Real-Time Monitoring of Privacy Abuses and Intrusion Detection in Android System

Shancang Li[1(✉)], Junhua Chen[2], Theodoros Spyridopoulos[2], Panagiotis Andriotis[2], Robert Ludwiniak[1], and Gordon Russell[1]

[1] School of Computing, Edinburgh Napier University, Edinburgh, UK
{s.li,r.ludwiniak,g.russell}@napier.ac.uk
[2] Cryptography Group, University of Bristol, Bristol, UK
{th.spyridopoulos,p.andriotis}@bristol.ac.uk

Abstract. In this paper, we investigated the definition of privacy, privacy abuse behaviours, and the privacy abuse in Android systems, which may be very useful for identifying the malicious apps from 'normal' apps. We also investigated the injection technology, service binding, and service proxy in Android system, which are widely used by normal apps to steal privacy information. A real-time monitoring system (app) is developed on Android system to monitor potential privacy data abuse. The app is able to monitor permission requests for all installed apps as well as analyse the potential privacy abuse behaviors.

1 Introduction

Mobile device is becoming a primary platform equipped with powerful sensing, computing, and networking capabilities [1]. The convenience to users and developers has made mobile device a fun and intelligent information-processing terminal. The popularity and advanced functionality of mobile devices eared the attention of hackers and cybercriminals. Android based devices are so widely used that millions applications (apps) have been developed related to daily life. It is reported that the Android is at the highest as greater than 80 % share of the global market in 2014 [2–4]. Meanwhile, the 99 % of mew mobile malware is designed to target Android [5]. As a result, it is very urgent to find a method to evaluate, monitor, and solve the security issues in Android. In Android, most apps are running in a "sandbox" and cannot affect other apps [6]. Therefore, most malicious apps are unable to break the system but they can steal the personal informations of Android device users, which should be treated as private data abuses [7–9]. To identify and manage user data and information is a very crucial and sensitive topic in Android system. Most of private data abuse behaviors could be detected when apps are submitted to official apps market, such as `Play Store`, etc."Permission system" is the most important part of security system in Android [6]. However, most apps ask for many more permissions than they require. It is very difficult for Android users to manage and understand the app permissions requests. This may causes the personal data is being abused against the users' wishes. Android users are unable to know what an app would

T. Tryfonas and I. Askoxylakis (Eds.): HAS 2015, LNCS 9190, pp. 379–390, 2015.
DOI: 10.1007/978-3-319-20376-8_34

really do. Installing an app takes just a couple clicks to choose whether to allow the required permissions. However in using an app, it is very difficult for users to change the granted permissions [9].

In Android systems, many apps are able to access to sensitive information on the mobile device. Misusing permissions to access these information is a major cause of privacy breaches and data leakage. The Android users are not able to control over the capabilities of apps once the apps have been granted the requested permissions upon installation. This makes it possible for malicious and intrusive apps to abuse the granted permission, data, and resources. This may expose sensitive information to unauthorized apps or code. In recent, a number of serious vulnerabilities have been reported that some malware and adware apps can authorize themselves properly to access the contacts, sms, email, and other sensitive information. Numerous security applications have been developed for most bands of mobile devices; however most of them are not currently targeting actual data, mobile malware attacks. The Android systems have to face the new challenge on user privacy preservation and security protection. In response to the growing threat, mobile device are developing built-in security features. In the meantime, users can protect their data by carefully cutinizing third-part applications, and avoiding suspicious information fishing.

Therefore, a real-time monitoring system is needed to protect users from privacy data abuse. It can help the users to make real decision by real time monitoring permissions according to the behaviors of an app. The system can protect users from privacy abuse before it happens. Actually, a number of security apps can be found in `Play Store`, such as `LBE Privacy Guard, Clueful Privacy Advisor, PrivacyScanner, 360 Mobile Safe etc.` [5–11]. These apps are developed to help users know the details of apps when they are running in Android by analysing their permissions, network flows, access, etc. However, it is still difficult to tell user what apps really do. The goal of this paper is to analyse the behaviors and intent of recent types of privacy-invasive.

2 Privacy Abuse in Android System

A basic Android system includes four layers: *application layer, application framework, library* & *Android runtime*, and *linux kernel* [12,13]. Each app in Android runs in Dalvik virtual machine (DVM) and it is unable to affect other apps because each application runs in their own sandbox. The `AndroidManifext.xml` defines all permissions of an app [1,14].

2.1 Potential Vulnerabilities in Android System

Although the permission system in Android is designed to protect the users information, it is widely reported that many apps are able to steal user data [9]. It isn't strong enough to protect users' privacy and prevent abuse [13], which can tell users the accessibility to privacy for each app, but unable to identify malicious apps from normal apps.

Table 1. List of privacy in android systems

Privacy	Related to
Call history	Privacy
Contact list	Privacy
Sms	Privacy
Sending short message	Identity
Phone number (IMSI)	Identity
IMEI	Identity
GPS location	Privacy & User's physical freedom
Camera	Privacy & Users' life embarrassing fact
WiFi	Network
Bluetooth	Network

2.2 Privacy in Android

The definition of privacy should fit the environment. Privacy data and information are issues for users of all types of electronic device [10]. A running app can access the 'call history', 'contact list', 'sms', 'location', *etc.*, some of these information are sent to developers or specific sites without users knowing about or being able to opt out of the practice. Mobile Android users are more concerned with privacy on their devices and they especially worry about the threat of malicious apps [10], such as `call logs, contacts, sms, phone numbers, locations,` etc. Table 1 lists the commonly used privacy information.

2.3 Privacy Abuse Behaviors in Android

It is difficult to define the *privacy abuse behaviour*. In [15] and [16], behaviors such as *"identifier disclosure", "short message service misuse", "spy camera"*, and *"location leakage"* are defined as privacy abuse behaviors. If the identifiers are leaked, the mobile devices could be tracked or sensitive information could be abused. The misusing of `sms` may lead serious problem in some checking system. Similarly, the location leakage can cause tracking or more serious things. The privacy abuse is seriously on Android system. Although many vulnerabilities could be fixed by a system patch but it is not an ultimate solution since the patch may be unable to fix all problems [17]. In this work, we summarized the potential privacy abuse behaviors in Android systems as Table 2.

2.4 Privacy Abuse Detecting

There are three commonly used methods to detect malware apps: *static, dynamic, extra information*, as shown in Table 3. The *static* method statically check the permissions, imported package, instruction (opcode), et al. of apps but

Table 2. List of privacy in android systems

Privacy	Potential privacy abuse behavior
Reading call history	The call history contains information about users' life, work, and network, which is a potential privacy abuse behavior
Reading contact List	Contact list leakage can leave owner of device and it's contact member in a risk state
Reading sms	Sms contains very private information of users. Furthermore, some payment systems are use sms as a second authentication method
Sending short message	Attempting to send a sms by malware app can cause serious results
Reading phone number	The phone number is an identity of users
Reading IMSI number	The international mobile subscriber identity (IMSI) is also an identity of SIM card of users
Reading IMEI number	The international mobile equipment identity (IMEI) is the identity of device
Reading GPS location	The location of user or mobile device is a kind of sensitive information
Using camera	Improper use of camera or photos in mobile device can cause privacy leakage
Using WiFi	Untrust WiFi can cause information leakage
Using bluetooth	Similar to WiFi

cannot analyse apps at runtime. *Dynamic* methods check the runtime behaviors of apps, such as *system calls, etc.* The *extra information* methods use extra information such as *the author of apps, description of apps* to analyse apps for judging whether apps are malware. On the other hand, the `service manager` can be used to detect the privacy abuse behavior, with which the attacker can use the `ptrace` function to inject the specific code to steal privacy information.

2.5 Resist Privacy Abuse

The permissions list and comments can help users make decision as to whether an app might need the permissions. Furthermore, the potential behavior of privacy abuse can be real-timely detected by hooking the `service manager`, which are logged for further analysis to help user to decide to uninstall apps or not. These methods are used to help users to resist the privacy abuse before it happens.

3 Real-Time Monitoring of Privacy Data Abuse System

This section will describe the detailed design of the system, which aims at real-time monitoring the permissions and behaviors of all installed apps to protect

Table 3. List of privacy in android systems

Methods	Advantages	Disadvantages
Static analysis	Stable, don't affect running apps	Cannot real-timely detect potential privacy abuse behaviors
Dynamic analysis	Accurate, can real-timely detect behaviors	May affect running apps
Extra information	Stable, Don't affect running apps	(1) Cannot real-timely detect behaviors of apps; (2) Inaccurate; (3) The extrainformation may be faked

users from privacy abuse. When finding some suspect apps, the system will blocked these apps and send notification to user. All suspect behaviors will be logged for further analysis. With this system, users are able to decide to block a malware behavior. The system contains following four basic tasks: (1) It can list permissions of all installed apps in an Android device; (2) It is able to monitor all predefined suspect behaviors; (3) The system can read/write comments for each apps; (4) The injection module can replace a function in a process of Android, which demonstrates how a fake `ioctl` function replace the original function; (5) A proxy services is implemented that can work as a proxy between applications and original services, which is able to detect potential privacy abuse behaviors.

3.1 Architecture Design

The system contains multiple four modules, including `activity module, inject process module, online comment service module` and `background service module`. The `background service` contains tow submodules: `service proxy and log record`. The `inject process module, service proxy module` and `log record` module are the core components. Figure 1 shows the architecture of the system.

3.2 Activity Module

The `activity module` can interact with Android. The system uses activity module to read the permission lists and of each installed app. Similarly, all logs can also be access via the watch log activity. The user can comment each app through reading/writing the logs stored in `log record` module. It can also be online accessed with the `java online comment` service. With the `notify activity`, the system can notify users when suspect privacy behaviors are found.

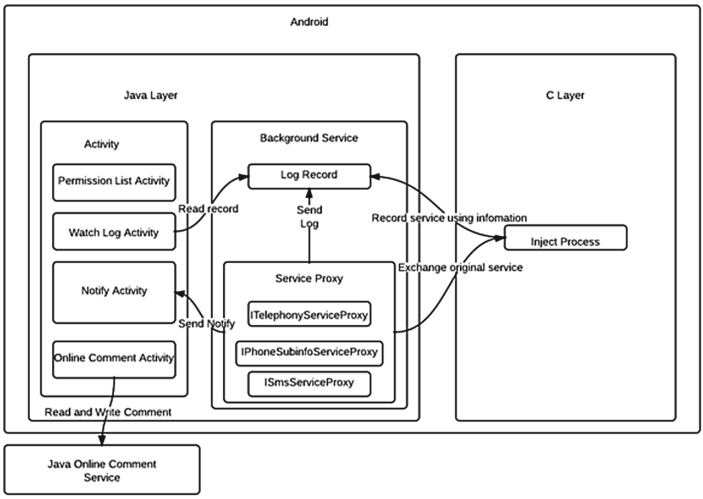


Fig. 1. Structure of real-time privacy abuse monitoring system

3.3 Inject Module

The `inject module` is used to inject third part code into the `service manager` process and replace original function, which contains two submodules: `inject program` and `hook library`. The injection process following steps: (1) `background service` module starts the inject program by forcing target process to load hook library; (2) the inject program edits the memory of target process and replace the address of original function with that of function in hook library; (3) the `service manager` loads functions in the hook library. The `inject module` is also responsible for registering binder service and returning the handle of service. The developed system can analyse and log the request of services from client apps. When `service proxy` module registers service in `service manager`, the system will record the address of binder handle. If need, it can replace the handle of proxy services. Figure 2 shows the injection process in Andriod.

3.4 Background Service Module

It contains two submodules, `log record` and `proxy service`. It first be initialized when the Android device is turned on. The background service module starts the `inject` module and then extracts some original service from `service manager` to create `proxy service`. Then, it sends the binder address of original service to proxy service and registers proxy service in `service manager`. Finally, the `log record` module will be loaded in a new thread.

The `proxy service` can analyse the requests from client apps and the suspect requests is logged in `log record` module. If some suspect behaviors are found, the `proxy service` will call `notify activity` and ask the users make

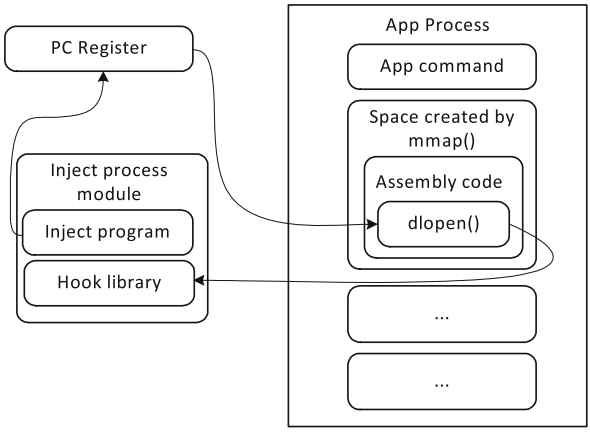


Fig. 2. Injection in an app process

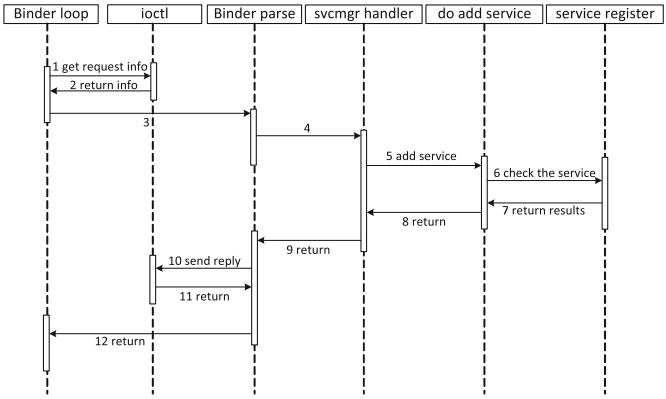


Fig. 3. Find the handle of service in `service manager`

a decision on whether to accept the request. The `log record` module can store and analyse logs received from `proxy service` and `injection module`. When the `activity module` asks for logs, the `log record` module will send a container of logs to `activity module`. Figure 3 shows the registration of service in `service manager`. Figure 4 describes the workflow of using binder in communication between proxy and service.

3.5 Java Online Comment Service Module

This module runs on a remote machine and a simple protocol is used to communicate with the `online comment` activity. The aim of this module is to enable user to online comment an apps.

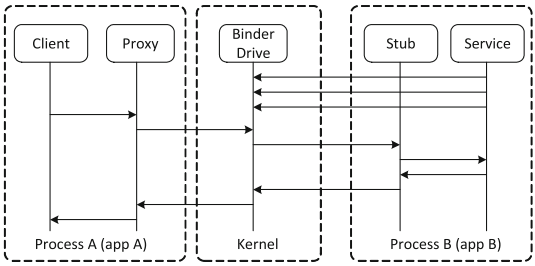


Fig. 4. Workflow of binder using in communication between proxy and service

4 System Implementation

This section addresses the details of implementation of real-time monitoring system. The system is implemented as an app and the app development environment includes Eclipse, Android Development Tools (ADT) plug-in and Android Software Development Kit (SDK). Android Native Development Kit (NDK) is used to develop the `Injection module` that is implemented in C.

4.1 API Hook

The key technology in injection is to use API hook to replace the address in Global offset Table (GOT). The injection technology is used to force target process to load hook library, where the address of target process in GOT is replaced with that of attack process. In this system, we demonstrate the workflow of app injection as shown in Fig. 5. It includes two steps:

- *Injection*, is a method that one process invades the workflow of another process. The *ptrace* function is used to force target process to allocate a space to save the injection assembly code. By doing this, the target process can be forced to load the injected.
- *GOT*, contains actual addresses of functions. By changing the address of target function in GOT of a target app, target process, it is possible to replace a function in an app.

4.2 Hook Functions

After injecting the code and replacing the address in GOT, the target process will load the hooked function instead of original target function. In Android system, the `service manager` is a name system which provides client apps service binder address according to the requested. In this work, it is found that the information of request of service from client apps can be easily fetched through the hooked `ioctl`. Actually, the handle of service can be easily exchanged here. The hooked `ioctl` can record the handle of service that created in `service proxy` module.

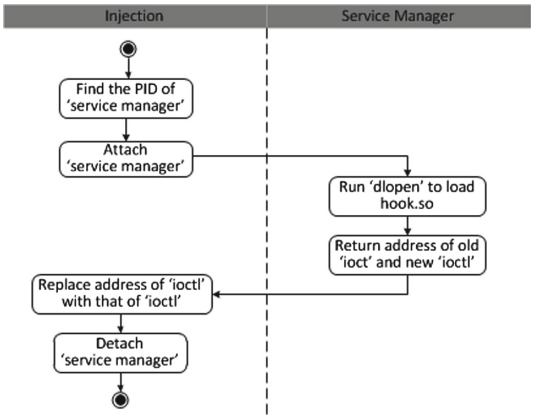


Fig. 5. workflow of injection

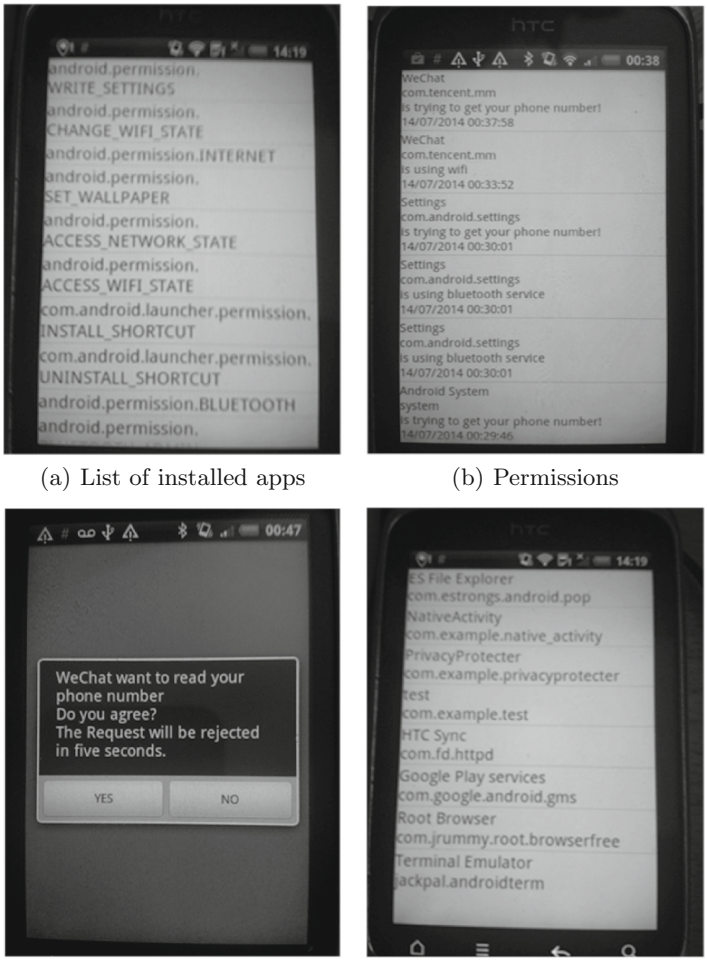
The hooked `ioctl` can load the original `ioctl` to send the handle of proxy service to client app. Then, the client apps will use the proxy service instead of the original service.

4.3 Test and Comparison

A `HTC Wildfire S A510e` mobile phone is used to test the developed system, where the Andriod version is 2.3.5. Figure 6(a) shows the list of all installed apps and Fig. 6(b) shows all the permissions of them. The system is able to real-time monitor the suspect privacy abuse behaviors, such as 'read phone number', 'read/write contacts', 'send/receive sms', etc., as defined in Sect. 2. Figure 6(c) shows an example of the logs of privacy abuse of the popular social

Table 4. Comparison with LPG and 360

	Devleoped system	360 mobile safe	LBE privacy guard
Real-time monitoring	Yes	Yes	Yes
Comprehensive protection	No	Yes	No
Injection detection	Yes	No	No
Contact access	Yes	Yes	Yes
SMS reading/writing	No	Yes	Yes
Camera using detection	Yes	No	No
Notification	Yes	No	No
Log	Yes	No	No
Online comment	Yes	No	No
Root needed	Yes	Yes	Yes



(a) List of installed apps (b) Permissions

(c) Logs of privacy abuse (d) Notification of reading phone number

Fig. 6. Developed system over HTC Wildfire S 510e

app: "WeChat", where "WeChat" is trying to read phone number and the system found this suspect behaviors and send notification to the user of the device. Figure 6(d) shows an example of notification when the system found that the app is reading the phone number.

Actually, there are a number of commercial privacy protection apps have been developed and released on Google Play that are able to real-time monitor privacy abuse, such as `Clueful Privacy Advisor (CFA), LBE Privacy Guard (LPG), 360 Mobile Safe (360), etc.` The 360 is able to monitor privacy data leakage as well as malware detection. In Table 3, we compared our developed system with the two popular commercial apps: LBE Privacy Guard

(LPG) and 360 Mobile Safe (360) in terms of suspect behaviors detection capabilities.

From Table 4 we can see that our developed system are able to detect more suspect behaviors than 360 Mobile Safe LBE Privacy Guard.

5 Conclusion

This paper analyse the potential dangerous behaviours of *app*s and proves that it is possible to real-timely monitor the privacy abuses on Android devices. An app is developed on a rooted `HTC A510S Wildfire` mobile phone to demonstrate all the technology proposed above.

Acknowledgments. This work was partially supported by the European Unions Prevention of and Fight against Crime Programme "Illegal Use of Internet" - `ISEC 2010 Action Grants (HOME/ 2010/ISEC/AG/INT-002)`.

References

1. Chen, J.: Realtime monitoring of private data abuses in android system. M.Sc. Thesis, University of Bristol, September 2014
2. Ong, J.: Report: android reached record 85% smartphone market share in Q2 (2014). Xiaomi now fifth-largest vendor. http://thenextweb.com/google/2014/07/31/android-reached-record-85-smartphone-market-share-q2-2014-report/ (2014). Accessed 4 August 2014
3. Zhou, Y., Singh, K., Jiang, X.: Owner-centric protection of unstructured data on smartphones. In: Proceedings of the 7th International Conference on Trust and Trustworthy Computing (TRUST 2014), Crete, Greece, June 2014
4. Zhou, W., Wang, Z., Zhou, Y., Jiang, X.: DIVILAR: diversifying intermediate language for anti-repackaging on android platform. In: Proceedings of the 4th ACM Conference on Data and Application Security and Privacy (CODASPY 2014), San Antonio, TX, March 2014
5. Kelly, J.: Report: 97% Of mobile malware is on android. This is the easy way you stay safe. http://www.forbes.com/sites/gordonkelly/2014/03/24/report-97-of-mobile-malware-is-on-android-this-is-the-easy-way-you-stay-safe/ (2014). Accessed 4 August 2014
6. Tang, W., Jin, G., He, J., Jiang, X.: Extending android security enforcement with a security distance model. In: IEEE 2011 International Conference on Internet Technology and Applications (iTAP), pp. 1–4 (2011)
7. Google play: clueful privacy advisor. Available from: https://play.google.com/store/apps/ (2014). Accessed 22 July 2014
8. Google play.: privacy scanner (Antispy) free. Available from: https://play.google.com/store/apps/details?id=net.hobbyapplications.privacyscanner (2014). Accessed 24 July 2014
9. Androids permissions system is broken and google just made it worse. Available from: http://www.howtogeek.com/177904/androids-permissions-system-is-broken-and-google-just-made-it-worse/ (2015). Accessed 15 February 2015

10. Gates, C.S., Chen, J., Li, N., Proctor, R.W.: Effective risk communication for android apps. IEEE Trans. Dependable Secur. Comput. **11**(3), 252–265 (2014)
11. Mobile safe. Available from: https://play.google.com/store/apps/details?id=com.qihoo360.mobilesafe (2014). Accessed 1 September 2014
12. Gargenta, A.: Deep Dive into android IPC binder framework at android builders summit. Available from: http://events.linuxfoundation.org/images/stories/slides/abs2013_gargentas.pdf (2013). Accessed 26 April 2014
13. Jiang, D., Fu, X., Song, M., Cui, Y: A security assessment method for android applications based on permission model. In: 2012 IEEE 2nd International Conference on Cloud Computing and Intelligent Systems (CCIS), vol. 2, pp. 701–705 (2012)
14. Kuzuno, H., Tonami, S.: Signature generation for sensitive information leakage in android applications. In: 2013 IEEE 29th International Conference on Data Engineering Workshops (ICDEW), pp. 112–119 (2013)
15. Wei, T.E., Jeng, A.B., Lee, H.M., Chen, C.H., Tien, C.W.: Android privacy. In: 2012 IEEE International Conference onIn Machine Learning and Cybernetics (ICMLC), vol. 5, pp. 1830–1837 (2012)
16. Wu, L., Du, X., Fu, X.: Security threats to mobile multimedia applications: camera-based attacks on mobile phones. IEEE Commun. Mag. **52**(3), 80–87 (2014)
17. Cannon, T: Android data stealing vulnerability. Available from: http://thomascannon.net/blog/2010/11/android-data-stealing-vulnerability/ (2010). Accessed 11 April 2014
18. Jiang, X.: Android 2.3 (Gingerbread) data stealing vulnerability. Available from: http://www.csc.ncsu.edu/faculty/jiang/nexuss.html (2011). Accessed 11 April 2014