Malware Defense in Mobile Network using Dynamic Analysis of Android Application

Miss. Ashwini A. Dongre
M. E. 4th sem, Dept. of Computer Science and engineering
P. R. Patil College of engineering Amravati, India
Coolashwini.dongre@gmail.com

Prof. C. J. Shelke
Prof. Dept. of Computer Science and engineering
P. R. Patil College of engineering Amravati, India
Chetanshelke7@gmail.com

***Abstract*—** Today Android has the biggest market share as compared to other operating system for smart phone. As users are continuously increasing day by day the Security is one of the main concerns for Smartphone users. As the features and power of Smartphone are increase, so that they has their vulnerability for attacks by Malwares. But the android is the operating system which is more secure than any other operating systems available for Smart phones. The Android operating system has very few restrictions for developers and it will increase the security risk for end users. I am proposing an android application which is able to perform dynamic analysis on android program. To perform this analysis i have to deploy the android application, In this proposed system I am going to deploy android application on a cloud. This application executes automatically without any human interaction. It automatically detects malware by using pattern matching algorithm. If malware get detected then user get inform that particular application is malicious and restrict the user from installing application.

Keyword : *Android, vulnerability, malwares, smart phones*

*****

I. INTRODUCTION

Today world is changing from the Internet world to a mobile world where more and more access to information is done by previously dumb phones. In an interconnected mobile world, the interactions among mobile devices, systems, and people are growing rapidly. At the same time people are more and more concerned about the security issues and fast transmission of sensitive digital information over wireless channels. The security issues include issues such as the quick spread of viruses and malicious of software. According to F-Secure [4], there are more than 200 mobile viruses or malware programs are causing problems to the system. Also low computational power is a major issue in mobile system. Many organizations are increasingly interested in deploying mobile application to enhance productivity and enable new capabilities.

Malwares (e.g. virus, worms and Trojan horses) have been threats to computer systems for many years and it was only a question of time when the first malicious software writers would get interested in increasingly popular mobile platforms, such as Symbian OS. In 2004, the first articles about malware for smartphones [2,3] appeared saying that the next generation of targets are mobile devices. Since then, the number of malwares increased every month, and variants for various smartphone platforms appeared.

Smartphones adoption is rapidly increasing which is directly linked to the improved computational power and other utility factors. Garter states that [5,6], Sales of Mobile devices grew 5.6 percent in Third Quarter of 2011 whereas smartphones sales increased 42 percent. Android OS account for more than 50 percent of smartphones sales. In the Modern day sophisticated mobile phones have three capabilities – communication, computing, and sensing. So these capabilities provide useful service to the users, they also open up serious security and privacy concerns. The sales of such smartphones soar worldwide, the stage is set for the massive spread of mobile malware.

II. LITERATURE REVIEW

So far two approaches have been proposed for the analysis and detection of malware: static analysis [7, 13] and dynamic analysis [10, 8, 12]. Static analysis, mostly used by antivirus companies, is based on source code or binaries inspection looking at suspicious patterns. Although some approaches have been successful, the malware authors have developed various obfuscation techniques especially e_ective against static analysis [11]. On the other hand, dynamic analysis or behavior-based detection involves running the sample in a controlled and isolated environment in order to analyze its execution traces. Egele [9] provides a complete overview of automated dynamic malware analysis techniques

David Dagon et al. alerted the community in 2004 predicting the feasibility of malware in mobile phones [14]. Even if wi-fi and Bluetooth were considered as the most probable infection paths, the growth of smartphone sales with continuous Internet connectivity made the prediction come true. Concretely, in June of the same year, the first malware specially written for Symbian OS platform was discovered [15]. After the infection success carried out by Cabir malware and its variants [19], researchers proposed approaches and

developed different mechanisms in order to detect malware in smartphones.

Due to the lack of smartphone malware patterns by that time, most of anomaly detection techniques used the battery power consumption as the main malware detection system feature [16,17,18]. These techniques were based on checking and monitoring mobile phones power consumption and comparing them with the normal power consumption pattern to detect anomalies. These techniques are specially designed to detect attacks targeting battery life.

Schmidt et al[20]. employ static analysis on executables to extract their function calls using the *readelf* command. They then compare these function call lists with those from Linux malware executable in order to classifying the executables using learning algorithms. In contrast, our static analysis approach is based on automated analyses of Android packages. Moreover, Android malware samples across a range of existing families are employed in our work rather than Linux malware executables. Other earlier non-Android based papers have explored data mining and machine learning techniques for malware identification including for example [21] and [23].

Blasing et al[22]. Presented an Android Application Sandbox (AAS) that uses both static and dynamic analyses on Android applications to automatically detect suspicious applications. Compared to AAS, our approach covers a much wider range of pattern attributes extracted not only from the application code logic but also scrutiny of resources, assets and executable libraries where malicious payload could be lurking. Additionally, these attributes contribute to a ranked feature set which drives our Bayesian classification model. Apvrille and Strazzere employ a heuristics approach based on static analysis for Android malware detection. Their heuristic engine uses 39 different flags and then outputs a risk score to highlight the most likely malicious sample. Our approach shares similarity in the reverse engineering technique, but differs by utilizing a machine learning based method that offers more flexibility.

W. Enck, D. Octeau, P. McDaniel and S. Chaudhuri presented review paper on "a study of Android application security". Introduces the ded decompiler which generate android application source code directly from its installation image. Also they design and execute a horizontal study of smart phone applications based on static analysis of 21 million lines of recovered code. This analysis uncovered pervasive use, misuse of personal or phone identifiers and deep penetration of advertizing and analytics networks [24].

D. Barrera, H. Güne¸ S. Kayacık, P.C. van Oorschot, A. Somayaji discuss on 'a methodology for empirical analysis of permission-based security models and its application to android'. According to paper, the proposed methodology is of independent interest for visualization of permission based systems beyond current Android-specific empirical analysis. Authors provide some discussion identifying potential points of improvement for the android permission model, trying to augment quality where required without increasing number variety of permissions or overall difficulty [25].

C. Gibler, J. Crussell, J. Erickson and H. chen case learn on 'AndroidLeaks: automatically detecting potential privacy leaks in android applications on a large scale'. Under this published, they have presented a static analysis framework for automatically finding potential leaks of sensitive data in android applications on a large scale. AndroidLeaks severely reduces the number of applications and the number of traces that a security auditor must verify manually [26]

III. SYSTEM IMPLEMENTATION AND WORKING

As we know that, the security of smart phone grows rapidly because new smart phones are launches in the market like-wise the android malware authors are also introduce new malwares which are harmful to mobile network. To detect that harmful malware there are many systems which are based on the permission called permission based malware detection systems but some of them not detected all the malware present in the system and some time it was detect such import permission which are helpful to run the app so the system get failed to detect malware. To overcome such type of problems I introduced a system which is signature based detection system. Following figure shows the System Architecture

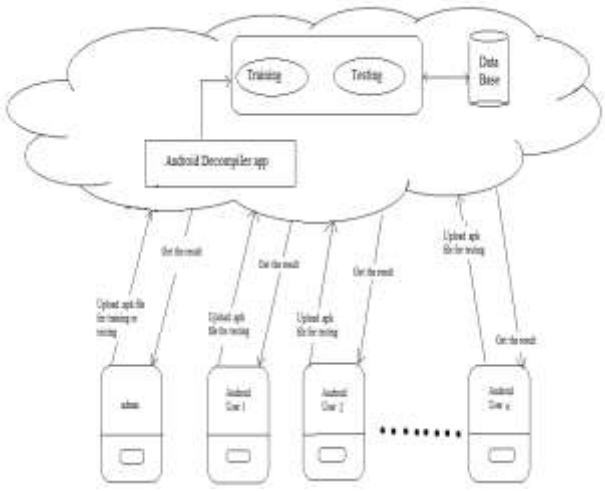

Fig. 1. System Architecture

1.Android application for uploading the .apk file on the cloud

The first module of a system is an android application which will install on a users smart Phone. In our System as the apk files are decompile on the cloud So, we requires to upload apk file on a cloud. This application is used select and upload a apk file to cloud.

This Android application provides Option like Training and testing once we click on evaluate apk button it will uploads a apk file to a cloud for the selected purpose.

2 . Android decompiler app on cloud.

The second module of a system is android decompiler application which is at cloud.

This application is used to decompile the apk file which is uploaded by the user via android application. This application converts apk file to the java file i.e. performs reveres engineering on a APK files.

3.Training and evaluating app on cloud.

a) at the time of training we have to select the option training and upload the .apk file on the cloud, it get decompile and form the signatures with the help of decompiler application. By using Hmm model we can train our database by adding signature in to the my sql database which is on the cloud.

b) at the time of evaluation we have to upload .apk file on cloud by selecting a testing option, then app get decompile with the help of decompiler app and match the signatures which are present in the database with the help of HMM model and gives the answer on the user monitor about the type of the app.

HMM Model

The System uses the HMM model for Pattern matching to detect the Malicious Signatures. A statistical model that has states and known probabilities of the state transitions is called a Markov model . In such a Markov model, the states are visible to the observer. In contrast, a hidden Markov model (HMM) has states that are not directly observable . HMM is a machine learning technique. HMM acts as a state machine. Every state is associated with a probability distribution for observing a set of observation symbols. The transition between the states has fixed probabilities. We can train an HMM using the observation sequences to represent a set of data . We can match an observation sequence against a trained HMM to determine the probability of seeing such a sequence. If the probability is high, the observation sequence is similar to the training sequences. HMMs are used in protein modeling . HMM can also be used to detect certain types of software piracy detection .

Working

The user will try to install an application to his/her device at that time If the user is administrator then he/she having a options as training and testing.User has to select the option, if user selects option as training i.e. malware app or spyware or safe app then he/she has to clicks on the upload apk button and select a file for uploading to a cloud .then system shows the message that wait while uploading file.

Once the file uploaded to the cloud, .apk file is pass to decompilation app to perform reverse engineering i.e. converts .apk file to java code then it forms the signature which are store in the database .

If user want to evaluate the file then he/she simply select evaluate the .apk and click on upload apk button and select file which we wants to check . on the cloud it decompile first and then perform pattern with the signatures present in database once the testing is done user can get the result on his/her android application.

IV. RESULT ANALYSIS

1. Battery power consumption

As we know that any application that runs on a Smart Phones Should consume some amount of battery power depending on its activities. Most of the android malware detection applications are runs all time in a background, so it will consumes more battery power.

In our System, cloud is used as the major weapon, all the processes like decompilation of a apk file, Training the databse, Testing a new APK file via Pattern Matching of signatures after uploading are done on the cloud so no one process run on the background. Hence it takes less battery of our smartphones.

2. Less Storage space

Android applications which will installed on a smart phones requires different Storage space to for storing different important files required by these applications. The different malware detection applications available requires certain amount of storage space to stores the signatures or information about the malwares, as we know that new malwares are introduced daily so we have to update the database of a installed malware application. Once we update the database it will increases the size of the application, so the required space by application becomes increasing and it will causes the availability of storage space for the user.

To overcome such problems, In this system we maintain the database of the signatures on a cloud. That means we can update the application database by training on a cloud, so it will minimizes the storage space of the application.

3. Centralized database

As project uses the cloud as a server and it contain database of all the signatures while training. And for the testing purpose we use the database for pattern matching so all users access that the remote server from different places . hence database is centralized.

Table 1. Result analysis with existing systems.

Cases	Avast Antivirus	Avg Antivirus	Implemented system
Storage space on device	12.8 mb	27 mb	1.28 mb
Battery consumption	7 %	6%	2%
Database	Database is on device	Database is on device	Database is on cloud
Ram	30 mb	22 mb	17.74 mb
Updation of database	User has to install	User has to install	Admin has to install
Decompilation od app	No	No	Yes
Background playing	Yes	Yes	No
Centralized database	No (only for one user)	No (only for one user)	Yes – available for all user

V. CONCLUSION AND FUTURE SCOPE

Thus I introduced a security service for Smartphone's, which off loads the detection of malicious applications from the Smartphone into the cloud. As Smartphone's are very much prone to malwares hence we introduces new approach of using cloud as a security weapon for providing security. I proposed a system that detects the malicious code and stores that malware into database which is on the cloud and then report to user who wants to installed that application on his / her device.

As I proposed the above system, which detect the malicious code from new application and stores the malware in the database of cloud. But in future try to make a self replicate program which will report about malware without database.

REFERENCES

[1] Burguera I., Zurutuza U. and Tehrani S.N., Crowdroid: behaviour-based malware detection system for Android, 1st ACM workshop on Security and privacy in smartphones and mobile devices, 15-26 (2011)

[2] D. Dagon, T. Martin, and T. Starner, "Mobile phones as computing devices: The viruses are coming!" IEEE Pervasive Computing, vol. 3, no. 4, pp. 11–15, 2004.

[3] M. Piercy, "Embedded devices next on the virus target list," IEE Electronics Systems and Software, vol. 2, pp. 42–43, Dec.-Jan. 2004

[4] F-secure, "New Century in Mobile Malware," F-secure, 2006. [Online]

[5] Gartner Press Release, Egham, UK, November 15,2011http://www.gartner.co it/ page.jsp? id=1848 514

[6] McAfee Labs Q3 2011 Threats Report Press Release, 2011 http://www.mcafee.Com/us/about/news/2011/q4/201111 21-01.aspx

[7] Mihai Christodorescu and Somesh Jha. Static analysis of executables to detect malicious patterns. In Proceedings of the 12th conference on USENIX Security Symposium - Volume 12, pages 12{12,Berkeley, CA, USA, 2003. USENIX Association.

[8] Mihai Christodorescu, Somesh Jha, and Christopher Kruegel. Mining speci cations ofmalicious behavior. In Proceedings of the the 6th joint meeting of the European software engineering conference and the ACM SIGSOFT symposium on The foundations of software engineering, ESEC-FSE '07, pages 5{14, New York, NY, USA, 2007. ACM.

[9] Manuel Egele. A survey on automated dynamic malware analysis techniques and tools. ACM Computing Surveys, to appear

[10] T. J Lee and J.J. Mody. Behavioral classi_cation. In Proceedings of EICAR 2006, April 2006.

[11] Andreas Moser, Christopher Kruegel, and Engin Kirda. Limits of static analysis for malware detection. In Proceedings of the 23rd Annual Computer Security Applications Conference, ACSAC'07, pages 421{430,

[12] Konrad Rieck, Thorsten Holz, Carsten Willems, Patrick D□ussel, and Pavel Laskov. Learning and classi_cation of malware behavior. In Proceedings of the 5th international conference on Detection of Intrusions and Malware, and Vulnerability Assessment, DIMVA '08, pages 108{125, Berlin, Heidelberg, 2008. Springer-Verlag.

[13] Asaf Shabtai, Robert Moskovitch, Yuval Elovici, and Chanan Glezer. Detection of malicious code by applying machine learning classi_ers on static features: A state-of-the-art survey. Inf. Secur. Tech. Rep., 14:16{29, February 2009.

[14] David Dagon, Tom Martin, and Thad Starner. Mobile phones as computing devices: The viruses are coming! IEEE Pervasive Computing, 3:11-15, October 2004.

[15] Cabir, Smartphone Malware. http://www.f-secure.com/v-descs/cabir.shtml.

[16] G A Jacoby and Nathaniel J Davis Iv. Battery-based intrusion detection. In Global Telecommunications Conference, 2004. GLOBECOM '04, pages 2250 { 2255. IEEE, 2004.

[17] Hahnsang Kim, Joshua Smith, and Kang G. Shin. Detecting energy-greedy anomalies and mobile malware variants. In Proceeding of the 6th international conference on Mobile systems, applications, and services, MobiSys '08, pages 239{252, New York, NY, USA, 2008. ACM.

[18] Timothy K. Buennemeyer, Theresa M. Nelson, Lee M. Clagett, John P. Dunning, Randy C. Marchany, and Joseph G. Tront. Mobile device pro_ling and intrusion detection using smart batteries. In Proceedings of the Proceedings of the 41st Annual Hawaii International Conference on System Sciences, HICSS '08, pages 296{, Washington, DC, USA, 2008. IEEE Computer Society.

[19] A. Schmidt and S. Albayrak. Malicious software for smart-phones. Technical Report TUB-DAI 02/08-01, Technis-che Universit¨at Berlin, DAI-Labor, Feb. 2008. http: //www.dai-labor.de

[20] A D Schmidt. "Detection of Smartphone Malware", PhD thesis, Technischen Universit¨at Berlin, 2011.

[21] Thomas Bl¨asing, Aubrey-Derrick Schmidt, Leonid Batyuk, Seyit A. Camtepe, and Sahin Albayrak. "An android application sandbox system for suspicious software detection", in 5th International Conference on Malicious and Unwanted Software (Malware 2010) (MALWARE'2010), Nancy, France, France.

[22] J. Bergeron, M. Debbabi, J. Desharnais, M. M. Erhioui, Y. Lavoie, and N. Tawbi. "Static detection of malicious code in executable programs", In Proceedings of the Symposium on Requirements Engineering for Information Security (SREIS'01), 2001.

[23] Enck W., Octeau D., McDaniel P. and Chaudhuri S., A Study of Android Application Security, The 20th USENIX conference on Security, 21-21, (2011)

[24] Barrera D., Güne H., Kayacık S., Oorschot P.C. van and Somayaji A., A Methodology for Empirical Analysis of Permission-Based Security Models and its Application to Android, 17th ACM conference on Computer and communications security, 73–84 (2010)

[25] Gibler C., Crussell J., Erickson J. and Chen H., Android Leaks: Automatically Detecting Potential Privacy Leaks In Android Applications on a Large Scale, 5th international conference on Trust and Trustworthy Computing, 291-307 (2012)

[26] http://www.ijritcc.org/browse/volume-2-issues/nov-14-volume-2-issue-11/25 "Review of Malware Defense in Mobile Network using Dynamic Analysis of Android Application"