TAXONOMY: MOBILE MALWARE THREATS AND DETECTION TECHNIQUES

Lovi Dua and Divya Bansal

Computer Science Department, PEC University of Technology,
Sector 12, Chandigarh 160012, India
dualovi@gmail.com, divya@pec.ac.in

ABSTRACT

Since last-decade, smart-phones have gained widespread usage. Mobile devices store personal details such as contacts and text messages. Due to this extensive growth, smart-phones are attracted towards cyber-criminals. In this research work, we have done a systematic review of the terms related to malware detection algorithms and have also summarized behavioral description of some known mobile malwares in tabular form. After careful solicitation of all the possible methods and algorithms for detection of mobile-based malwares, we give some recommendations for designing future malware detection algorithm by considering computational complexity and detection ration of mobile malwares.

KEYWORDS

Smart-phones, Malware, Attacks, Static analysis, Dynamic analysis

1. INTRODUCTION

Now, there is a thin line difference between Smart-phones, PCs(Personal Computers) and other newly emerged devices like tabs, notebooks and laptops as all are now connected technologies. Due to various services like social networking and gaming provided by smart-phones with the help of applications, these are exposed to gain some confidential information from mobile-devices. Smart-phone OSs includes symbian, android, palmOS and embedded Linux etc. Android is the popular platform for smart-phone based malware authors as any third-party vendor can create applications for android phones and deploy it on android market. Sometimes, even trusted applications are able to leak user's location and phone's identity and share it on server without its consent. Due to this growing skill-set of cyber-criminals who device their algorithms for breaching privacy, embarrassing service-provider and bring inconvenience to the users. So, it requires special care to secure these networked devices from malwares with the help of anti-developed techniques and algorithms for detection. This paper focuses on describing mobile-based threats and its counter detection techniques.

Dhinaharan Nagamalai et al. (Eds) : ACITY, WiMoN, CSIA, AIAA, DPPR, NECO, InWeS - 2014
pp. 213–221, 2014. © CS & IT-CSCP 2014 DOI : 10.5121/csit.2014.4522

1.1 Current State of Study

This section discusses some current malwares reported by security researcher groups. In 2010,different types of mobile malwares are found including DroidDream, Geinimi, GGTracker, Plankton Tonclank and HongTouTou. These malwares are much like original Cabir worm. LookOut security firm reported that over one million of android devices are affected in first half of 2011[21]. In 2012, it is reported by Homeland security department that 79 percent of the mobile threats were targeted to Android operating systems. In January 2012, Symantec identified Trojan horse named AndroidCounterclank for stealing information [3]. Security firm Kaspersky found in 2013 that 98 percent of malware was directed at android platform.

1.2 Organization of paper

In this paper section 2 will discuss mobile device attack vectors and types of detection techniques for mobile malwares. Section3 will discuss detection techniques and algorithms proposed by various researchers and section 4 will give conclusion by analyzing various techniques proposed by different researchers followed by some future recommendations.

2. MOBILE MALWARES

Malware exhibits malicious behavior which targets to mobile phones without user's consent by adding or changing malicious code into software system. Malware is employed intentionally to cause harm to system by gaining confidential information from the device and modifying file contents etc [4]. Malicious executables are further classified into following categories: virus, worm, trojan-horse and botnets. Virus injects malicious code into existing programs to infect them which in turn infects other programs when gets executed. On the other hand, worms spread over the network by exploiting vulnerabilities on the computers connected to network. Trojan appears as benign program but do some malicious activities and botnet gives the attacker ability to remotely control set of user's devices [21].

2.1 Mobile Device Threats

Numerous attack vectors exist which compromises security of mobile devices [5]. Three main categories of attacks could be carried over mobile devices which includes- malware attacks, grayware attacks and spyware attacks described as:-

2.1.1 Malware- These kind of attacks steal personal data from mobile devices and damage devices [22]. With device vulnerabilities and luring user to install additional apps, attacker can gain unauthorized root access to devices. Some of the malware attacks are listed as:-

- Bluetooth attacks: With Bluetooth attacks, attacker could insert contacts or SMS messages, steals victim's data from their devices and can track user's mobile location. Blue-bugging is kind of blue-tooth attack through which attacker could listen conversations by activating software including malicious activities [22].

- SMS attacks: Through SMS attacks, attacker can advertise and spread phishing links. SMS messages can also be used by attackers to exploit vulnerabilities [22].

- GPS/Location attacks: User's current location and movement can be accessed with global positioning system (GPS) hardware and then information can be sold to other companies involved in advertising[22].

- Phone jail-breaking: With jail-breaking, an attacker can remove security implications of operating system like it allows OS to install additional and unsigned applications. Users are attracted to install them as they could get additional functionality [22].

- Premium rate attacks: They posed serious security concerns because premium rate SMS messages could go unnoticed until attacker faces thousands or dollars of bill on his device as they don't need permissions to send SMS on premium rated numbers [22].

2.1.2 Grayware: Grayware include applications which collects the data from mobile devices for marketing purposes. Their intention is make no harm to users but annoy them.

2.1.3 Spyware: Spyware collects personal information from user's phone such as contacts, call history and location. Personal spyware are able to gain physical access of the device by installing software without user's consent. By collecting information about victim's phone, they send it to attacker who installed the app rather than the author of the application.

2.2 Behavioral Classification

Malware may also be classified on the basis of their behavior. Table 1 depicts behavioral classification of some known malwares as shown below:-

Table 1: Malware Behavioral classification

Malwares	Behavior	Description	Operating System
FlexiSPY	Stealing user credentials	Track user information such as emails, photos, browser history and then send it to server.	Symbian, Windows Mobile and BlackBerry.
Fake player	Content delivery manipulation	Runs in background when clicking on media player application. Send SMS Messages to premium rated numbers.	Android OS
Zitmo(Zeus In the Mobile)	Stealing user credentials	Forwards incoming SMS messages from mobile phones to remote server for access of bank accounts.	Android OS
Skuller	Content delivery manipulation	It overwrites system files without user's knowledge as a result smart-phones would stop working and had been switched off.	Symbian OS
Genimi	SMS Spam	It sends multiple spam messages containing phishing links.	Android OS
Hong Tou Tou	Search engine optimization	Improves website ranking in search engines.	Android OS

2.2 Malware detection techniques

Malware can be analyzed with the help of detection techniques. Malware analysis is the process of studying code, behavior and functionality of malware so that severity of attack can be measured. Detection techniques are broadly categorized into three types- static analysis, dynamic analysis and permission-based analysis as shown in Fig 3.1. Figure depicts that static analysis can be done with parameters-static code analysis, taint tracing and control flow dependencies. Dynamic analysis considers parameters including-network traffic, native code and user interaction. Permission-based analysis can be done with the help of permissions specified in manifest file. In literature, various techniques exist for detection of mobile malware.

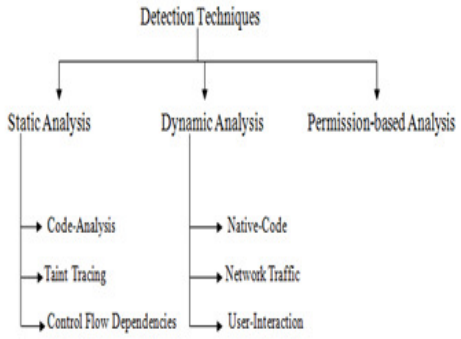

Figure 3.1 Malware detection techniques

2.2.1 Static analysis

Static analysis investigates downloaded app by inspecting its software properties and source code. It is an inexpensive way to find malicious activities in code segments without executing application and notifying its behavior. Many techniques can be used for static analysis: de-compilation, decryption, pattern matching and static system call analysis etc. However, obfuscation and encryption techniques embedded in software makes static analysis difficult. Static analysis is further categorized into two categories- misuse detection and anomaly detection traditionally used by anti-viruses.

2.2.1.1 Misuse detection: Misuse detection uses signature-based approach for detection of malware based on security policies and rule-sets by matching of signatures. In static analysis, data flow dependency and control flow dependencies in source code that would help to understand the behavior of apps.

2.2.1.2 Anomaly detection: Anomaly detection uses machine learning algorithms for learning of known malwares and predicting unknown malware. This approach is suitable for identifying action of malware rather than pattern. Here, methods are used to construct suspicious behavior of applications and then observed signatures are matched against database of normal behavior applications. It is able to distinguish between malicious and normal behavior by training network with classifier such as support vector machine (SVM).

2.2.2 Dynamic analysis

Dynamic analysis involves execution of application in isolated environment to track its execution behavior. In contrast to static analysis, dynamic analysis enables to disclose natural behavior of malware as executed code is analyzed, therefore immune to obfuscation attempts. Various heuristics are considered for monitoring dynamic behavior which includes-monitoring network activity, file changes and system call traces. Android applications can run in an Android SDK, a mobile device emulator running on desktop computer for emulation of software and hardware features except generating phone calls. For testing purposes emulator supports Android Virtual Device(AVD) configurations. When applications start running on the emulator, it can use all services like to invoke other applications, accessing network state, play audio and video, store and retrieve data. Console output is used for debugging, logging of simulated events such as generating phone calls and receiving SMS messages and kernel logs can also be obtained.

2.2.3 Permission-based analysis

Permissions play key role while analyzing android applications .They are listed in Manifest.xml file while each application is installed. Install time permissions limits application behavior with control over privacy and reduces bugs and vulnerabilities [2]. Users have right to allow or deny the installation of applications but he cannot go for the selection of individual permissions. These permissions are required in android applications because the use of resources in android phones is based on these permission set. Some researchers are able to detect malicious behavior of android applications on the basis of permissions specified in Manifest.xml.

3. RELATED WORK

3.1 Static analysis

Kim *et al*. [11] proposed framework for detection and monitoring of energy greedy threats by building power consumption from the collected samples. After generating power signatures, data analyzer compares them with signatures present in a database. Batyuk *et al.*[18] proposed system for static analysis of android applications . First, they provide in-depth static analysis of applications and present readable reports to user for assessment and taking security relevant decisions-to install or not to install an application. Then the method is developed to overcome security threats introduced by the applications by disabling malicious features from them. Ontang *et al.*[19] proposed Secure application Interaction Framework (Saint) by extending android security architecture for protection of interfaces and enhancing interaction policies between calling and callee applications.

Wei *et al.*[15] proposed a static feature-based approach and develop system named Droid Mat able to detect and distinguish android malware . Their mechanism considers the static information including permissions, intents and regarding components to characterize android malware , clustering algorithm is applied to enhance malware modeling capability .K-Nearest Neighbor algorithm classify applications as benign and malicious applications. Finally their results are compared with well known tool Androguard, published in Blackhat 2011 and it is found that DroidMat is efficient as it takes only half time than Androguard to predict 1738 applications.

Bose *et al.* [12] present behavioral detection framework for representation of malware behavior by observing logical ordering of applications actions. Malicious behavior is discriminated from normal behavior by training SVM. System is evaluated for both real-world and simulated mobile malwares with 96% accuracy.

Schmidt *et al.*[10] describes a method for symbianOS malware analysis called centroid based on static function call analysis by extracting features from binaries and clustering is applied for detection of unknown malwares. VirusMeter [9] is proposed to detect anomalous behavior on mobile devices by catching malwares which are consuming abnormal power .Machine learning algorithms helped to improve its detection accuracy. pBMDS [20] an approach through which user-behavior is analyzed by collecting data through logs of key-board operations and LCD displays and then correlated with system calls to detect anomalous activities. Hidden markov model(HMM) is leveraged to learn user-behavior and malware behavior for discrimination of differences between them.

3.2 Dynamic analysis

Batyuk *et al.* [8] proposed an android application sandbox (AA Sandbox) system for analysis of android applications consists of fast static pre-check facility and kernel space sand-box. For suspicious application detection, both static and dynamic analysis is performed on android applications. AASandbox takes APK file and list out following files by decompressing them-Androidmanifest.xml, res/, classes.dex. Manifest file holds security permissions and description of application. Res/ folder defines layout, graphical user interface (GUI) elements and language of application. Classes.dex file contains executable code for execution on dalvik virtual machine which is then de-compiled to java files with baksmali and then code is searched for suspicious patterns. Monkey program designed for stress testing of applications generates pseudo random sequences of user-events such as touches and mouse-clicks. It is used to hijack system calls for logging operation and helpful to get the logging behavior of application at system level. Around 150 applications are collected for testing and evaluation.

Min *et al.* [13] proposed run-time based behavior dynamic analysis system for android applications. Proposed system consists of event detector, log monitor and parser. Event trigger is able to simulate the user's action with static analysis. Static analyzer generates manifest.xml and java code with the help of application .apk file. Semantic analysis find list of risk based permissions, activities and services including other information such as hash code and package name. Data flow analysis creates control flow graph (CFG) of the application by mapping of user-defined methods and API calling. By running application in a customized emulator with loadable LKM, sensitive information about application can be captured such as sent SMS , call log and network data for entry address of system calls. Logs recorded with debugging tool logcat for sensitive behavior sent to Log parser. Log monitor gathers log data as the application runs and parser analyzes log data by picking sensitive information and filtering out unnecessary information. By collecting 350 apps from the Amazon Android Market, results found that about 82 applications leak private data.

Enack *et al.* [14] proposed Apps-playground framework for automatic dynamic analysis of android applications. Designed approach is able to analyze malicious applications in addition to applications leaking private data from smart-phones without the user's consent. Dynamic analysis should possess detection techniques including ability to explore application code as much as

possible and the environment should be as much real that malicious application could not obfuscate. Automatic analysis code integrates the detection, exploration and disguise techniques to explore android applications effectively. Detection techniques detect the malicious functionality while app is being executed .It includes taint tracing which monitor sensitive APIs with TaintDroid such as SMS APIs and kernel level monitoring for tracing of root exploits.Automatic exploration techniques are helpful for code coverage of applications by simulating events such as location changes and received SMS so that all application code is covered. Fuzzy testing and intelligent black box execution testing is used for automatic exploration of android applications. Disguise techniques create realistic environment by providing data such as International mobile equipment identity(IMEI), contacts, SMS, GPS coordinates etc.

Enck *et al.* [7] proposed TaintDroid for dynamic analysis. First dynamic analysis tool used for system wide analysis of android applications by tracking flow of sensitive information through third-party applications. TaintDroid integrates multiple granularities at object level i.e, variable, method, message and file level. It is able to monitor how the sensitive data are used by applications and then taints are labeled. TaintDroid is tested on around 30 applications and it is found that 15 of them uses personal information.

3.3 Permission-based analysis

Johnson *et. al.* [16] proposed architecture for automatic downloading of android applications from the android market. Different algorithms employed for searching of applications such as downloading applications by application category. With static analysis, required permissions can be obtained based on its functionality. Permission names are searched in android source code and then mapped with API calls to know that whether requested permissions are correct or not. Program examines all smali files of application to obtain list of method calls used in an application. Each method call is then compared with method call listed in permission protected android API calls to know exact permissions. Restricted permission set is compared with all the permissions specified in AndroidManifest.xml file to find out extra permissions, lacking of permissions and exact permission set required for its functionality.

Zhou *et al.* [17] proposed DroidRanger for systematic study on overall health of both official and unofficial Android Markets with the focus on the detection of malicious apps. DroidRanger leverages a crawler for collection of apps from the Android Market and saved into local repository. Features extracted from collected apps include requested permissions and author information. Two different detection engines are used for detection of known and unknown malwares. First detection engine is permission-based behavioral foot-printing scheme able to distil apps requiring dangerous permissions such as SEND_SMS and RECEIVE_SMS permissions. Therefore, number of apps to be processed for second detection engine is reduced. In second step, multiple dimensions for behavioral foot-printing scheme chosen for listening of all system-wide broadcast messages if they contains receiver named android.provider.Telephony.SMS_RECEIVED. Obtained callgraph associates API calls to specific components specified in a rule. For example- by calling abortBroadCast function with specific rule, a method is obtained to detect apps monitoring incoming SMS messages. Second detection engine includes some heuristics to detect suspicious apps and zero-day malwares. Heuristics attempts to dynamically fetch and run code from untrusted websites which is further monitored during run-time execution to confirm whether it is truly malicious or not.

4. CONCLUSION

Smart-phones are becoming popular in terms of power, sensor and communication. Modern, smart-phones provide lots of services such as messaging, browsing internet, emailing, playing games in addition to traditional voice services. Due to its multi-functionality, new security threats are emerged for mobile devices. In this paper, we presented survey on various techniques for detection of mobile malware. We have categorized various mobile malware detection techniques based on features extracted from them and monitoring system calls as they provide us low level information. We have analyzed that information-flow tracking, API call monitoring and network analysis provide more deeper analysis and useful information for detection of mobile malware.

5. RECOMMENDATIONS FOR FUTURE

Following are some recommendations for designing algorithm to detect mobile-based applications containing malwares.

1. Multiple sources for feature extraction should be used for building feature-set to detect mobile malwares.
2. There should be national or international database for reporting malware incidents so that developers are aware of distinct vulnerabilities related to mobile malwares.
3. Artificial intelligence algorithms(neural network-based) to improve detection ratio.
4. Machine to machine communication and authentications tools must be used in between multiple device platforms

REFERENCES

[1] F-Secure. Trojan:symbos/yxe, http://www.virus.fi/v-descs/trojan_symbos_yxe.shtml.
[2] Manifest.permission,Androiddeveloper, http://developer.android.com/reference/android/Manifest.permission.html
[3] Android.Counterclank Found in Official Android Market, http://www.symantec.com/connect/fr/blogs/androidcounter
[4] M.L.Polla ,F. Martinelli, D.Sgandurra: A Survey on Security for Mobile Devices: Communications Surveys and Tutorials, pp.446-471.IEEE(2013)
[5] McAfee Labs Q3 2011 Threats Report Press Release, 2011, http://www.mcafee. com/us/about/news/2011/q4/20111121-01.aspx
[6] M. Chandramoha, H.Tan: Detection of Mobile Malware in the Wild.:Computer (Volume:45 , Issue: 9) ,pp.65-71(2012)
[7] W.Enck, P. Gilbert, B.G. Chun, L.P.Cox, J.Jung, P.McDaniel, A.P.Sheth: TaintDroid: an information-ow tracking systemfor realtime privacy monitoring on smart-phones.:In OSDI'10 Proceedings of the 9th USENIX conference on Operating systems design and implementation,pp.1-6 ,USENIX Association Berkeley, CA,USA (2010)
[8] T.Blasing, L.Batyuk, A.D.Schimdt, S.H.Camtepe, S.Albayrak,:An Android Application Sandbox System for Suspicious Software Detection.
[9] L.Liu,G.Yan, X.Zhang, S.Chen,: VirusMeter: Preventing Your Cell phone from Spies.: Proceedings of the 12th International Symposium on Recent Advances in Intrusion Detection.,pp.244-264, Springer-Verlag,Berlin, Heidelberg(2009).
[10] A.D.Schmidt, J.H.Clausen,S.H.Camtepe, S.Albayrak: Detecting Symbian OS Malware through Static Function Call Analysis: In Proceedings of the 4th IEEE International Conference on Malicious and Unwanted Software,pp.15-22.IEEE(2009).

[11] H.Kim, J.Smith, K.G.Shin,:Detecting energy-greedy anomalies and mobile malware variants: In MobiSys 08: Proceeding of the 6th international conference on Mobile systems, applications, and services,pp.239-252.ACM,NewYork(2008).

[12] A. Bose,X.Hu, K.G.Shin, T.Park: Behavioral detection of malware on mobile handsets:In MobiSys 08: Proceeding of the 6th international conference on Mobile systems, applications, and services,pp.225-238.,ACM,NewYork(2008).

[13] L.Min,Q.Cao: Runtime-based Behavior Dynamic Analysis System for Android Malware Detection: Advanced Materials Research,pp.2220-2225.

[14] V.Rastogi, Y.Chen, W.Enck: AppsPlayground: Automatic Security Analysis of Smartphone Applications:In CODASPY'13 Proceedings of the third ACM conference on Data and application security and privacy,pp.209-220.ACM,NewYork(2013)

[15] D.J.Wu,C.H.Mao,T.E.Wei,H.M.Lee,K.P.Wu: DroidMat: Android Malware Detection through Manifest and API Calls Tracing.: In Information Security (AsiaJCIS), 2012 Seventh Asia Joint Conference ,pp.62-69.IEEE,Tokyo(2012)

[16] R.Jhonson, Z.Wang, C.Gagnon, A.Stavrou,: Analysis of android applications' permissions.:In Software Security and Reliability Companion (SERE-C) Sixth Inter-national Conference,pp.45-46.IEEE(2012)

[17] Y.Zhou,, Z.Wang, W.Zhou,X.Jiang: Hey, You, Get o_ of My Market: Detecting Malicious Apps in O_cial and Alternative Android Markets: In Proceedings of the 19th Network and Distributed System Security Symposium,San Diego,CA(2012).

[18] L.Batyuk,M.Herpich,S.A.Camtepe,K.Raddatz,A.D.Schmidt,S.Albayrak:Using static analysis for automatic assessment and mitigation of unwanted and malicious activities within Android applications.: In 6th International Conference on Malicious and Unwanted Software,pp.66-72.IEEE Computer Society(2011)

[19] M.Ongtang,S.E.McLaughlin,W.Enck,P.D.McDaniel,:Semantically rich application-centric security in android:In Proceedings of the 25th Annual Computer Security Application Conference (ACSAC),pp.340-349(2009)

[20] L.Xie, X.Zhang, J.P.Siefert, S.Zhu: pBMDS: a behavior-based malware detection system for cellphone devices.:In Wisec'10 Proceedings of the third ACM conference on Wireless network security,Hoboken,pp.37-48.ACM,USA(2010)

[21] A.P.Felt ,M.Finifter,E.Chin,S.Hanna,D.Wagner:A survey of mobile malware in the wild.:In Proceedings of the 1st ACM workshop on Security and privacy in smart phones and mobile devices,pp.3-14.ACM,NewYork(2011)

[22] D.Stites, A.Tadimla :A Survey Of Mobile Device Security: Threats, Vulnerabilities and Defenses./urlhttp://afewguyscoding.com/2011/12/survey-mobile-devicesecurity-threats-vulnerabilities-defenses