Madam: Effective and Efficient Behavior-Based Android Malware Detection and Prevention

Asha Patel[1], Almas Safora[2], Amina Javeriya Sultana[3], Basharth Sultana[4]
[1,2,3,4]*Dept. of Computer Science and Engineering, Guru Nanak Dev Engineering College, Karnataka, India*

***Abstract*- Android users are constantly threatened by an increasing number of malicious applications (apps), generically called malware. In this paper we present MADAM, a novel host-based malware detection system for Android devices which simultaneously analyses and correlates features at four levels: kernel, application, user and package, to detect and stop malicious behaviours. MADAM has been designed to take into account those behaviours characteristics of almost every real malware which can be found in the wild. MADAM detects and effectively blocks more than 96% of malicious apps, which come from three large datasets with about 2,800 apps, by exploiting the cooperation of two parallel classifiers and a behavioural signature-based detector.**

1. INTRODUCTION

Smart phones and tablets have become extremely popular in the last years. At the end of 2014, the number of active mobile devices worldwide was almost 7 billion, and in developed nations the ratio between mobile devices and people is estimated as 120.8%. Given their large distribution, and also their capabilities, in the last two years mobile devices have become the main target for attackers. Android, the open source operative system (OS) introduced by Google, has currently the largest market share, which is greater than 80%. Due to the openness and popularity, Android is the main target of attacks against mobile devices 98.5%), with more than 1 million of malicious apps currently available in the wild. Malicious apps (generically called malware) constitute the main vector for security attacks against mobile devices. the official market for Android apps, has hosted apps which have been found to be malicious2. Along with the vast increase of Android malware, several security solutions have been proposed by the research community, spanning from static or dynamic analysis of apps, to applying security policies enforcing data security, to run-time enforcement.

2. LITERATURE SURVEY:

1) "Maximum Damage Malware Attack in Mobile Wireless Networks," Malware attacks constitute a serious security risk that threatens to slow down the large-scale proliferation of wireless applications. As a first step toward thwarting this security threat, we seek to quantify the maximum damage inflicted on the system due to such outbreaks and identify the most vicious attacks. 1) using larger transmission ranges and media scanning rates to accelerate its spread at the cost of exhausting the battery and thereby reducing the overall infection propagation rate in the long run; or 2) killing the node to inflict a large cost on the network.2. "A Social Network Based Patching Scheme for Worm Containment in Cellular Networks," Recently, cellular phone networks have begun allowing third-party applications to run over certain open-API phone operating systems such as Windows Mobile, I phone and Google's Android platform. The performance of these partitioning algorithms is compared against a benchmark random partitioning scheme. Through extensive trace-driven experiments using real IP packet traces from one of the largest cellular networks in the US, we demonstrate the efficacy of our proposed counter-mechanism in containing a mobile worm.

3. PROBLEM IDENTIFICATION

It has been recently reported1 that almost 60% of existing malware send stealthy premium rate SMS messages. Most of these behaviours are exhibited by a category of apps called Trojanzed that can be found

in online marketplaces not controlled by Google. However, also Google Play, the official market for Android apps, has hosted apps which have been found to be malicious2. Along with the vast increase of Android malware, several security solutions have been proposed by the research community, spanning from static or dynamic analysis of apps, to applying security policies enforcing data security, to run-time enforcement. However, these solutions still present significant drawbacks

4. SYSTEM SPECIFICATIONS

HARDWARE REQUIREMENTS:

- System: Intel Core i3 2.8 GHz.
- Hard Disk: 250 GB.
- Monitor: 15" VGA Colour.
- Mouse: Logitech.
- Ram: 1 GB.

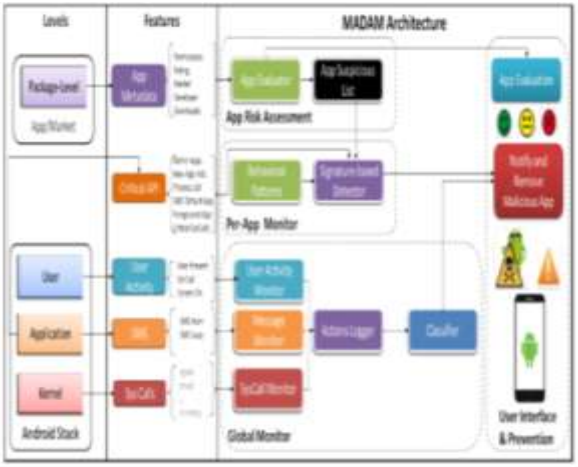
SOFTWARE REQUIREMENTS:

- Operating system: Windows 7.
- Coding Language : Java, Android SDK
- Data Base : SQLite DB

5. METHODOLOGY

The amount of malicious Android apps and malware families is continuously increasing. In fact, according to, more than ten millions of malicious apps for Android were available at the end of 2013. More recently, a report for the first half of 2014 presents an increase consisting of 20 new malware families. Notwithstanding the huge number of malicious apps and threats, Android malware can be grouped into a more limited and manageable number of classes, representative of a specific (malicious) behaviour. For these reasons, we propose the following behaviour-based malware taxonomy (behavioural classes of malware) 3 : 1) Botnet: malware that open a backdoor on the device, waiting for commands which can arrive from an external server or an SMS message. 2) Rootkit: malware that perform buffer overflow to get super user (root) privileges on the device.

6. MADAM ARCHITECTURE

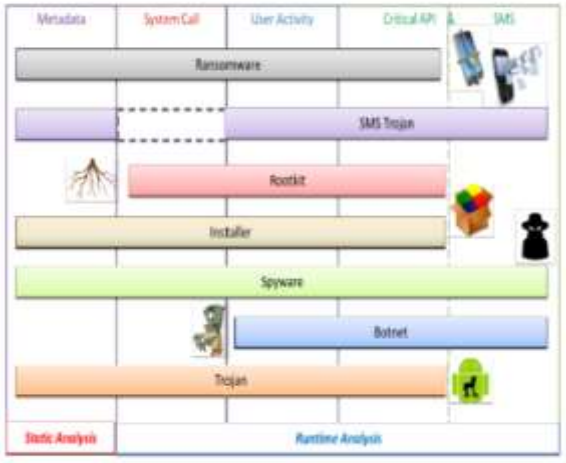


To derive the features at the four system levels, and to detect and prevent a misbehaviour, MADAM can be logically decomposed into four main architectural blocks, which are depicted in Fig. 1 (in particular, see "Madam Architecture"). The first one is the App Risk Assessment, which includes the App Evaluator that implements an analysis of metadata of an app package (apk) (permission and market data), before the app is installed on the device. This evaluation computes the app's risk score, i.e. the likelihood that the app is a malware. Based on this risk evaluation, this component populates a set of suspicious apps (App Suspicious List),which will be monitored at run-time. The second block is the Global Monitor, which monitors the device and OS features at three levels, i.e. kernel (SysCall Monitor), user (User Activity Monitor) and application (Message Monitor).These features are monitored regardless of the specific app or system components generating them, and are used to shape the current behavior of the device itself. Then, these behaviors are classified as genuine (normal) or malicious (anomalous) by the Classifier component. The third block is the Per-App Monitor, which implements a set of known behavioral patterns to monitor the actions performed by the set of suspicious apps (App Suspicious List), generated by the App Risk Assessment, through the Signature-Based Detector.Finally, the User Interface & Prevention component includes the Prevention module, which stops malicious actions and, in case a malware is found, handles the procedure for removing malicious apps using the User Interface (UI). The UI handles notifications to device user, in particular: (i) the evaluation of the risk score of newly-downloaded apps by the App Evaluation, (ii)

the reporting of malicious app (Notify) and (iii) to ask the user whether to remove them

7. IMPLEMENTATION

System Call Monitor The System Call Monitor is used by the Global Monitor to intercept the system calls reported in Table 2 (first eleven columns). These system calls are related to file operations and network access. We have chosen to intercept these calls because we have observed that the greatest amount of operations issued on Android, and consequently by malware, are translated as operations on files at low-level. To intercept system calls on Android, MADAM exploits a kernel module to hook the critical system calls through system call table overriding. The kernel module is loaded through the insmod command and interacts with the rest of the MADAM framework through a shared buffer User Activity and Message Monitor The User Activity and Message Monitor allow MADAM to intercept calls to security relevant API functions, namely related to SMS messages and user activity. As we have previously recalled, these features are critical from a security point of view to detect SMS sent to premium-numbers and/or without the user knowledge. MADAM hijacks security relevant methods, by monitoring their actual parameters and controlling the final outcome of the action. In particular, MADAM hijacks the SendTextMessage() and SendDataMessage() methods to control the events of outgoing SMS messages6



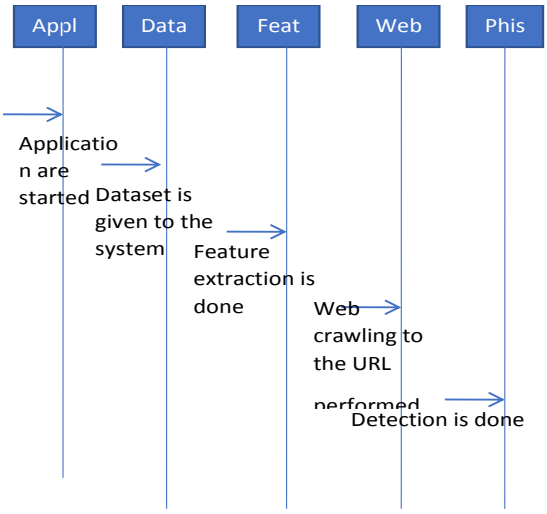
Relevant features tof the Detection of the Sen Maiware Behavioral Classes

8. DESIGN

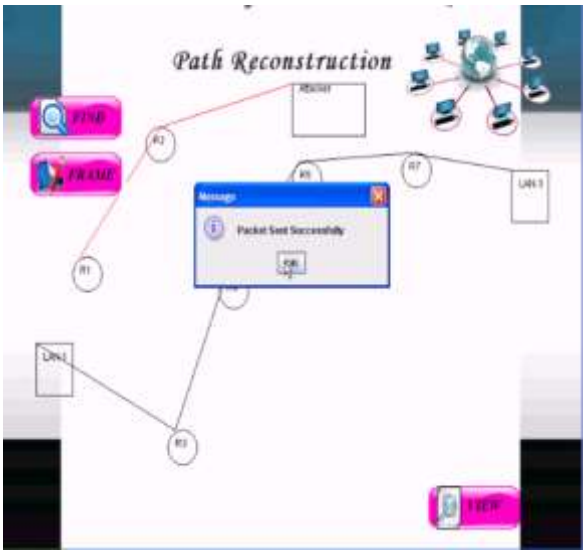
System Design is the next development stage where the overall architecture of the desired system is decided. The system is organized as a set of sub systems interacting with each other. A data-flow diagram (DFD) is a graphical representation of the "flow" of data through an information system. DFDs can also be used for the visualization of data processing (structured design).

Data Flow Diagram:

Sequence Diagram:



9. OUTPUT



10. CONCLUSION & FUTURE SCOPE

Starting from the end of 2011, attackers have increased their efforts toward Android smartphones and tablets, producing and distributing hundreds of thousand of malicious apps. These apps threaten the user data privacy, money and device integrity, and are difficult to detect since they apparently behave as genuine apps bringing no harm. This paper proposes MADAM, a multi-level host-based malware detector for Android devices. By analyzing and correlating several features at four different Android levels, MADAM is able to detect misbehaviors from malware behavioral classes that consider 125 existing malware families, which encompass most of the known malware.

REFERENCES

[1] "Global mobile statistics 2014 part a: Mobile subscribers; handset market share; mobile operators,"http://mobiforge.com/research-analysis/global-mobile-statistics-2014-part-a-mobilesubscribers-handset-market-share-mobile-operators, 2014.

[2] "Sophos mobile security threat reports," 2014, last Accessed: 20 November 2014. [Online]. Available: http://www.sophos.com/en-us/threat-center/mobile-security-threat-report.aspx

[3] M. G. Christian Funk, "Kaspersky security bullettin 2013," December 2013. [Online]. Available: http://media.kaspersky.com/pdf/KSB 2013 EN.pdf

[4] A. Reina, A. Fattori, and L. Cavallaro, "A system call-centric analysis and stimulation technique to automatically reconstruct android malware behaviors," EuroSec, April, 2013.

[5] S. Bugiel, L. Davi, A. Dmitrienko, T. Fischer, A. Sadeghi, and B. Shastry, "Towards taming privilege-escalation attacks on android,"in 19th Annual Network and Distributed System Security Symposium, NDSS 2012, San Diego, California, USA, February 5-8,2012, 2012.

[6] M. Backes, S. Gerling, C. Hammer, M. Maffei, and P. von Styp-Rekowsky, "Appguard fine-grained policy enforcement for untrusted android applications," in Data Privacy Management and Autonomous Spontaneous Security, ser. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2014, pp. 213–231.

[7] Y. Zhou, X. Zhang, X. Jiang, and V. W. Freeh, "Taming information-stealing smartphone applications (on android)," in Proceedings of the 4th International Conference on Trust and Trustworthy Computing, ser. TRUST'11. Berlin, Heidelberg: Springer-Verlag, 2011, pp. 93–107. [Online]. Available: http://dl.acm.org/citation.cfm?id=2022245.2022255

[8] W. Enck, P. Gilbert, B.-G. Chun, L. P. Cox, J. Jung, P. McDaniel, and A. N. Sheth, "Taintdroid: An information-flow tracking system for realtime privacy monitoring on smartphones," in Proceedings of the 9th USENIX Conference on Operating Systems Design and Implementation, ser. OSDI'10. Berkeley, CA, USA: USENIX Association, 2010, pp. 1–6. [Online]. Available: http://dl.acm.org/citation.cfm?id=1924943.1924971

[9] S. Bugiel, L. Davi, A. Dmitrienko, S. Heuser, A.-R. Sadeghi, and B. Shastry, "Practical and lightweight domain isolation on android," in Proceedings of the 1st ACM Workshop on Security and Privacy in Smartphones and Mobile Devices, ser. SPSM '11.

[10] New York, NY, USA: ACM, 2011, pp. 51–62. [Online]. Available: http://doi.acm.org/10.1145/2046614.2046624