Review on Evaluating Efficiency of Android Anti-Malware System

Ms S.M. Shelake[1] Prof. P. B. Dhainje[2]
[1]M.E. Student [2]M.Tech, MBA (IT), PhD
[1,2]Department of Computer Science & Engineering
[1,2]SIETC, Paniv Solapur University, Solapur

Abstract— In this paper we have reviewed and studied various malware detection tools. There are various attacks are applied on anti-malwares. Such attacks can be detected by anti-malware systems but some of them can't detect unknown and all types of attacks. We have reviewed different techniques of malware samples detection, such as resistance to various common obfuscation techniques are analyzed. The possible solution for improving the current state of malware detection on mobile devices can be found using a larger malware samples.

Key words: Malware, Anti-Malware, Obfuscation Techniques, Transformation

I. INTRODUCTION

Day by day the popularity of mobile apps is increasing rapidly. These apps increases ease of access within less time. But with this security must be serious about these devices. Today Android has problems related with security. The Android apps are more vulnerable to malware attackers. There are many anti-malware software's available to protect the android apps against attacks. But with the growth of the android platform, it becomes target for malware authors. Google Play offers different paid and free offerings for anti-malwares. It can be observed that there are many malware threats attempting to break the security chains of android even though there is a huge security provided by different anti-malware softwares. So here, aim is to evaluate the efficiency of anti-malware tools on Android in the face of various techniques. To do so the term 'transformation' which refers to polymorphic changes is done on malware samples. It does not involve code-level changes. These transformed malwares are applied to the anti-malwares to check their efficacy. The technique called Droid Chameleon is used to conduct these common transformations. This technique transforms android applications automatically. That is the reason why we are trying to study the anti-malwares and their capability to defeat the attacks.

Generally, the efficacy of anti-malwares is concerned with following issues:

- Different Malwares and the way they are used for transformation attacks.
- How the anti-malwares are resistant to these transformation attacks.
- The probable ways to improve the efficiency of anti-malwares.

II. LITERATURE SURVEY

MALWARE DETECTION USING VARIOUS TOOLS

A. ADAM

ADAM, an automated and extensible system that can evaluate, the effectiveness of anti-virus using a various malware samples for the Android platform. It automatically transforms an Android malware sample into different variants through various repackaging and obfuscation techniques, while preserving the original malicious behavior Fig.1. shows the design of ADAM.

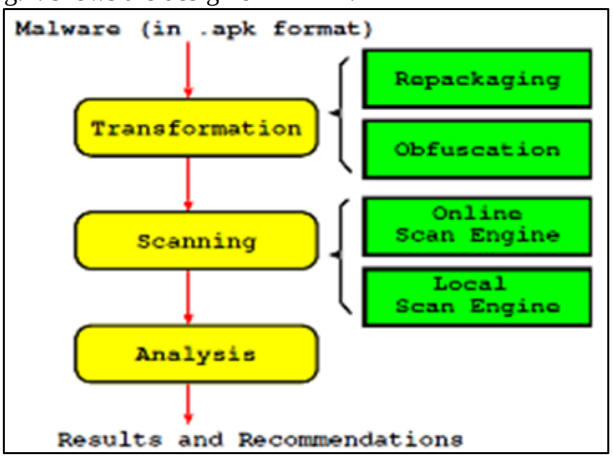


Fig. 1: Design of ADAM

ADAM can automatically transform an original malware sample to different variants via repackaging and obfuscation techniques in order to evaluate the robustness of different anti-virus systems against malware mutation [1]. ADAM is designed by connecting different building blocks. These blocks are used to test different anti-viruses against malware samples.

1) Advantages-

- It can be used for studies of very large-scale malware samples.
- As transformation is done manually there is no need to apply manual modification of malwares. This results less overhead on codes.

2) Limitation-

- ADAM is not always capable to avoid an anti-malware tool. It implements only some of transformations, such as renaming methods, introducing junk methods. It cannot be said that ADAM will always provide the better detection mechanisms.

B. Obfuscation Techniques

Obfuscation techniques have been the focus of much research due to their relevance. This helps to preserve privacy policies between sender and receiver. Fig.2. shows how obfuscation technique provides the protection of messages between Alice and Bob. The source message is compiled and object code is created which is then obfuscated and passed to the server. Server then passes it to the client i.e. Bob. The obfuscated object code is then deobfuscated to object code which is decompiled to original source. Executer does the actual execution.

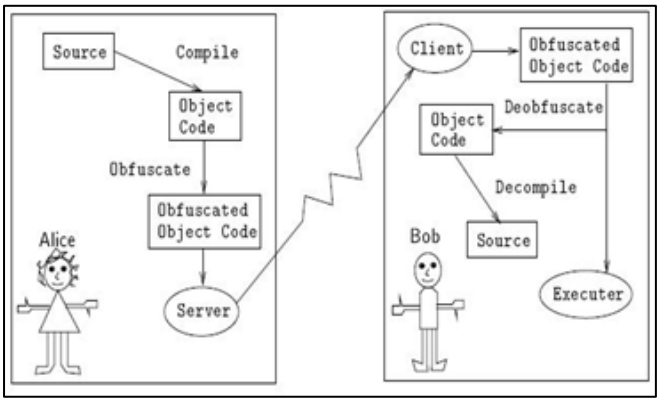


Fig. 2: Protection through obfuscation

1) Advantages-

Obfuscation can be easily used to trace software pirates.

2) Limitations-

The obfuscated software remains secret until the powerful deobfuscator to be built. Therefore, there must be short time period between the releases of obfuscated software versions.

3) Semantics-preserving Malware Detection

M. Christodorescu et al. [3] proposed malware detector that is used to find out the malicious behavior of a program. Most of times hackers use obfuscation to change the malwares. So, here the detectors use pattern-matching technique to search the obfuscations made by hackers.

a) Advantages-

1) It is totally syntax based technique. So, it is easy to be understood by detectors.
2) It has relatively low run time overhead.

b) Limitations-

It is mandatory to save the patterns of malicious instructions into templates which need to use of large databases.

C. Effective and Efficient Malware Detection at the End Host

Clemens Kolbitsch et al. [4] proposed a novel malware detection approach that is both *effective* and *efficient*, and thus, can be used to replace old anti-virus tool at the end host. This technique analyzes a malware to build a model that characterizes its behavior. Such models describe the information flows between the system calls essential to the malware's mission, and therefore, cannot be easily evaded by simple obfuscation or polymorphic techniques. Then, extract the program slices responsible for such information flows. For detection, execute these slices to match with these models against the runtime behavior of an unknown program.

1) Advantages

1) It can effectively detect running malicious code on an end user's host with a small overhead.
2) It generate effective tool that capture detailed information about the behavior of a malwares variation.
3) Scanner that can efficiently match the activity of an unknown program against this system.

2) Disadvantages

1) It cannot generate system call signatures or find a starting point for the slicing process.
2) The new algorithms should be implemented for above limitation.

D. Scalable and Accurate Zero-day Android Malware Detection

Michael Grace et al. [6] proposed proactive scheme to spot zero-day Android malware, It does not depend on malware samples and their signatures. It is an automated system called Risk Ranker to scalable analyze whether a particular app exhibits dangerous behavior (e.g. launching a root exploit or sending background SMS messages).

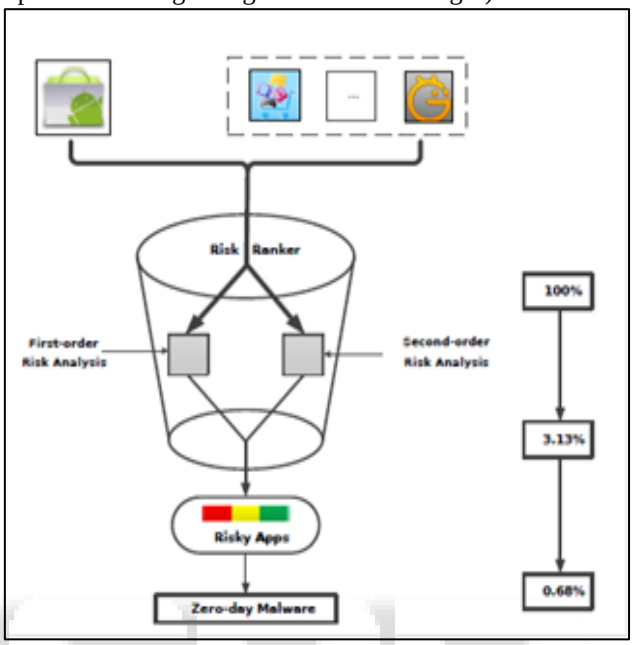


Fig. 3: The Risk Ranker Architecture

Fig. 3. Shows the overall architecture of Risk Ranker. It checks and translates potential security risks into corresponding detection mod-ules of two orders of complexity. The first-order modules handle non-obfuscated apps by evaluating the risks in a straightforward manner; the second-order modules capture certain behaviors (e.g., encryption and dynamic code loading) to detect malware.

E. Automated Remote Repair for Malware

The malicious network traffic increases because of intruders. The problem can be solved by using Airmid, which is an automated system for remote remediation of mobile malware. After the detection of malicious traffic, the cellular network interacts with the source device to identify its originality of that traffic [5]. Fig 3. Shows this approach

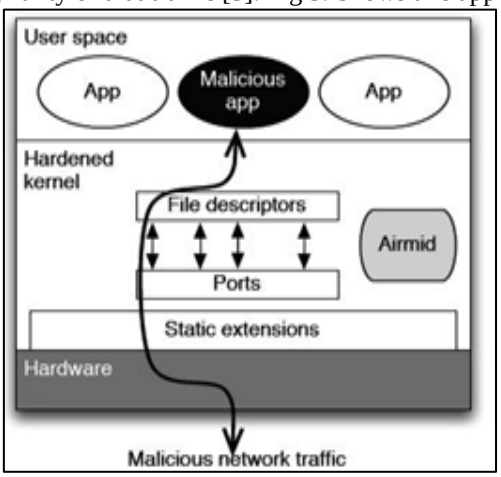


Fig. 4: malicious network traffic

1) Advantage-
- It has very low overhead.

2) Limitations-
- It does not tie to device and its security.
- It is not able to characterize the traffic of large amount of malwares.

F. Automated Security Analysis of Smartphone Applications

To do the automation of security analysis the tool Apps Playground is used. It integrates multiple components comprising different detection and automatic exploration techniques for this purpose [7].The system can be evaluated using multiple large and small scale experiments involving real benign and malicious application.

1) Advantage-

It gives effective analysis even with large number of applications.

2) Limitation-

It is less effective at automatically detecting privacy leaks and malicious functionality in application.

G. Detection of malicious apps in official and alternative Android markets

To find out malicious applications related to android permission based behavioral footprinting scheme is used. It is used for known malwares. Then a heuristics-based filtering scheme is applied to unknown malwares. This total system with known and unknown malicious families is called 'DroidRanger' [8].

1) Advantages-
- It helps to focus on both official and unofficial Android markets for detecting malicious apps.
- By using known and unknown malicious apps the detection proves to be scalable and efficient.

2) Limitation-

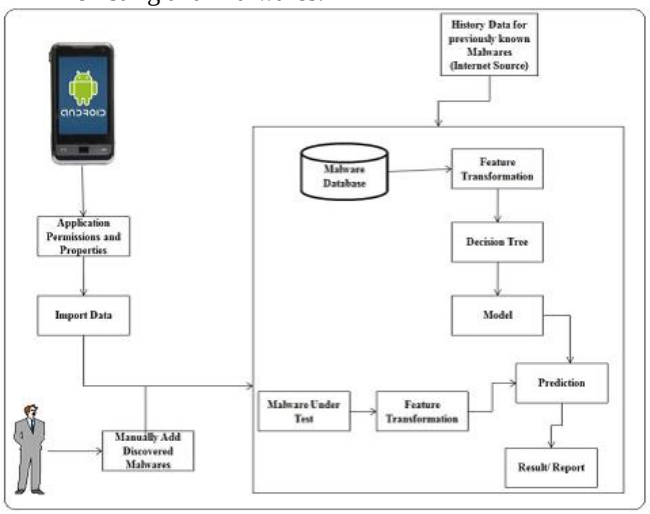
It needs rigorous policing process especially for unofficial marketplaces which is not satisfied by DroidRanger yet.

III. PROPOSED SYSTEM

The system should be able to improve the usage of android apps. The main objective is to explore possible ways to improve current Anti-malware solutions. The system should point out advanced detection techniques because of high-level bytecodes of android. The System should be able to check the whether the given android app is malicious or not.

To evaluate existing anti-malware software, a systematic framework called Droid Chameleon with several common transformation techniques that may be used to transform Android applications automatically. Some of these transformations are highly specific to the Android platform only. Based on the framework, known malware samples can be pass (from different families) through these transformations to generate new variants of malware, which are verified to possess the originals malicious functionality

- The system should be able to evaluate different transformation attacks against anti -malwares of Android.
- The system should be able to check the abilities of anti-malwares to protect software from malware attacks.
- The system should be able to check the vulnerabilities in existing anti-malwares.



IV. CONCLUSION

In this we have analyzed different anti- malwares that can be used for avoidance of different malware attacks. ADAM tool, obfuscation techniques can be used for privacy preserving but with fewer transformations Malware detectors that use obfuscation requires pattern matching techniques. A framework based on DroidChameleon uses even more transformations using which more accurate efficiency of anti-malware tools can be found.

REFERENCES

[1] Zhenlong Yuan, Yongqiang Lu, and Yibo Xue ,"DroidDetector: Android Malware Characterization and Detection Using Deep Learning" IEEE TSINGHUA SCIENCE AND TECHNOLOGY ISSNll1007-0214ll10/10ll pp114-123 Volume 21, Number 1, February 2016.

[2] Vaibhav Rastogi, Yan Chen, and Xuxian Jiang "Catch Me If You Can: Evaluating Android Anti-Malware Against Transformation Attacks" IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY, VOL. 9, NO. 1, JANUARY 2014.

[3] Rastogi, Y. Chen, and X. Jiang, "Droid Chameleon: Evaluating Android antimalware against transformation attacks", in Proc. ACM ASIACCS, May 2013, pp. 329-334.

[4] V. Rastogi, Y. Chen, and W. Enck, "AppsPlayground: Automatic security analysis of Smartphone applications", Proc.ACM Siggraph, in Proc. ACM CODASPY, Feb. 2013, pp. 209-220.

[5] Y. Nadji, J. Giffin, and P. Traynor, "Automated remote repair for mobile malware", in Proc. 27th Annu. Computer. Security Appl. Conf., 2011, pp. 413-422.

[6] M. Frederickson, S. Jha, M. Christodorescu, R. Sailer, and X. Yan,"Synthesizing nearoptimal malware specifications from suspicious behaviors" , in Proc. IEEE Symp. SP, May 2010, pp. 45 60.

[7] M. Christodorescu, S. Jha, S. Seshia, D. Song, and R. Bryant,"Semantics-aware malware detection", in Proc. IEEE Symp. Security Privacy, May 2005, pp. 32-46.

[8] C. Collberg, C. Thomborson, and D. Low, "A taxonomy of obfuscating transformations", Dept. Computer. Sci., Univ. Auckland, Auckland, New Zealand, Tech. Rep. 148, 1997.
[9] ProGuard, [Online]."http://proguard.sourceforge.net/", (2013, Dec. 3).
[10] M. Zheng, P. Lee, and J. Lui, "ADAM: An automatic and extensible platform to stress test Android anti-virus systems", in Proc. DIMVA, Jul. 2012, pp. 1-20.
[11] M. Christodorescu, S. Jha, and C. Kruegel, "Mining specifications of malicious behavior," in Proc. 6th Joint Meeting Eur. Softw. Eng. Conf., ACM SIGSOFT Symp. Found. Softw. Eng., 2007, pp. 5-14.
[12] H. Lockheimer. (2012, Feb.). Android and Security [Online]. Available: http://googlemobile.blogspot.com/2012/02/android-and-security.html .
[13] F-Secure, Helsinki, Finland. (2012). Mobile Threat Report Q3 2012 [Online]. Available: http://www.f-secure.com/static/doc/labs_.

