp. 16-29, Elsevier 2009.
12. Asaf Shabtai, "Malware Detection on Mobile Devices", IEEE Eleventh International Conference on Mobile Data Management ISBN 978-0-7695-4048-1/10, 2010.
13. Asaf Shabtai, "Intrusion Detection for Mobile Devices using the Knowledge-Based, Temporal Abstraction Method", the journal of Systems and Software 83, pp. 1524-1537, Elsevier 2010.