International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 2 Issue 6, June - 2013

Malware Detection On Linux Powered Smartphone's Using ID3 Algorithm.

Mr. Rahul Y. Pawar, PG Student D.Y.Patil College of Engg. Akurdi Pune.
Mrs. Deepali Gothawal, Assitant Professor D.Y.Patil College of Engg. Akurdi Pune.

Abstract— **Mobile phones play a very important role in our daily life. It is becoming increasingly important to devolve a reliable and efficient method for securing Smartphone's against unauthorized access. Mobile malwares are causing harm by leaking of user's private information, leaking the battery power, and by automatically generated multimedia messages or making long-distance calls. Smartphone's with open operating systems are getting popular now a days. Increased exposure of open source Smartphone's also increased the security risk. Android is most popular open source operating system for mobile platforms amongst all mobile platforms. Android uses permissions for each application to protect phone resources. Anti-malware software is to play an essential role in defending against mobile malware. The most of the detection software relies on latest aware signature database to detect malware. Since the number of malware designed for Android devices is increasing fast, Android users are expecting some security solutions aimed at preventing malicious actions from damaging their Smartphone's. There are two main techniques of detecting malwares, first is static detection (signature based) and dynamic detection (behavior based).**

Index Terms— **Smartphone, Malware detection, Android operating system, Classifiers, Machine learning algorithm.**

I. INTRODUCTION

In the last few years mobile devices like smart phones, tablets and Personal digital devices, have changed by increasing the number and complexity and capabilities drastically. Current mobile devices offer a large amount of applications and services as compared to personal computers. At the same time, security threats targeting mobile devices have emerged in an increasing number. In 2012, malware attacks increased by 165 percent across all platforms: in case of Android which is the platform with the highest malware growth rate by the end of 2012. Due to this popularity, malicious users and hackers are taking advantage of both the limited capabilities of mobile devices as well as the lack of standard security mechanisms to design mobile-specific malware that access sensitive information, steal the user's phone credit, or deny access to some applications or functions.

The official Google Android market gains almost 100 % increase in size in 2011 and 2012, surpassing 350,000 applications in March 2012. Android developers continue to create applications for Android's OS, malware developers will continue to create Malware for the system, as well. Malware has been a threat for PCs for many years and in light of the rapid increase of Smartphone sales over the last few years, it was only a matter of time before malware developers became interested in staging their attacks on the smartphone platform. In particular, 2011 and 2012 saw a growing interest among malware developers in waging attacks on Android's OS. Malware usually destroys valuable and sensitive information in infected systems. Malware is also commonly used to exploit infected devices and obtain benefits from them. In the same way as malware damages computers, it can also perform attacks on smart phones, given that they have similar operating features. This observation makes it clear that it is necessary to enhance protection of smartphone devices in the same way as we did with computers some years ago.

The Android market is an open market system. This means that Android developers can upload their applications, also called third-party applications, to Android's official market without them being filtered by any certification authority that would check the trustworthiness of the applications. As, there is increase in the odds that the Android market will have a greater variety of applications and content, but it facilitates infection by malware applications, as there is no certification authority for applications to be analyzed.

II. PRILIMINARY

There are three types of threats posed by third-party smartphone applications and discuss the security measures that are intended to detect and prevent them. The mobile threat model includes three types of threats: malware, grayware, and personal spyware. As we are interested in malwares; personal spyware and grayware have different attack vectors, and also have different motivations, they require different defense mechanisms.

A) Malware
Malware is malicious code for the purpose of hacking data, damaging the device, or disturbing the user by gaining access to a device the attacker defrauds the user into installing the malicious application or gains unauthorized remote access by taking advantage of device vulnerability.

B) Personal Spyware
Spyware is specially installed for spying by collecting personal information such as text message or current location. With personal spyware, the attacker gains physical access of the device and installs the software without the user's permissions. Personal spyware sends the private information of victim to the person who installed the spyware onto the victim's device, instead of to the author of the application.

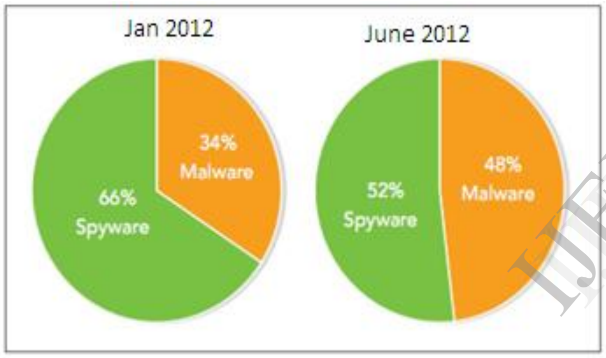

Fig 2.1 : Spyware vs Malwares

C) Grayware
Some legitimate applications collect user data for the purpose of marketing or user prowling. Pieces of grayware provide real functionality and value to the users. The companies used to distribute grayware should disclose their collection habits in their privacy policies, with variation in clarity.

2.1 Malware Detection in Mobile Devices
New security threats emerge against mobile devices as the devices computing power and storage capabilities evolve. It addresses the issue of augmenting current intrusion detection approaches with host-based intrusion detection models for mobile devices. The paper show that host-based approaches are required, only network-based monitoring alone cannot detect encountered future threats. Some of the data types on mobile devices outlined that could be used to construct intrusion detection models.

2.1.1. Smartphone Intrusion Detection:
Several publications in the field of smartphone intrusion and malware detection have been made in the past years where no specific approach prevailed. Promising battery power based mechanisms were introduced in where these approaches depend on the quality and age of the battery. Anomaly and behavior-based approaches are dynamic approaches which are later introduced. These systems suffer from a high computational burden that is moved to an external server in most cases. Rather than these publications, Android gives us permissions to modify the system even at kernel-level. Therefore a light weight application can be developed. so light-weight on device function call analysis is investigated for smartphones.

2.1.2. Intrusion Detection By Static Signature Analysis:
Essential characteristics like system and library functions are extracted and form the basis for identifying malware. Identification is done by data mining classifiers. Static analysis of applications is a well researched technique. Zhang and Reeves propose a static analysis to establish a similarity measure between two executables in order to identify metamorphic malware. Wang, Wu and Hsieh present data mining methods to discriminate between normal executables and malwares, whose dynamically linked libraries and application programming interfaces are statically extracted. Support vector machines technique is used for feature extraction, training, and classification. Eskin apply machine learning methods on a data set of malicious executables.

2.1.3. The andromaly framework
Andromaly a framework for detecting malware on Android mobile devices. The proposed framework uses a Host-based Malware Detection System for continuously monitors various features and events obtained from the mobile device and then applies Machine Learning anomaly detectors to classify the collected data as normal (benign) or abnormal (malicious). Since no malicious applications are yet available for Android, For testing four malicious applications were developed, and evaluated Andromalys ability to detect new malware based on samples of known malware. Several combinations of feature selection method, anomaly detection algorithms, and the number of top features are used in order to and the combination that yields the best performance in detecting new malware on Android. Result of experiments suggests that the proposed framework is effective in detecting malware on mobile devices in general and on Android in particular.

2.2 Collaborative Malware Detection on Android

Here focus is on static and light-weight mechanisms for detecting malware presence on Android devices. Static approach is for detecting malware allows us to use simple classifiers which are not very resource consuming and therefore fit very well to mobile needs. Collaborative malware detection [7] employs collaboration for security approach to extend our Malware detection results. In collaborative approach it is very easy to update the database by coping the malwares dataset.

III. PROBLEM STATEMENT

Personal Digital Assistants (PDAs), mobile phones and recently smartphones have evolved from simple mobile phones compact minicomputers which can connect to the Internet and corporate intranets. Smartphones are designed as programmable, networked devices and susceptible to various threats such as malwares, viruses, Trojan horses, and worms, all are first developed for of desktop platforms. These mobile devices provides facility to users to access and browse the Internet, communication through emails, SMSs, and MMSs, also can be connect to other devices for synchronization and activate various applications, which makes scope of attack on the devices. In addition, most of the static detection systems depend on the existence of an updated malware signature repository, therefore when new malwares encounters at the system, the antivirus users are not protected in such attacks and dynamic detection systems proposed.

3.1 Problem Specification

The challenges for smartphone security are becoming very similar to those that personal computers encounter and common desktop security solutions are often being downsized to mobile devices. As a case in point, analyzed common desktop security solutions and evaluated their applicability to mobile devices. However, some of the desktop solutions (i.e., antivirus software) are inadequate for use on smartphones as they consume too much CPU and memory and might result in rapid draining of the power source. Since the response time of antivirus vendors may vary between several hours to several days to detect and identify the new malware and then by generating new signature, update their client's signature database, in such cases hackers have a slight edge of opportunity. Some malware instances may target a specific and relatively small number of mobile devices (e.g., for extracted confidential information or track owners location) and will therefore take quite a time till they are discovered. The aim of this project is to develop a system at server side that detects malicious applications from the list of android applications. The features are extracted from android phone, then according to the entropy of the feature they are selected as input to the mining algorithm. ID3 algorithm is used for malware detection.

IV. ALGORITHM STARTEGRY

The proposed system uses ID3 Algorithm for malware detection, features are extracted from application level as well as kernel level for better efficiency some feautres are selected from the extracted features. Information gain algorithm is used for feature selection and then inputs are given after transforming the features to the ID3 algorithm.

There are three aspects of Algorithmic strategy:
1. Entropy
2. Information Gain
3. Decision Tree

Decision tree learning is one of the methods used in data mining. The aim of the algorithm is to create a model that predicts the value of a target variable based on several input variables. Each internal node corresponds to the input of the input variables; there are edges which correspond to each of the possible values of each child. Each leaf represents a value of the target variable given the values of the input variables represented by the path from the root to the leaf.

Entropy and Information gain Algorithm

Entropy is used to determine how informative a particular input attribute is about the output attribute for a subset of the training data. In information theory, entropy is a measure of the uncertainty about a source of messages. For example, if a message source always sends exactly the same message, the receiver does not need any information to know what message has been sent. The entropy of such a source is zero: there is no uncertainty at all. Information gain (IG) measures the amount of information in bits about the class prediction. Task of feature selection in these applications is to improve a performance criterion such as accuracy, but often also to minimize the cost associated in producing the features. For Maximum Entropy problem Feature Selection Challenge offered a great test bed for evaluating feature selection algorithms on datasets with a very large number of features as well as relatively few training examples. Due to the large number of the features in the competition datasets, The filtering approach is followed to feature selection: selecting features in a single pass first and then applying an inductive algorithm independently. For selection of features from applications like permissions from whole list of android permissions, Information gain algorithm gives best results amongst other feature selection algorithms.

V. SYSTEM ARCHITECTURE

The proposed system employs decision tree classification techniques for realizing a Malware Detection System. Under such an approach, the malware detector continuously monitors various features and events obtained from the system and then applies standard decision tree classifiers to classify collected observations as either normal (benign) or abnormal (malicious).

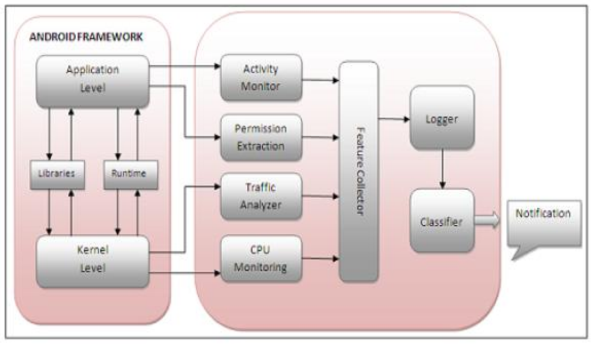

Fig. 5.1 : Block Diagram of Proposed System

5.1 Feature Extraction

In proposed system malwares are detected using Data mining Algorithm. For testing data set, features are extracted from installed applications. In android operating system generally features are clustered in two main categories.

1. Application Framework
2. Linux Kernel

Features belonging to groups such as various system level applications, Messaging, and Phone Calls belong to the Application Framework category and were extracted through APIs provided by the framework, whereas features belonging Linux Kernel category to groups which includes Scheduling, Keyboard, Touch Screen and Memory belong to the Linux Kernel category. Generally the features/group of features based on their availability (i.e., the ability and ease of their extraction) and based on our subjective estimation of the features that will be most helpful in detecting a malware will be selected.

5.1.1. Permissions:

Every APK must include a manifest file that, among other things, requests permission to access certain restricted elements of the Android operating system. These elements include access to various hardware devices (e.g. GPS, camera), sensitive features of the operating system (e.g. contacts), and access to certain exposed parts of other applications. For example, the permission android. Permission.INTERNET requests the right to access the Internet, and android.permission.READ CONTACTS requests the right to access the user's phone contacts database. Once the list of requested permissions extracted, they are divided into two groups: standard built-in permissions and non-standard permissions. For standard permissions, a binary vector generated, where each entry corresponds to a built in permission which is set to 1 if the application requests that permission, and if it has no particular permission it is set to 0.

5.1.2. System calls:

Android is a framework composed by several functional blocks that communicate using the mechanisms provided by the underlying Linux kernel and an increase in the smartphone activity causes directly a sharp increase in the occurrences of these system calls, all of which concern buffer operations, or communications between the framework components. This is why the change in the number of occurrences of these system calls is generally related with the idle nature of user. Hence, for building the training set, we consider as standard vectors those with a low number of occurrences of these system calls and the user idle, and set where the number of system call occurrences is high and the user is active.

5.1.3. SMS Monitoring:

Monitoring SMS usage is semantically difficult through system calls only and SMS messages can be used to harm the user, stealing her credit. Moreover, SMS are strongly related with the user activity. In a normal usage, an SMS is sent after the user has composed the message, which requires an active interaction. However, some applications send or receive SMS to provide some kind of services. Since, SMS is a costly service, if the amount of data that are sent with a message compared, then applications should avoid SMS as communication channel as much as possible, and they must have requiring that the user actively agrees with the sending of each message. Some applications send messages when the user is idle, they can be considered suspicious.

5.1.4. Traffic Analyzer:

In Android framework once the permission to use internet is given user can use the network of mobile phone directly without any restriction. Due to this applications can do misuse the internet by using Internet permission, so it is necessary to monitor the data traffic used by each application. For monitoring the data traffic some kernel level API and PIDs are used and the traffic is monitored through the running application.

5.2 Feature Selection

In Machine Learning applications, feature extraction is done by extracting feature in large numbers, some of which irrelevant or redundant, it may mislead the learning algorithm, overstating, reducing generality, and increasing model complexity and run-time. These negative impacts are even more crucial when machine Learning methods applied on mobile devices, since they are having limited scope by means of storage-capabilities, processing and battery power. Applying fine feature selection in a preparatory stage enabled to use our malware detector more efficiently, with a faster detection cycle. A problem was raised when it had to decide how many features to choose for the classification task from the feature selection algorithms output ranked lists. In order to get proper result by selecting an arbitrary number of Features, Information gain algorithm is used as selection algorithm, which gives highest accuracy amongst other algorithm of feature selection.

5.3 Feature Transformation

Feature transformation is link between features which are selected for generation of log and the input to the ID3 algorithm. When Features are extracted they can be in different format (String, Numbers ,Characters), so it is unable to load the log directly into ID3 algorithms as an input. Here two types of Feature transformation are done.

1. Binary Logic
2. Range Conversion

For some features binary logic is used because there are permissions which should be transformed into Y/N format. If permission for particular activity is given then transform it into 1, otherwise 0. Some features are in Digits so for them range conversion is applied, one range is decided beyond that behavior exploits.

VI. IMPLEMENTATION OF PROPOSED SYSTEM

Google's Android is a comprehensive software framework targeted towards such smart mobile devices (i.e., smartphones, PDAs), and it includes an operating system, a middleware and a set of key applications. Android is growing as an open-source, Community-based framework which provides APIs to most of the software and hardware components. It allows third-party developers specifically to develop their own applications. The application code is developed in the Java programming language based on the APIs provided by the Android Software Development Kit (SDK), but developers can also develop and modify kernel-based functionalities, which is not common for smartphone platforms. The basis of the malware detection process consists of real-time preprocessing, collection, monitoring and analysis of various system metrics, such as Permissions, System calls, CPU consumption, number of sent packets over the internet, number of running processes and battery level. System and usage parameters, changed as a result of specific events, may also be collected (e.g., keyboard/touch screen pressing, application start-up). After collection and preprocessing, the extracted features are sent to analysis by various detection units like processors, each employing its own expertise to detect malicious behavior. The weighting process is per threat type, virus assessments are weighted separately from worm TAs and also includes a smoothing phase (combining the generated alert with the past history of alerts) in order to avoid instantaneous false alarms. After the weighting phase, a proper notification is displayed to the user.

6.1 Android Framework

Android framework will include various libraries which are supportable for running an application on linux kernel. There are two types of features present in android framework which are application level and kernel level. In Android framework various features are taken in to consideration for feature extraction:

- Permissions
- SMS Count
- Traffic Analysis
- CPU usage

6.2 Threat monitoring unit

The Feature Extractors communicate with various components of the Android operating system, including the Linux kernel framework and the Application Framework layer in order to collect feature metrics, while the Feature Monitoring triggers the Feature Extractors and requests new feature measurements every pre-defined time interval. In addition, the Feature Monitoring unit may apply some preprocessing on the raw features that are collected by the Feature Extractors.

6.3 Collector

Collector will rely on main service, it will collect various feature collected from both application level and kernel level of android framework. It will provide input of collected features to feature selection algorithm that is information gain algorithm. Collector will collect permissions from android manifest file, SMS count will be collected from activity monitor, number of Wi-Fi packets are collected through traffic
analyzer. CPU Consumption is collected through CPU usage.

6.4 Loggers

Logger will generate log file which will be sent to server through servlet application. The Loggers provide logging options for debugging, calibration and experimentation with detection algorithms. The Configuration Manager manages the configuration of the application such as active processors, active feature extractors etc.

6.5 Alert Handler

Alert Handler triggers an action as a result of a dispatched alert (e.g. alert in the notification bar, uninstallation of an application by sending notification via SMS or email, disconnecting any communication channels). Alert handler will mainly send notification message after the result of classification algorithm.

VII. EXPERIMENTS

The purpose of the experiments is to evaluate the ability of the proposed malware detection system to distinguish between benign and malicious applications. For each sub experiment 10 datasets extracted from two different devices used, on which decision tree algorithm, algorithm for feature selection evaluated. As presented in table 1, the datasets were collected by activating various applications for 10 min each. The proposed system has been implemented and tested on real devices (Samsung Galaxy and Sony Xperia) to understand the users' experience. The tests have been performed with more than 50 popular applications and several user behaviors to measure the false positives; on the average, a user receives less than 5 false positives per day, and the overall performance overhead is acceptable, i.e. 3 percent of memory consumption, 7 percent of CPU overhead and 5 percent of battery. Each one of the ten datasets contained feature vectors of some benign applications and some malicious applications. All feature vectors in a group were added to the corresponding dataset. The reason for choosing some games and some tools is to guarantee that the different classes are equally represented in each dataset. The feature vectors for the malicious applications were collected and added to each one of the 10 datasets. The dataset is divided the data to training set and testing set as follows: The training set contained all the feature vectors of the some benign applications and some malicious applications collected from the first device. The testing set combination some benign and malicious applications as in the training set collected from the second device which was not used for training. The experiments for each one of the 10 datasets and for each configuration done. Each time the feature vectors of a different device were included in the training set and the feature vectors of the other device were used to produce the testing set.

VIII. RESULTS

In order to calculate accuracy of detection algorithm and feature selection schemes, the following standard metrics employed: True Positive Rate (TPR) measure, which is the Detection Rate in the intrusion detection community classified correctly ; False Positive Rate(FPR) which is the proportion of negative instances or set of applications which misclassified; and Total Accuracy which is proportion of absolutely correctly classified instances, either positive or negative; where, TP is number of positive instances classified correctly; FP is the number of negative instances misclassified; FN is the number of positive instances misclassified; and TN is the number of negative instances classified correctly.

Exp. No.	Device	App.	FPR	TPR	AUC	Accuracy
1	Sony Xperia	40	0.113	0.907	0.895	0.908
2	Samsung Galaxy	37	0.122	0.796	0.837	0.860
3	Sony Xperia	43	0.147	0.913	0.918	0.880
4	Samsung GAlaxy	29	0.231	0.878	0.877	0.828

To reduce the occurrence of false positives, the proposed system has to learn how the user behaves in an early phase, which is called learning phase, where false positives are directly added to the classifier knowledge base without any user intervention. However a new learning phase can be initiated actively by the user when she wants to update the classifier with the generated false positives, for example by a newly installed application.

$$TPR = TP / (TP + FN)$$

$$FPR = FP / (FP + TN)$$

$$\text{Total Accuracy} = TP + TN / TP + TN + FP + FN$$

IX. CONCLUSION

Smart phones are envisioned to provide promising applications and services. At the same time, smart phones are also increasingly becoming the target of malware. A mandatory access control based defense to blocking malware that launch attacks through creating new processes for execution. The security requirements of Smartphone applications and augment the existing Android operating system with a framework to meet them. Secure Application interaction, which would provide security for personal data like SMS, video data, audio data etc. from other applications who can read data present on android phones. To combat more elaborated malware which redirect program owes of normal applications to execute malicious code within a legitimate security domain existing security systems should be updated.

X. FUTURE ENHANCEMENT

Much work is going on in the field of smart phone security; still there is enough scope to do better in this field. Following is list of future work inspired from the proposed work. Other than permissions and Wi-Fi packets features can be extracted and input to the Algorithm can be given. Total phone based application can be done by implementing algorithm on smart phone. System level defense work can be done by changing installation procedure of android applications.

XI. REFERENCES

[1] Adrienne Porter Felt, Matthew Finifter, Erika Chin, Steve Hanna, and David Wagner, "A Survey of Mobile Malware in the Wild", University of California, Berkeley
SPSM, ACM 2011.

[2] A. Shabtai, Y. Fledel, Y. Elovici, \Andromaly: a behavioral malware detection framework for android device", Springer Science+Business Media, LLC 2011.

[3] Asaf Shabtai,\Malware Detection on Mobile Devices", Deutsche Telekom Laboratories at Ben-Gurion University and Department of Information Systems Engineering, Ben-Gurion University, Israel, IEEE 2010.

[4] A. Shabtai, Y. Fledel, Y. Elovici, \Securing Android-Powered Mobile Devices Using SELinux", IEEE Security.

[5] Machigar Ongtang, Stephen McLaughlin, William Enck, \Semantically Rich Application-Centric Security in Android", The Pennsylvania State University,University Park, IEEE, 2009.

[6] Gianpaolo Macario, Marco Torchiano, Massimo Violante , An In-Vehicle Infotainment Software Architecture Based on Google Android", National Tsing Hua University, IEEE, 2009.

[7] Aubrey-Derrick Schmidt, Rainer Bye, \Static Analysis of Executables for Collaborative Malware Detection on Android", Sabanci University, Istanbul, 2009.

[8] David Barrera, P.C. van Oorschot, \Secure Software Installation on Smartphones", Carleton University, November 3, 2010.

[9]Androiddevelopers,
http://developer.android.com/guide/basics/what-is-android.html

[10] Nicolai Kuntze, Roland Rieke, \Secure mobile business information processing", Fraunhofer Institute for Secure Information Technology Darmstadt,germany,2010.

[11] Man Cao, Baixing Quan, Tianzhou Chen, Xueqing Lou, \Construction of Mobile Internet Courses", College of Computer Science Zhejiang University Hangzhou, P.R. China,IEEE 2010.

[12] David Barrera, P.C. van Oorschot, \Secure Software Installation on Smartphones", Carleton University, November 3, 2010.