A Review On Different Techniques Used To Detect The Malicious Applications For Securing The Android Operating System

Priteshkumar Prajapati, Dhruvi Bhagat, Dr. Parth Shah

Abstract: Android is an open source operating system which is mainly used for the smartphones, there are other operating systems also exist for smartphones such as IOS and windows. Majority of the people have the Android operating system and as it is open source malware attacker mainly attacks this system to gain user personal information or to harm the privacy. There are different ways to attack through the malicious application and thus its prevention and detection is the major issue. This paper discusses some of the techniques used to detect malicious applications to secure the Android operating system.

Index Terms: Android Security, Malicious Application, Machine Learning Approaches, T2Droid, Fuzzy Logic, Malware

1 INTRODUCTION

The Android operating system is an open-source operating system and due to this there is much application which is primarily needed is already installed but the ones that are not available are easily available by the third party applications which are Google play store. Majority of the people have the Android operating system and as it is open source malware attacker mainly attacks this system to gain user personal information or to harm the privacy. The mobile application is getting affected while the application is installing or while the user is granting the permission. Android OS covers 75% to 80% of the mobile devices in the market. To protect the privacy of your personal information user first need to activate the authentication mechanism in their Smartphone research says about 29% people do not lock their phones by any authentication mechanism and this can lead to the sensitive data theft [1]. Below are some of the Techniques and approaches to detect and analysis malware in application.

2 LITERATURE SURVEY

2.1 STATIC APPROACH

Nowadays the mobile devices such as smartphones, smartwatches are used more by the people [2]. These devices are having an Android operating system. The Android operating system is an open-source operating system and due to this there is much application which is primarily needed is already installed but the ones that are not available are easily available by the third party applications which are Google play store. Any third party application can be installed from google play store. Android platform is more popular and so there are chances that applications are affected by Android botnet. These botnets are controlled by the botmaster. Botmaster uses this botnet to get the information from the mobile devices as the personal information is more available in the mobile devices compare to pc. The goal of this research is to detect the mobile botnet that affects the mobile application. The mobile application is getting affected while the application is installing or while the user is granting the permission. The approach that is used is the Static detection technique. The static detection technique is used to detect the mobile botnets that affect the security of the Android operating system. Command and control server is the most important components of a mobile botnet. There are four layers of detection technique that are MD5, permissions, broadcast receiver, and background services modules. In this research, they have used 1400 mobiles botnet application and 1400 mobile benign application to test initially. Using this static detection technique, they will develop the lightweight mobile application which will detect the botnet while installing any of the applications.MD5 is the 1st filter test for the application in this there is already the database having some malicious application MD5 will compare the application with the malicious list if it is matched then the test will not pass otherwise the application have passed the test. 2nd is the permissions while installing the application the user grants the permission and upon that the botnets are used. So there are many permissions available the type of the permission is divided on the basis of the risk. These permissions are Normal, Dangerous, Signature and Signature-Or-System. The Android operating system uses broadcast messages to transmit reports to the applications. The combination of dangerous permission that is the INTERNET and the broadcast receiver highly affect the user.3rd is the background services they don't require user interface they run in the background to provide functionality to the user. The botmaster has more interest in these services, as they want to communicate with C & C server of mobile botnets all the time. For the evaluation, they had to use SVM, KNN, J48, Bagging, Naïve Bayes, and Random Forest Machine Classifiers algorithms. The evaluation of the algorithm was performed by measuring the true positive ratio, false positive ratio, and accuracy using the equations. The result shows that the Naïve Bayes classifier algorithm for permissions, while J-48 gives us the slightly better results for broadcast receivers of both benign and botnet and the SVM and Random Forest produced the same percentage of accuracy. The application passes the four layers of the static detection system.

2.2 DETECTION USING FUZZY LOGIC

Authors in paper [3] discuss about third party applications are

- *Priteshkumar Prajapati, K.D. Patel Dept. of IT, CSPIT, CHARUSAT, Changa, Gujarat, India. E-mail: pritesh.pnp.007@gmail.com*
- *Dhruvi Bhagat, K.D. Patel Dept. of IT, CSPIT, CHARUSAT, Changa, Gujarat, India. E-mail: dhruvibhagat226@gmail.com*
- *Dr. Parth Shah from K.D. Patel Dept. of IT, CSPIT, CHARUSAT, Changa, Gujarat, India. E-mail: parthshah.ce@charusat.ac.in*

available in the Google play store which may have security issues. So due to this Android OS security is a break. Here we will improve the effectiveness of the malicious software detection system. The method to detect the malicious software based on its behaviour. In this research, the malicious software sample was taken where both malicious software as well as secure software behaviour was described and developed. For the experiments, classical, neural network and support vector machine methods were carried out. For neural network method activation function were chosen and training for the same was conducted. For the core support vector machine Radial basis function was chosen. Authors represented the sample which has the 100 vectors of software behaviour. Neural network and support vector machines have been started on 68 and tested on the other 32 values of the vectors of the experiment. A finite experimental sample of objects assigned Xm and requires to break the sample into the disjoint subset and these subsets are called clusters. And this number of the cluster should be assigned with a function i.e. a:x-> y. Eventually, problem is to define a set with an optimal number of a cluster according to clustering quality. The experiment which was conducted was based on the support vector machine. Fuzzy logic was used as the method was against a high interference level which includes misclassification. The experimental sample includes 67 software vector and 33 software vector was tested sample with modification. Some additional features are added in the form of the membership function for fuzzy logic there are total 6 steps which need to be followed for the malicious software detection. In result of the best one was about 60% and on average 23, 4% among all tested examples. These methods improve the effectiveness of the detection through behavioural character.

2.3 MACHINE LEARNING APPROACH

In paper [4] 4th quarter of 2015, the worldwide stack of Android was 80.7%, IOS 17.7%, and Microsoft 1.1%. Android is an open source OS mainly used for the mobile purpose, in android architecture, it has the main four components activities, services, content providers, and a broadcast receiver. Malware is of different types such as a virus, worms, and Trojan which will help malware writer to get confidential information of the user. In this machine learning approach is used to detect the malware in that three methods are used name supervised, semi-supervised and unsupervised. Taint Droid helps to track and detect the malware by using third-party application flow of sensitive information. In supervised pre-labelled data is used to train and the annotated data is read by the system and the result comes in true positive rate which is 0.90 and false positive rate 0.23 with the accuracy of 0.83. In semi-supervised learning, they have used the CHABADA and extended WHYPER and it is precise up to 95.41% but using the F-measure it is more precise than the CHABADA by 209.6%. In unsupervised learning they used the clustering and KNN algorithm is implemented to detect the malware. All in all, unsupervised learning gives you the best result with the accuracy, precision, and recall without any human effort as there is no data required to collect.

2.4 ENERGY-BASED ATTACKS THROUGH HTML5

As per paper [5]; Attackers do not only attack the Android application through malicious application to gain the user data but they can also affect them through the energy based attack in this attack the battery of the smartphone is effected by using the energy-based denial of service attack. This attack does not require any type of physical access the HTML5 Functionalities is used to trigger this attack through the normal web browsing and thus it makes easy to drain the battery and it can also affect the battery life. if any audio file which is malicious is played in the web browser in the background in the loop then that malicious audio file will run and consume the battery and the draining of the battery becomes faster compare to the previous one. Every component of the smartphone needs the particular power that it utilizes from the battery, if that power doesn't get to that components then that component may don't work so the energy based attack is harmful, Basically, this attack is done by the multimedia files while browsing over the web.

2.5 BIMODAL BEHAVIORAL BASED AUTHENTICATION

Nowadays there are physiological biometrics such as face and iris and others calculated based on behavioural biometrics parameters such as gesture, keystrokes, and sensors. Bimodal behavioural biometric authentication [6] framework that provides a high accuracy detection performance to check whether the user is authenticated or not. here study has been made on the data given by the 52 participants for the 30 days and according to their behaviour, they contain two modalities for the authentication that are touch gesture and keystrokes modalities. In touch gestures pattern are recognized in this parameters such as touch-up, touch-down, pressure point is measured and they also have the coordinates while in keystroke the user keyboard rhythm is recorded and according to that analysis is done and every user has different pattern while typing. This active authentication framework has two main phases enrolment phase and recognition phase, in enrolment phase data are collected and analyse and then added to feature template but in recognition phase, they are not added to the feature template instead they are compared directly with the saved user identities. In this two models are used classification and anomaly detection model well, in classification they are trained on data of legitimate user and imposters while in other only legitimate user data are trained. As a result of this analysis to enhance decision accuracy they leverage decision fusion method. In terms of accuracy and error rate, this model outperforms in terms of a unimodal based system.

2.6 ANDROID PERMISSION BASED SOLUTION

As per paper [7], Market share of Android in 2016's first quarter was 84.1% while IOS was at 14.8% and windows having 0.7% apart from this other where 0.2%. Well, Android Operating System is permission-based model in which the user is asked to grant the permission which is needed for using the application and while granting the permission it becomes easy for the malware writer to get personal information and data easily and thus the security is needed for the Android application, this is done as application have same SHARED USER ID with the same certificate. Due to this many attacks take place such as permission escalation attack (In this malicious apps merges with other apps to get information), confused deputy attack, direct collision attack (It occurs due to same shared user id and certificate of two applications and occurs when application communicates directly), indirect collision attack (application communicated to third-party application) and TOCTOU (Time of Check and Time of Use which mainly occurs due to naming constraints) attack. Issues occurring due to android permission after the 6th version the permission are classified into two parts that are a normal permission and dangerous permission. In normal permission, it includes by default permission such as

KILL_BACKGROUND_PROCESSES and SET_WALLPAPER. In dangerous permission, it includes permission such as READ_CONTACTS, WRITE_CONTACTS, and ACCESS_FINE_LOCATION. Well, we consider the two-factor authentication as secure one but it is not the same all the time the OTP comes as the SMS text message but what if any malicious code is installed on your device and it helps the attacker to get the code so, it is also unsafe. Compared to Android, IOS is more secure as firstly IOS doesn't allow unknown sources application you can download the app from the App store only and while signing technology Android follows self-signing while in IOS you first have to apply and the Apple INC. will check your whole application as well as code and then give you verification and even after that if any changes in the code occur it will be tracked by them. So, to have a secure Android application while giving permission some steps should be taken to avoid the privacy issue as well as data theft. If your data are with the owner of application sometimes they make money by selling the data.

2.7 NETWORK TRAFFIC BASED ANALYSIS

The approach in [8], device type verification done through analysis of the network behaviour, digital forensics techniques involve networks which often depends on data extraction from targeted devices. In fact, MAC address it can also easily have spoofed and tools like Nmap and xProbe can able to identify OS type using active network probing and used to scan the networks. In this, they have conducted 200 experiments in which Android device is pinged 100 times in this average inter-packet spacing is calculated for the unloaded device and loaded device. while to note that during the critical data transferred no unauthorized device have access to that network. While giving permission to access the network interrogate the device for its identity and after that go further for its identity. Device identification is the main concern in the field of network security and digital forensics. This experiment is designed to examine the capability of our approach in actively and passively discerning with three different OS that is IOS, Windows, and Android. In this experiment wireless network is connected to three mobile devices and two network cards, A laptop with Ubuntu 14.04.3 LTS with two network cards that are NIC1 and NIC2 while the devices are iPhone 6S running IOS 9.1, Samsung S6 Edge which Android 5.1 and Nokia Lumia 521 running Windows 8.10 then laptop is pinged to mobile devices then thousand times and ICMP responses using Wireshark in active experiment while in the passive one, NIC2 was placed on monitor mode as well as connected to router wirelessly per mobile devices 250000 TCP/IP network packets are captured. Machine learning algorithm is used to demonstrate can be trained to accurately and dynamically identify a large number of mobile devices. this hypothesis for the driver of this behaviour is the power saving mechanism and the collected data can be used to build an accurate dynamic classifier for mobile OS detection.

2.8 MUTATION ANALYSIS BASED TESTING

As discussed in paper [9], Quality of the application is the ongoing problem here as there required some faults with the release time only there are many faults occurs due to the system changing and thus the testing techniques which are old techniques and thus we need some new techniques. Some of the testing problems are there and that can be solved by the techniques such as Android mutation testing and other four techniques are there, in this two sets of faults are taken and they are mined from our subject apps and source code repositories. some of the testing, problems are unique lifecycle of the Android components intensive use of XML, two types of screen orientation, varied network connections, limited battery life and common faults such as null pointer exception made by most of the user. In Android, mutation technique there are six new operators named service lifecycle method deletion(SMDL), Text view deletion(TVD), Fail on back(FOB), Location modification(LCM), Wake Lock Release Deletion(WRD) and Wi-Fi connection disabling(WCD). In this nine experiment subject apps, android mutation is extended by Java by implementing all mutation operators after those non-Equivalent mutants were identified and killed and then mutation test set against each fault and recorded whether it is detected or not. Other testing techniques are Monkey, Dyno droid, PUMA, and A3E from the result of the natural occurring faults the Android mutation detected 18, monkey 6, Dyno droid 7, PUMA 3 and A3E 1 out of 25 faults and from the hand-seeded faults the Android mutation detected 360, monkey 130, Dyno droid 138, PUMA 121 and A3E 79 out of 437 faults.

2.9 ANDROID PERMISSION USING PETRI NETS

To provide security while cross-application communication and is complex by nature. so using high-level Petri nets analysis is done for the android permission framework in paper [10]. this framework has the multiple levels of the permission such as overall application-level permission and component level permission, while component level permission is denoted by Cperm. petri net is a model which is used for the modelling and analysis of Android permission and it is a graphical representation of modelling states, it also has high-level features such as data processing and defining, one of the forms of the high-level Petri nets is predicate transition nets and it is divided into the static semantics and dynamic semantics after this a model is prepared corresponding to the permission related events. firstly, the model is prepared according to the different permission level mentioned in the android manifest file. then they will find the permission related events content access and component invocation, while the further steps include modelling installation and un-installation, application execution, component invocation, data access, dynamic permission granting and revoking, complete permission schema and finally creating the model in PIPE+ Environment. PIPE+ tool has many external tools with checking abilities but due to the complexity of the nested qualifiers, it doesn't work properly. The two main goals in this are firstly correctness of the individual transition constraints as well as the detecting the behaviour of the permission related security vulnerabilities. In the potential permission violation, it checks the user declared permission name and group name conflict and the improper delegation. From this, they also found that the dynamic permission granting helps to limit the scope and duration of a potential security violation.

2.10 T2DROID- APPLICATION ANALYZER AND OTHER

Some of the Android application is malicious and thus to detect them various techniques are used in paper [11]; here dynamic analysis methods are used to detect it by evaluating the behaviour of that application during the runtime. T2Droid (Trust Zone based trace-analyser for android application) is the analyser that will help to detect the malware application using the Android API function call traces and kernel syscall traces.

There are three main test cases of the T2Droid, the first time the application is executed according to the number of API calls. The second time when requested by the backend of the application and third time while targeted especially at the testing of the application. T2droid runs in the ARM processor with the trusted zone which makes the design of the T2Droid more secure. to detect the machine learning classifier is essential and there is 3 life cycle of the classifier, first selecting and training, second using the classifier at the runtime and the third one re-training the classifier. With the help of KNN classifier, the T2Droid achieved the accuracy of 0.98 and precision of 0.99. Apart from this DDOS attacks discussed in paper [12] are also possible in android operating system.

3 CONCLUSION

Here from the above study, we came to know that Android uses different approaches for the detection of the malware from the application. This techniques and approaches are best in their own way but the best detection and testing approach for the Android security is T2Droid as it gives the Accuracy about 0.98 and precision about 0.99 and it helps to analyze the application and can work as the testing purpose for the application. This approach is useful as T2Droid uses API calls and syscall for the process.

4 REFERENCES

[1] Egelman, Serge, Sakshi Jain, Rebecca S. Portnoff, Kerwell Liao, Sunny Consolvo, and David Wagner. "Are you ready to lock?." In Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security, pp. 750-761. ACM, 2014.

[2] Anwar, Shahid, Jasni Mohamad Zain, Zakira Inayat, Riaz Ul Haq, Ahmad Karim, and Aws Naser Jabir. "A static approach towards mobile botnet detection." In Electronic Design (ICED), 2016 3rd International Conference on, pp. 563-567. IEEE, 2016

[3] Zhernakov, S. V., and G. N. Gavrilov. "Malicious software detection in operating system (OS) for mobile devices (the case of Android OS)." In Actual Problems of Electronics Instrument Engineering (APEIE), 2016 13th International Scientific-Technical Conference on, vol. 2, pp. 163-165. IEEE, 2016.

[4] Jamil, Qudsia, and Munam Ali Shah. "Analysis of machine learning solutions to detect malware in android." In Innovative Computing Technology (INTECH), 2016 Sixth International Conference on, pp. 226-232. IEEE, 2016.

[5] Fiore, Ugo, Aniello Castiglione, Alfredo De Santis, and Francesco Palmieri. "Exploiting battery-drain vulnerabilities in mobile smart devices." IEEE Transactions on Sustainable Computing 2, no. 2 (2017): 90-99.

[6] Mahfouz, Ahmed, Tarek M. Mahmoud, and Ahmed Sharaf Eldin. "Bimodal behavioral authentication framework based on decision fusion." In Information and Communication Systems (ICICS), 2017 8th International Conference on, pp. 368-373. IEEE, 2017.

[7] Karthick, S., and Sumitra Binu. "Android security issues and solutions." In Innovative Mechanisms for Industry Applications (ICIMIA), 2017 International Conference on, pp. 686-689. IEEE, 2017.

[8] Malik, Nikunj, Jayanarayan Chandramouli, Prahlad Suresh, Kevin Fairbanks, Lanier Watkins, and William H. Robinson. "Using network traffic to verify mobile device forensic artifacts." In Consumer Communications & Networking Conference (CCNC), 2017 14th IEEE Annual, pp. 114-119. IEEE, 2017.

[9] Deng, Lin, Jeff Offutt, and David Samudio. "Is Mutation Analysis Effective at Testing Android Apps?" In Software Quality, Reliability and Security (QRS), 2017 IEEE International Conference on, pp. 86-93. IEEE, 2017.

[10] He, Xudong. "Modeling and Analyzing the Android Permission Framework Using High Level Petri Nets." In Software Quality, Reliability and Security (QRS), 2017 IEEE International Conference on, pp. 232-239. IEEE, 2017.

[11] Yalew, Sileshi Demesie, Gerald Q. Maguire, Seif Haridi, and Miguel Correia. "T2Droid: A TrustZone-based dynamic analyser for Android applications." In Trustcom/BigDataSE/ICESS, 2017 IEEE, pp. 240-247. IEEE, 2017.

[12] Prajapati, Priteshkumar, Patel, Nidhi and Shah, Parth. “A Review of Recent Detection Methods for HTTP DDoS Attacks." International Journal of Scientific & Technology Research, 8(12), pp. 1693-1696. IJSTR, 2019.