

A Review on Camera Based Attacks on Android Smart Phones

¹Anushree Pore, ²Mahip Bartere

¹M.E. Scholar, GHRCEM, Amravati, Maharashtra, India

²Assistant Professor, GHRCEM, Amravati, Maharashtra, India

Abstract

Nowadays, almost all the smart phones have features like camera and touch screen. These features may lead attacks on our smart phones. Modern smart phone platforms let users customize their device via third-party applications found on “app stores” or traditional websites. Application provenance is a problem so users are constantly at risk of installing malicious apps that steal personal data or gain root access to their device. For example, while using such malicious application, the response from application provider may contain the hidden request to have control on different devices connected to our mobile such as camera, front or main no issues phone is been attacked, recognizing our current location through main camera as it will show our surroundings and trying to recognize PIN's through front camera. This paper reviews new security threats are emerged for mobile devices and survey on various techniques for detection of mobile malware.

Keywords

Camera Based Attacks, WatchDog, Anti-Thief

I. Introduction

Mobile phones are becoming important part of our day to day life specially the smart phones, since they are involved in keeping in touch with friends and family, doing business, accessing the internet and other activities. Andy Rubin, Google's director of mobile platforms, has commented: “There should be nothing that users can access on their desktop that they can't access on their cell phone” [1]. Growth in smart phone sales is depicted in the figure below.

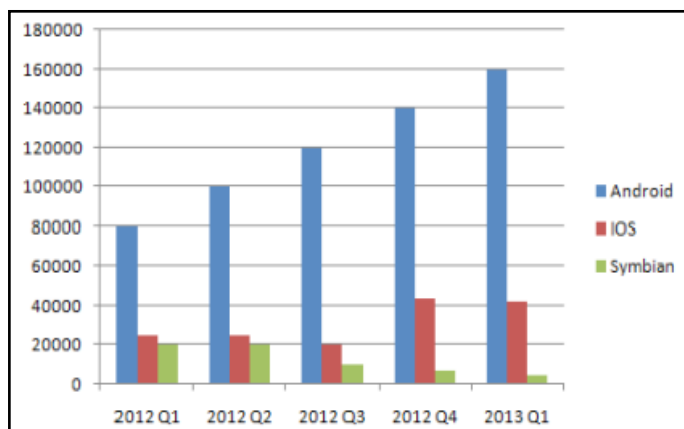


Fig. 1: Smartphone Sales Worldwide

It indicates that smart phone sales are continuously on rise and more and more people are becoming dependent on these devices. As these smart phones are going to outnumber the world's total population in 2014, securing these devices has assumed paramount importance. Owners use their smart phones to perform tasks ranging from everyday communication with friends and family to the management of banking accounts and accessing sensitive Work related data. These factors, combined with limitations in administrative device control through owners and security critical

applications like the banking transactions, make Android-based Smart phones a very attractive target for hackers, attackers and malware authors with almost any kind of motivation.

Smart phones retrieve apps from application markets and run them within a middleware environment. Existing smart phone platforms rely on application markets and platform protection mechanisms for security. The fig. 2 shows the general architecture of smart phones.

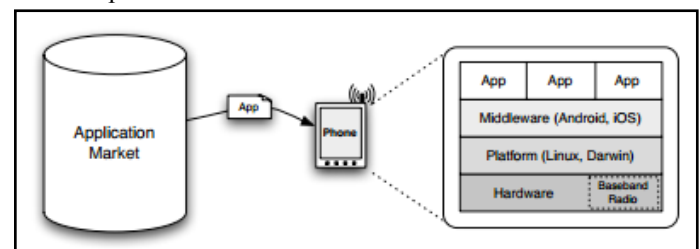


Fig. 2: General Smart Phone Architecture

II. Literature Review

As, all the smart phone uses the application from the market, smart phones are possible to get attacked through such malicious application. Next section gives the detail about such security threats.

A. Mobile Device Threats

Numerous attack 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:-

1. Malware

These kinds of attacks steal personal data from mobile devices and damage devices [2]. 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:-

(i). 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 [2].

(ii). SMS Attacks

Through SMS attacks; attacker can advertise and spread phishing links. SMS messages can also be used by attackers to exploit vulnerabilities [2].

(iii). 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 [2].

(iv). 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 [2].

(v). Premium Rate Attacks

They posed serious security concerns because premium rate SMS messages could go unnoticed until attacker faces thousands of dollars of bill on his device as they don't need permissions to send SMS on premium rated numbers [2].

(vi). Grayware

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

(vii). 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.

B. Static Analysis

Static analysis investigates downloaded app by inspecting its software properties and source code. However, obfuscation and encryption techniques embedded in software makes static analysis difficult. Static analysis is further categorized into two categories- signature-based detection and behavior-based detection traditionally used by anti-viruses.

Kim et al. [7] 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. [14] 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. [15] 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. [11] 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. [8] 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. [6] 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.

L Xie et. al. [16], pBMDS 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.

C. 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.

Batyuk et al. [4] 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 define 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. [9] 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 userdefined 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. [10] 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. [3] proposed TaintDroid for dynamic analysis. First dynamic analysis tool used for system wide analysis of android applications by tracking flow of sensitive information through thirdparty 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.

D. Permission-Based Analysis

With the help of listed permissions in manifest.xml, various researchers are able to detect applications malicious behavior. [2] These permissions have the ability to limit application behaviour by controlling over privacy and reducing bugs and vulnerabilities. Johnson et. al. [12] 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. [13] 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.

E. Related Work

Soundcomber [17] is a stealthy Trojan that can sense the context of its audible surroundings to target and extract highvalue data such as credit card and PIN numbers. Stealthy audio recording is easier to realize since it does not need to hide the camera preview.

Xu et al. [18] present a data collection technique using a video camera embedded in Windows phones. Their malware (installed as a Trojan) secretly records video and transmits data using either email or MMS. Windows phones offer a function, ShowWindow(hWnd, SW_HIDE), which can hide an app window on the phone screen. However, it is much more complicated (no off-the-shelf function) to hide a camera preview window in an Android system. In this work, we are able to hide the whole camera app in Android. Moreover, we implement advanced forms of attacks such as remote-controlled and real-time monitoring attacks. We also utilize computer vision techniques to analyze recorded videos and infer passcodes from users' eye movements.

Several video-based attacks targeted at keystrokes have been proposed. The attacks can obtain user input on touch screen smartphones.

Maggi et al. [19] implement an automatic shoulder surfing attack against modern touch-enabled smartphones. The attacker deploys a video camera that can record the target screen while the victim is entering text. Then user input can be reconstructed solely based on the keystroke feedback displayed on the screen. However, this attack requires an additional camera device, and issues like how to place the camera near the victim without catching an alert must be considered carefully. Moreover, it works only when visual feedbacks such as magnified keys are available.

iSpy [20], proposed by Raguram, shows how screen reflections may be used for reconstruction of text typed on a smartphone's virtual keyboard. Similarly, this attack also needs an extra device to capture the reflections, and the visual key press confirmation mechanism must be enabled on the target phone. In contrast, our camera-based attacks work without any support from other devices.

Longfei Wu [21] implemented the attacks on real phones, and demonstrate the feasibility and effectiveness of the attacks. Furthermore, they propose a lightweight defense scheme that can effectively detect these attacks.

III. Disadvantages

As mentioned above, the role a spy camera plays depends on the way it is used and who is in control of it. In the following, we discuss some threats and benefits of using a spy camera.

A. Leaking Private Information

A spy camera works as a thief if it steals private information from the phone. First, the malware finds a way to infect the victim's smartphone. For example, it appears to be a normal app with legitimate use of a camera and the Internet. On one hand, it performs the function it claims. On the other hand, it runs a background service to secretly take pictures or record videos, and store the data with obscure names in a directory that is seldom visited. Then these data are sent out to the attacker when WiFi (fast

and usually unlimited) access or other connection is available.

B. Watchdog

Watchdog is another thing a spy camera can do. Nobody wants other people to use or check his/her phone without permission. A spy camera can stealthily take pictures of the phone user and deter those who use or check other people's phones.

C. Anti-Thief

On the other hand, a spy camera could play a completely different role if it is used properly. When a user loses his/her phone, the spy camera could be launched via remote control and capture what the thief looks like as well as the surrounding environment. Then the pictures or videos along with location information (GPS coordinates) can be sent back to the device owner so that the owner can pinpoint the thief and get the phone back.

IV. Counter Measure

In this section, we discuss possible countermeasures that can protect Android phones against these spy camera attacks. In an Android system, no application programming interface (API) or log file is available for a user to check the usage of a camera device. Hence, detection of camera-based attacks requires modification to the system. So, the application can be developed which detects the hidden request in the response from the application provider. Such app will check the hidden request and presents an alert dialog including the name of the suspicious app is displayed, and what kind of hidden request is for will be displayed, for e.g. app wants to use camera, this is the hidden request called spy camera attack. Besides, the detailed activity patterns of suspected apps are logged so that the user can check later.

V. Conclusion

Now days more than 1 million Android device activated Android has very few restrictions for developer, increases the security risk for end users. In this paper we have reviewed security issues in the Android based Smartphone. The integration of technologies into an application certification process requires overcoming logistical and technical challenges. Android provides more security than other mobile phone platforms. Moreover, in this paper, we study camera-related vulnerabilities in Android phones for mobile multimedia applications. We discuss the roles a spy camera can play to attack or benefit phone users.

References

- [1] Google bets on Android future. [Online] Available: <http://news.bbc.co.uk/2/hi/technology/7266201.stm>
- [2] D.Stites, A.Tadimla: A Survey of Mobile Device Security: Threats, Vulnerabilities and Defenses. [Online] Available: <http://afewguyscoding.com/2011/12/survey-mobile-device-security-threats-vulnerabilities-defenses>.
- [3] W.Enck, P. Gilbert, B.G. Chun, L.P.Cox, J.Jung, P.McDaniel, A.P.Sheth: TaintDroid: An information on tracking system for 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)
- [4] T.Biasing, L.Batyuk, A.D.Schmidt, S.H.Camtepe, S.Albayrak: An Android Application Sandbox System for Suspicious Software Detection.
- [5] McAfee Labs Q3 2011 Threats Report Press Release, 2011, [Online] Available: <http://www.mcafee.com/us/about/news/2011/q4/20111121-01.aspx>
- [6] 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.
- [7] 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, New York, 2008.
- [8] A. Bose, X.Hu, K.G.Shin, T.Park, "Behavioral detection of malware on mobile handsets", In MobiSys08: Proceeding of the 6th international conference on Mobile systems, applications, and services, pp. 225-238, ACM, New York, 2008.
- [9] L.Min, Q.Cao, "Runtime-based Behavior Dynamic Analysis System for Android Malware Detection: Advanced Materials Research, pp. 2220-2225.
- [10] 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, New York, 2013.
- [11] 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.
- [12] R.Jhonson, Z.Wang, C.Gagnon, A.Stavrou.: Analysis of android applications' permissions.: In Software Security and Reliability Companion (SERE-C) Sixth International Conference, pp.45- 46. IEEE(2012)
- [13] 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). International Journal of Distributed and Parallel Systems (IJDPs) Vol.5, No.4, July 2014.
- [14] 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)
- [15] 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)
- [16] 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.
- [17] R. Schlegel et al., "Soundcomber: A Stealthy and Context-Aware Sound Trojan for Smartphones", NDSS, 2011, pp. 17-33.
- [18] N. Xu et al., "Stealthy Video Capturer: A New Video Based Spyware in 3g Smartphones", Proc. 2nd ACM Conf. Wireless Network Security, 2009, pp. 69-78.
- [19] F. Maggi, et al., "A Fast Eavesdropping Attack against Touchscreens", 7th Int'l. Conf. Info. Assurance and Security, 2011, pp. 320-25.

- [20] R. Raguram et al., “ispy: Automatic Reconstruction of Typed Input from Compromising Reflections”, Proc. 18th ACM Conf. Computer and Commun. Security, 2011, pp. 527–36.
- [21] Longfei Wu et. al., “Security Threats to Mobile Multimedia Applications: Camera-Based Attacks on Mobile Phones”, Security in Wireless Multimedia Communications, IEEE Communications Magazine, March 2014, pp. 80-87.