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. 4, Issue 2, February 2016

Android Malware Detection: A Study and Comparison of the Most Promising Techniques

Rahul Balasubramanian[1], Akshay Modi[2], Prof. Dipti Pawade[3]

B. E. Student , Dept. of I.T., K. J. Somaiya College of Engineering, Mumbai, India[1,2]

Assistant Professor, Dept. of I.T., K. J. Somaiya College of Engineering, Mumbai, India[3]

ABSTRACT: The number of Android system users has seen an exponential rise in the past few years. The open source nature of the Android platform and its ever increasing popularity has led to the development of numerous malicious applications aimed at attacking the system. The protection of unsuspecting Android users from such malwares is paramount and hence considerable efforts been put into malware detection. Numerous researchers have put forth some really promising techniques of malware detection in Android. This paper has presented a detailed study of the most promising malware detection techniques thereby aiding Android developers in choosing the most appropriate malware detection technique for their system.

KEYWORDS: Android, Malware detection, Permission based, System call, Malware signature.

I. INTRODUCTION

A malware is a program that infects the user's system and performs hostile operations on behalf of the user without his/her knowledge of the same. The number of android based system users has seen a steady rise in recent years. The fact that Android employs a Linux Kernel and is open source makes it vulnerable to being exploited by individuals with malicious intent. This makes malware detection a very crucial aspect in the evolution of the Android platform. Android facilitates an online market enabling individual developers to build applications to be uploaded to the market. For Android market users, there is no way to verify the credibility of the applications. Certain applications may be genuinely benign, whereas some may be prove to be malicious. Although the system asks for user permissions before proceeding with the installation of any application, the user generally has insufficient knowledge of the implications of granting the mentioned permissions and may simply proceeds with the installation. Once installed on the user's system, the malicious applications may perform certain functions without the user's knowledge. This may lead to sensitive data being leaked from the user's device and unauthorized access of contents through the device without the user even being aware of such a happening. Protection of users from such applications is paramount and hence has seen a lot of efforts being put in to detect malicious applications.

II. CLASSIFICATION OF MALWARE DETECTION TECHNIQUES

Although a number of techniques have been proposed for malware detection, they can be broadly classified into two categories. The methodologies used for android detection are classified as follows:

A. *Signature Based Approach:*

This method relies on known malware code signatures for detecting potential threats. A repository containing known malicious code signature is used as a reference to inspect the application under scrutiny. A match in the signature patterns triggers an alarm notifying the user of the associated threats of the application. This method however relies on human expertise to examine the signatures and hence is prone to human error apart from being time consuming.

B. *Anomaly Based Detection:*

This is a two phase approach involving the learning phase followed by the detection phase. The malware detector creates a set of valid and malicious behaviors exhibited by applications during the training phase. Based on the

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. 4, Issue 2, February 2016

conclusions drawn regarding the behavior of applications during the training phase, the detector tags an application safe or malicious. The detector fails to detect malicious behaviors that were not encountered during the training phase, hence making the system prone to false alarms.

III. MALWARE DETECTION TECHNIQUES

A number of methods for malware detection on the Android platform have been put forth. A comprehensive study of a few proposed methods has been listed below.

Agematsu H. et al. have proposed a method of securing Android applications [5]. This approach involves notifying the user or the security manager whenever an application makes an attempt to run a security related API. The following rule set needs to be imposed in order to implement this method of malware detection:

1. It is mandatory for the operating system to include a security manager.
2. It is mandatory for every application to inform the security manager about all security related events.
3. The android market is authorized to remove all applications violating rule (2).

The onus of notifying the user/security manager regarding security related events may rest with the developer or the Android OS. The latter is a more forcible approach of implementing the proposed idea. This requires the event notification generation codes to be embedded into all the original API's, thus enabling the developer to simply use the API's and the API's take care of notifying the security.To detect malwares using this approach a database referring to the vulnerabilities of permission based events can be maintained. The event checker of the security manager is invoked every time an API is invoked by the application and checks the database if the associated permissions may be associated to some malicious functions. The application manager may either notify the user or terminate the application if a malicious application is detected.

Granting a particular permission to the application may enable it to run a number of related API's. For example, once the user grants the "READ_PHONE_STATE" permission the application may run API's which include "getLine1Number" , "getDeviceId" , "getSimContryIso" , "getVoiceMailNumber" and so on. It becomes very difficult to the ordinary user to determine what features would be extracted once a permission is granted. The proposed system monitors the API calls made by the application to notify the user about the features extracted by the application.

Shuang Liang et al. have proposed a Permission Combination based scheme for detecting malicious Android applications [6]. This approach involves classifying applications based on the combination of permissions requested by the application. The list of permission that the application seeks if first obtained from theAndroidManifest.xml file. The malware detector inspects the requested permissions in groups of k ($k \geqslant 1$). The data set is generated using the permission sets extracted from known malicious and benign applications. Once the data set for malicious as well as benign applications is generated, the application under scrutiny can be analyzed. If the application requests permission sets similar to the malicious samples, it is declared malicious. On the contrary, an application requesting a set of permissions procured most frequently by benign samples is marked safe. It is however difficult to generate a generalized rule set for all applications. Hence rule sets were generated for each malware family and the application being examined is put to test against the rule set of each malware family. The proposed method was put to test for numerous values of k.

The results of the tests conducted by the authors of the proposed method indicate that there is a tradeoff between the hit rate of malicious applications and the pass rate of benign applications with varying values of k. The hit rate decreases with an increase in the value of k (primarily because more permissions need to be requested to be declared malicious). The results are summarized in table 1.The summarized results clearly indicate that the best results are obtained for k=6 (grouping permissions in sets of 6).

Table 1. Relationship between Hit Ratio and K [6]

k	Hit rate (%)	Pass rate (%)
1	96.63	49.38
2	95.82	44.11
3	94.39	55.36
4	94.49	61.35
5	90.51	61.10
6	83.87	87.53

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. 4, Issue 2, February 2016

Suleiman Y. Yerima et al. have proposed a malware detection approach based on Bayesian classification [7]. This method makes uses of statistical analysis to determine the malicious nature of the application under consideration. The application properties are determined using:

1. API detectors-To monitor the API calls being made by the application.
2. Command detector-To monitor libraries and resources that can be used to host malicious scripts or hidden payloads.
3. Permission detectors-To monitor the permissions that the application seeks.

Based on the properties revealed by the above mentioned detectors, a feature vectors are generated for the Bayesian classifier model. Features are ranked based on mutual information in order to ensure selection of most relevant features. The probabilities of the application under scrutiny being benign and malicious are determined using Bayes theorem. An application represented by feature vector r is classified benign if:

P(application being benign for vector r)> P(application being suspicious for vector r)

The results of the tests conducted by the authors of the proposed system indicate that the accuracy of malware detection depends upon the size of the feature set used for evaluating the nature of the application(whether benign or malicious). Results show that the accuracy of the system increases with the increase in the size of the feature including up to 15 features and shows only marginal improvement thereafter. On the contrary the error rates fall with an increase in the size of the feature set.The accuracy and error rates observed for different sized feature sets are summarized in table 2.

Table 2. Summary of Accuracy and Error rates for Different Sized Feature Sets [7]

Feature set size (top features)	Accuracy (%)	Error rate (%)
5	84.5	15.5
10	91.8	8.2
15	92.1	7.9

Takayuki Matsudo et al. have proposed a security advisory system at the time of installation to safeguard users from malicious applications [8]. Users are generally unaware of the consequences of granting certain permissions to the application. Hence a system to warn the user about the nature and the potential threats of the application was proposed. This method, like the previous ones is based on analyzing the permissions requested by the application under examination. The proposed system includes a dataset comprising of sets of permissions frequently requested by malicious applications and a dataset comprising of the sets of permissions frequently requested by benign applications. The system also fetches information regarding the user ratings and the total number of downloads of the concerned application from the Android market. These three factors are used to determine the risk factor associated with the application on a scale of 5.

Since this system intimates the user about the risk associated with the application, the risks associated with the application are categorized as:

1. Stealing Information
2. Stealing Location
3. Automatic Billing
4. Others.

The risk factor calculated for the application is presented to the user along with its category, so as to ensure that the user understands the potential threats of the application. The market reputation including user ratings (based on table 3), total number of downloads (based on table 4) and the user reviews are also presented to aid the user in determining whether to proceed with the installation or not. The results of the initial risk assessment performed by the authors of the proposed system indicates a high rate of correct detection (true positives). However, the rate of incorrect detection of benign applications as malwares is also quite high (false positives). The results are summary is given in table 5.

Table 3.User ratings and their allotted weights [8]

User review rating	0-3	3-4	4-5
Weight (w1)	0	1	2

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. 4, Issue 2, February 2016

Table 4.Number of downloads and their allotted weights [8]

Total Downloads	<1000	1000-50000	>50000
Weight (w2)	0	1	2

Table 5: Result Summary [8]

Rate of true positives (%)	Rate of false positives (%)
95.6	62.1

The results of the initial risk assessment followed by the number of downloads information provides sufficient information to classify the applications as malicious or benign and hence notify the user about the risk level associated with installing the application.

MasoudGhorbanian et al. have proposed a signature-based hybrid intrusion detection system to safeguard Android users from malicious applications [9]. The proposed system aims at detecting intrusions that intend to compromise the system. The system involves four sections:

1. Log file reading
2. Log file analysis
3. Controlling output
4. Storage.

The log files are created for the actions being performed on the system by the applications. The log files are matched with a pre-defined rule set during the log file analysis phase. Different pattern matching algorithms may be employed based on the efficiency of the same. The results of pattern matching performed during the analysis phase by the matching module are provided to the output module. The output module triggers the corresponding alert message if any matching pattern is identified in the log file and rule set for malicious application intrusions or makes an update in the log database if no such match was found.

Lei Cen et al. have proposed a probabilistic discriminative model based on regularized logical regression for malware detection [10]. The proposed method uses the decompiled source code to extract the features of the application. The features of the application may be extracted in any one of the following levels of granularity:

1. Package level
2. Class level
3. Function level.

Once the features are extracted, they are subjected to the feature selection process which uses information gain(IG) and Chi-square test(CHI) to select only the most relevant features out of all the extracted ones. The selected features are then subjected to logical regression for further analysis. Logistic Regression(LR) is a popular classifier as a probabilistic discriminative model. Given a feature vector X and the class label Y ϵ {benign, malicious}, the probability is used to predict the label Y of X as:

$$P(Y \mid X)= \sigma(w^T X+b) \text{ , where } \sigma(a)=(1 + \exp(-a))^{-1}$$

The values w and b are estimated during the learning phase. Gradient descent may be used to determine the maximum likelihood estimate (MLE) of w and b. The fact that a particular feature is favored by malicious samples is indicated by a high positive value of w for the sample. Regularization may be used to avoid the problem of over-fitting. Without regularization the LR model may over-fit the training data which may lead to poor performance with test data. The logical regression model shows the dependence of malicious attacks on individual features of the application, hence enabling detection of suspicious samples based on their features.

The result of the analysis of the proposed method by the authors leads to the following conclusions:

- A finer the level of granularity of extraction of source code features gives better performance.
- In case regularization is applied to the data set, lasso regularization provides the best performance.
- A combination of permission and source code features provides the best results in terms of successful detection.

Iker B et al.have proposed a framework called Crowdroid[1]. It is a machine learning based framework that detects malwares like Trojans on the basis of behavior of the application. Crowdroid is a lightweight client developed by the author. This application uses crowdsourcing philosophy where a user sends non personal but behavior-related data of each application they use to the server. This is followed by malware detection based on the call vectors by the server.The experimental results carried out by the author had 100% detection rate for self written malware.Crowdroid

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. 4, Issue 2, February 2016

was also tested on two real malware specimens: PJApps and HongToutou. The results were that author obtained 100% detection accuracy for PJApps and 85% detection accuracy for HongToutou.

This framework also makes use of k-means clustering algorithm for detection of malware. Clusters are formed as good clusters and bad clusters and malware is detected accordingly.Thus to summarize,Crowdroid is based on three components:

- *Data acquisition***:** where the application monitors the Linux Kernel System calls and sends them preprocessed to a centralized server.
- *Data manipulation***:** where the server does the job of parsing data and creating a system call vector per each instruction of the user within the application, hence creating a dataset of the behavior data for every application used.
- *Malware Analysis and Detection***:**where each dataset is clustered using partition clustering algorithm. Hence, after clustering benign application can be distinguished from the malicious one.

Enck W et al. described an approach 'Kirin' [2] that helps in mitigating malware. Kirin provides lightweight certification to android applications during installations. Kirin provides a method to customize security for production environment, thus supplementing Android's existing security framework by conservatively certifying an application based on its policy configuration. The steps involved in security assessments are:

1. Application installer extracts security information from target package manifest.
2. Kirin security service, against a set of security rules, evaluates the configuration of extracted security information
3. If the configuration fails to pass the rules, then the Kirin service can reject the application.

Kirin relies on the constructed security rules. Defining Kirin security rules requires a thorough knowledge of threats and existing malware mechanism. Security rules of Kirin are defined from the field of requirements engineering which is a part of software engineering. According to the analysis by the author on 311 applications spanning 16 categories the results were: 10 applications were tagged to have dangerous permissions and out of those 5 were potentially malicious.

Khodor H. et al. have described flaws in some message design decision and how an SMS application can exploit these vulnerabilities and have proposed some solutions for the same[3]. Since many operators worldwide provide the facility of credit/unit transfer, the application attacks on these services and transfers the units from the user illegally. The authors have described main features and vulnerabilities of android OS that allows the development and infection of SMS malware. Various solutions are proposed that addresses the following issues:

- The permission that give absolute control and decision of sending and receiving information through messages
- Use of ordered broadcast to hide or modify the SMS payload.

The proposed solutions include:

- Notification/Interruption to user on receipt of SMS message
- The user must grant explicit permission for every SMS sent transaction.
- User's approval at each sending attempt of SMS sending

For receiving SMS messages a proposed solution is to install a trusted SMS application which has the highest priority. As the application has the highest priority whenever a SMS arrives this application will be first to receive the notification and in tern it will notify the user. This ensures that no message remain hidden from the user.To mitigate the vulnerability of sending unauthorized SMS, a technique is used where Android is patched in order to request user's approval at each sending attempt.

Thanh has demonstrated the methods to select a sample of malware families and the methods to analyze them [4]. Various tools and methods are presented that are used to then detect the malware families. The author has listed and described visible symptoms of malware and presented how users can detect those malwares. Three different techniques of analyzing the malware families are presented. The author has selected recognizable characteristics from 58 malware families and 1485 malware samples and proposed solutions so that users can use them before installing the application. All Kungfu (Kungfu A –Kungfu E) malware were downloaded and checked for detection purpose. For ordinary mobile users the author has described a table for malware detection.The description of a few malware families is given in table 6.The complete table can be obtained from[4].

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. 4, Issue 2, February 2016

Table 6.Description about visible symptoms of malware [4]

Families	Visible Symptoms	User's Action
AnserverBot	It makes a new dialog to request and upgrade a new apps but does not show any icon.	You remember new apps name and check show icon on your home screen (request upgrade)
BaseBridge (AdSMS)	Abnormally, high bill to connect internet from data connect or GPRS. 360 Safeguard is installed additional.	Check the regular phone bill. Error message from 360 Safeguard or show 360 Safeguard icon
BeanBot	The device booting up or hanging up on a phone call.	Check the regular phone bill.
Gamblersms	Request provide a phone number and an email address.	View: Phone number and email
VDloader	no corresponding icon in the phone's app	A 3D waterfall wallpaper
Opfake	Its variant have the Opera icon	strange charges to your phone bill
FakeAngry (AnZhu)	Pop-ups displayed Bookmark Name/Bookmark URL.	Appear Screen Off And Lock apps

Gianluca D et al [11] have described MADAM, a Multi-level Anomaly Detector for Android Application, which monitors Android at both user level and kernel level to detect malware infections. MADAM is a Machine learning approach where initially there is a training phase and then followed by detection phase. At kernel level, MADAM monitors System calls. At user level MADAM performs two tasks:

1. Periodically measure the number of SMS sent in the time interval
2. Monitor user's idleness.

The detection rate of MADAM is 93%. A possible drawback of this approach can be that the framework will not identify those malwares that have not been encountered in the training phase.A summarization of the studied malware detection techniques is given in Table 7.

Table 7. Summary of malware detection techniques under consideration

Authors	Proposed malware detection technique	Factors considered for analysis
Gianluca D et al	Machine learning based malware detection technique: MADAM	Systems calls and users interactions
IkerBurguera et al.	Behavior-based malware detection technique: Crowdroid	System calls
Enck W et al.	Certification based technique of Android application: Kirin	Predefined Security rules
KhodorHamandi et al.	SMS malware	Users approval while sending of SMS
Thanh et al.	Detection of Malware families	Common characteristic of different Malware families
Agematsu H. et al.	Detection of malware using Security Manager	Security related API calls
Shuang Liang et al.	Permission-combination based scheme	Permissions
Yerima S.Y et al.	Malware detection approach using Bayesian classification	API calls, Linux system commands and Permissions
Matsudo T. et al.	Advisory system at the time of installing applications	Permissions, Market review and number of downloads
Ghorbanian M et al.	Signature-based hybrid intrusion detection system	Malware signatures
Lei Cen et al.	Probabilistic discriminative model using decompiled source code	Android API calls

IV. CONCLUSION AND FUTURE WORK

The study of the most promising Android malware detection techniques has given us some really decisive information in terms of selection of a malware detection mechanism for the system under consideration. The proposed malware detection techniques make extensive use of data mining and statistical analysis and hence have highlighted the relevance of these two fields to malware detection. With Android malwares continuously evolving, the development of suitable malware detection mechanisms has also become a continuously evolving process. The overheads involved with the studied malware detection techniques need to be minimized to increase the efficiency of the system. One possible method to decrease the computational overhead is by combining signature based analysis and data mining approaches as a two stage approach. The application could be subjected to a signature based test in sage 1 and only those applications that pass stage 1 could be subjected to analysis based on data mining in stage 2. The two stage approach

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. 4, Issue 2, February 2016

could possibly cut down computational complexity in stage 2 as the commonly known malware samples would be filtered in stage 1 itself. The development and analysis of such a system is a very promising prospect and has a lot of scope for future work.

REFERENCES

1. Iker B, Urko Z, Simin N T, "Crowdroid: Behavior-Based Malware Detection System for Android," In SPSM,ACM,October 2011.
2. Enck W, Ontang M, McDaniel P,"On Lightweight Mobile Phone Application Certification,"In CCS Proceedings of 16th ACM conference on computer and communication Security,New York,US,2009.
3. Khodor H, Ali C, Imad H. E, Ayman K, " In Android SMS Malware: Vulnerability and Mitigation", 27th International Conference on Advanced Information Networking and Applications Workshops,2013.
4. Thanh, "Analysis of Malware Families on Android Mobiles: Detection Characteristics Recognizable by Ordinary Phone Users and How to Fix It" Journal of Information Security, Pages 213-224,2013.
5. Agematsu, Kani, Nasaka, Kawabata, Isohara, Takemori, Nishigaki,"A Proposal to Realize the Provision of Secure Android Applications -- ADMS: An Application Development and Management System" in proc. of Sixth International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS) , Pages 677 - 682, 2012.
6. Shuang Liang; Xiaojiang Du, "Permission-combination based scheme for Android mobile malware detection" in IEEE International Conference Communications (ICC), Pages 2301 - 2306, 2014.
7. Yerima S.Y., SezerS.,McWilliams G.,Muttik I. "A New Android Malware Detection Approach Using Bayesian Classification" Advanced Information Networking and Applications (AINA)," IEEE 27th International Conference 2013.
8. Matsudo T., Kodama E., Jiahong Wang,Takata T., "A Proposal of Security Advisory System at the Time of the Installation of Applications on Android OS" 15th International Conference on Network-Based Information Systems (NBiS), Pages 261-267, 2012.
9. Ghorbanian M.; Shanmugam B.; Narayansamy G.; Idris N.B., "Signature-based hybrid Intrusion detection system (HIDS) for android devices" Business Engineering and Industrial Applications Colloquium (BEIAC), Pages 827-831, 2013.
10. Lei Cen, Christoher S. Gates, Luo Si, and Ninghui Li, "A Probabilistic Discriminative Model for Android Malware Detection with Decompiled Source Code," IEEE Transactions on Dependable and Secure Computing, Pages 400-412, 2015.
11. GianlucaD.;Fabio M.; Andrea S.;Daniele S.;" MADAM: a Multi-Level Anomaly Detector forAndroid Malware" Computer Network Security Lecture Notes in Computer Science , Pages 240-253,2012.