Spyware Detection based upon Hybrid Approach In Android Smartphones

Sumit Sharma
Assistant Professor, CSE Department,
Chandigarh University, Gharuan
Mohali, India

Parmjit Kaur
Dept. of Computer Science
CGC Group of Collages, Gharuan
Mohali, India

ABSTRACT

Smart phones, which were once a luxury product has now become a household product. This transformation has been due to the vast amount of functionalities which a smart phone provides in just a single device. Smartphone OS, such as Android, is an Open Source mobile platform that enables us to install third party applications. Due to large amount of applications that are available on the app store, it becomes very difficult for a user to distinguish between a malware free and malware containing application. In this paper, we are proposing a hybrid approach for detection of spyware genre applications out of all the installed applications.

Keywords

Android Privacy, Spyware, Android Permission Database, Frame Layout, Static Source Code Analysis

1. INTRODUCTION

In recent days, usage of Mobile Devices has become indispensable part of every human-being. Smartphones has gained more popularity than the all purpose personal computers. According to the CISCO report, till the completion of 2014, quantity of mobile-connected devices will go beyond the population of world. Nowadays, Smartphones represent only 27% of the mobile devices. But till the end of 2018, Smartphones will represent more than 50% of the total mobile devices [19]. Another reason for Smartphones usages is its expandable memory up to 128 GB (up to 64 GB Card Slot and 32/64 GB internal storage and 3 GB RAM) [9]. Hackers become more attractive towards smart phones than PC, and particularly the Google’s Android OS. An Android is an open source[1] OS (operating system) developed by the Open Handset Alliance and held by Google Inc [20]. Android becomes more popular because Google Play alone had 500,000 applications that are classified as free or paid, in May 2012.

In 2012, *A.PINTO* gives a report *Android Malware 400% increase* in which he describe how Android Malware has increased over the time. A recent study conducted by a MIT organization found that there is increase of about 400% in Android malware since summer 2010. How every time you open a website first a fraudulent website does come up? Yes, it is the malware which piggyback some other application and then starts performing stealthily in the background to harm user’s privacy. The main purpose of this report is to make the reader aware of the impressive grow of malware being targeted to smartphones [3].

Attacks are introduced in the system, when an attacker made the third party application in which he introduces the malicious code and upload it on the Google Play Store. Infected application is downloaded and installed by Android User on their smart phone. At the time of installation of an android app, the user is presented with all the permissions the application is supposed to use. But as this permission denial is not applicable to single permissions, the user has to either completely discontinue the installation process or accept all of them. Now, while the user makes use of that application and in the back-ground the app can mishandle the permission granted to it and steal the private data of the user. This type of misuse takes advantage of the Android Permission Model or we can say add the malware or malicious code in the system [10].

In this Paper, we describe how malwares are spreading through internet onto the mobile devices and henceforth proposing a three way spyware detection mechanism to enhance the security of user private data.

2. OVERVIEW OF ANDROID

2.1 Application and components

The programming language used for the development of android applications is Java, compiled into byte codes and then using dx converter, it will be converted to a dalvik executable (.dex) file. After that it will further be compiled into android package (apk) file, which can be installed on the android devices [8].

Intents are scattered between four components of application which is required for the development of Android applications. Detail of these major components is described below and developer must declare it in AndroidManifest.XML file [7].

2.1.1 Activity

Activity is a visible process which works in the foreground of the mobile screen and interacting with the user with the help of user interfaces [7, 20].

2.1.2 Service

Service is a background process which is not visible to the user or we can say that works without a screen User Interface. For e.g., it can perform the operations in background process like playing the music or downloading the file [7].

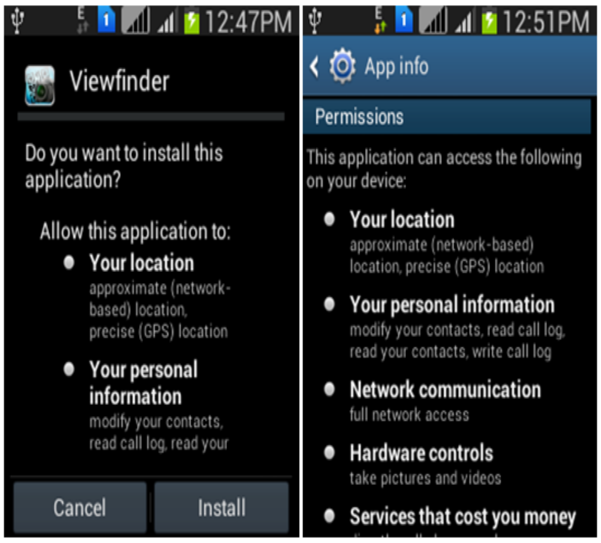
2.1.3 Broadcast Receiver/Receiver

Receiver is a component for android application development which is used to receive the Broadcast Intent from the Operating System. For e.g., if there is need to identify the operating system boot event then it has to obtain Broadcast Intent with the help of Broadcast Receiver [7].

After receiving Intents, broadcast receiver sent it to numerous applications. When it received the suitable Intent, Receivers are activated and then perform in the background process to control the event. The message passing mechanism of android is using the intents at the core. The major categories of intents being used are: *normal, sticky, and ordered*. Normal intents being broadcasted to all the receivers; ordered intents are bound to be circulated one by one leaving it to any application to stop the propagations whereas sticky intents remain active even if received by all the receivers for the sake of rebroadcasting to upcoming receivers [3].

2.1.4 Content Provider/Provider

Content Provider/Provider is a database i.e. it supplies data storage for applications and share data with other applications. For e.g., call history and list of contacts etc are managed [7].



12:47PM

Viewfinder

Do you want to install this application?

Allow this application to:

- Your location
  approximate (network-based) location, precise (GPS) location
- Your personal information
  modify your contacts, read call log, read your

Cancel Install

12:51PM

App info

Permissions

This application can access the following on your device:

- Your location
  approximate (network-based) location, precise (GPS) location
- Your personal information
  modify your contacts, read call log, read your contacts, write call log
- Network communication
  full network access
- Hardware controls
  take pictures and videos
- Services that cost you money

Fig 1: Android permission request (left) and the permissions of an installed application (right) [12]

Our main focus is to provide security in android smartphones by implementing a permission-based security mechanism. All the permissions are declared in AndroidManifest.XML file and give a list of default permissions in android.Manifest.permission class, and it also permits us to declare new permissions in AndroidManifest.XML file. In an application, the new permissions should be defined and started into a system when installation of application is completed on the system. Permission is authorized at the installation time. To complete its task, there is requirement of list of permissions for an application. This shows the set of requested permissions on the screen with the intention that the user can review them. As we know Permissions are based upon all or nothing basis so if the user allows then the installation process starts otherwise the application is not installed on the smartphones. Accordingly, the security principle implements the system in which they define that, *operations can perform by application only if they are permitted to do so or it would affect others parts of the system if they have right privilege to do so*[11].

2.2 Android Manifest file

The Android application is stored as .apk file as signed archive file. In .apk file, there is compiled codes of an application, binary resources and also signed with the developer's certificate. This package also contains an XML file, called the AndroidManifest.XML file [11]. The manifest also describe extra metadata for the application like icons and the version number of the application. This XML file is read by the Android system during installation of the application [18]. The information included in the manifest file is as follow:

- Set of application sub-components: In Manifest File, application is developed by a group of sub elements for components like activity, service, content provider, and broadcast receiver which we have defined earlier [11]. All Activity, Service and Provider are statically announced whereas Receiver is declared dynamically within the file [18].
- Set of permission declarations: Permission element is used to declare permission in an application like Access Coarse Location for accessing information about location of device. The permission is appended to a system after the installation of application [11].
- Set of permissions expected to be granted: An application uses <uses-permission> element for listing all the permissions which are required to complete its task. All the lists of permission are requested and shown on the screen while the installation time. It is depend upon the user either allows the installation by granting all the requested permissions or aborts it. In XML file, different terms are used for the requested permissions at the installation time i.e. *requested-permissions* and *use-permissions* is used for the permissions after being granted in the application.
- Set of permissions used for protection: In the XML file, android: permission attribute is used by application element and component elements. All the permissions required by the application are listed down in the AndroidManifest.XML file. The permission access could be asked for either the whole application i.e. in the application attribute or it could be asked by the developer for individual components of the application i.e. by the services, activities or broadcast receivers registered in the application. That is the reason there is need of checking the permissions asked by the application in the manifest file as well as evaluating the level at which it is asked [11].

2.3 Protection level

The protection level determines how the permission is granted. The Android supports 4 protection levels for permissions: level 0 to 3, i.e. normal, dangerous, signature, and signatureOrSystem [11].

➢ Level-zero (0) permissions or named as normal permissions which create a low-risk factor and automatically granted by the system because these are not seen or we can say it is hidden in a folded list of option on the screen and normally it only affect the applications scope. Example are VIBRATE and SET WALLPAPER etc [10].

➢ Level-one (1) permissions or dangerous permissions which poses a higher-risk permissions and it shown on the screen at the installation time of application, For example permit costly access to service like starting phone calls or usage of Internet or usage of devices sensors, or sensitive user data. A noticeable permission is the permission to examine the log files on dangerous-level [10].

➢ Level-two (2) permissions or signature permissions are only allowed if the application that is being installed, is signed by the private key corresponding to the same certificate as the application that initially defined this permission. These permissions can be used by developers, e.g., to distribute information between their particular application, while avoiding applications of other developers to gain access to this information. So even if the user would consent, signature permission cannot be allowed to applications signed with the private key corresponding to a different certificate [10].

- Level-three (3) permissions or signatureOrSystem permission can be allowed by the system to applications that are included in the systems image, or by applications that have been signed with the identical certificate as the system image. Highest category of permissions are preserved for a handset of manufacturers and Operating System developers. Representatives of the highest category are the permissions to install new application (packages) or to change security settings [10].

3. MALWARE AND ITS TYPES

Moser et al explained the definition for Malware in which he defines that the Software that "*deliberately fulfills the harmful intent of an attacker*" is normally referred to as malware or malicious software. Malware is generated by merging the words 'malicious' and 'software'[4]. There are different types of malware like "adware", "virus", "worm", "spyware" or "Trojan horse" etc which are used to categorize malware samples that show identical malicious behavior [1].

In this section, a brief summary for the different groups of malware programs which have been observed in the wild.

3.1 Viruses

Small program with dangerous intent which has capability to repeat itself is referred to as Virus. Virus code gets executed automatically, Whenever file is run. It may spread through any medium like network or corrupted media like USB drives, floppy disks to uninfected computer [4].

3.2 Worms

Worms are programs which has capability of replicating their programs. It makes use of network to transmit replica of itself to other systems invisibly without authorization of user. It may also affect the network by consuming its bandwidth. It does not require the support of any file unlike virus and worms. It may encrypt files, delete files, or send junk email. Examples of Worms are My Doom, Blaster, Sasser, Melissa etc [4].

3.3 Spyware

It is a combined phrase for software which observes and collects personal information about the user such as the email address, credit card number, pages frequently visited, and key pressed by user etc. Spyware usually comes into a system when free or trial softwares are downloaded [4].

Spyware are basically of two types: Commercial spyware and malicious spyware. Commercial spywares are oriented towards targeting a specific user and the installation process of the spyware is manual whereas the malicious spyware covertly steals data and transmits it to the third party just as the desktop version of the spyware works.

CarrierIQ is an example of commercial spyware which has been used by many handset manufacturers and network operators. This software is pre-installed on the mobile devices to gather data about all the activities done on the device like web searches, firmware, battery performance and application performance to allegedly increase the customer satisfaction by logging dropped calls and similar information. The issues related to CarrierIQ are that the user is not aware whether it is installed or not, furthermore even if the user knows about the existence of the spyware he/she cannot uninstall it without rooting the whole operating system. Additionally, there was no way for the normal users to identify what information the vendors deemed necessary to increase the user experience [12].

3.4 Adware

When malicious software is installed or application is used, Adware itself plays, displays, or downloads advertisements to a computer. Advertising-supported software or Adware code is mostly embedded into free software. Peer-to-peer clients and free games such as KaZaa, BearShare etc are examples of adware [4].

4. DETECTION PROCESS OVERVIEW

In this process, we propose an approach which involves three way mechanism to detect the spyware so that we can enhance security to the end-users' private data. The pre- requisite of the approach involves two initial setup steps where the location of the installed application is found and then the apk file is reverse engineered. The hybrid approach of using all three approaches collectively is explained here step by step:

4.1 Application Description

As we know, when developer develops any application then they make description about that application so that end-user can easily understand the application features and its functionality. When end-user checks any application from play-store then they visualize the description about that particular application. Our main focus is on description of the application because description is the only source that helps the end-user to understand about that applications functionality.

In Android OS permissions are declared in XML File. Permissions are all or nothing i.e. either users allow all requested permissions or refuse all of them without installation of application. The information is given to the end-user about those permissions, which their application needs according to the developer thinking, but does not gives information about those permissions which the application actually uses behind the user interface [17]. Because unauthorized user declare the extra permissions for their personal usage or for stealing the personal information of end-user and end-user cannot understand it. Our main purpose is to detect the spyware and adware according to the given description so that we can prohibit the harmful impact of unauthorized users and provide safety to end-users. This can be done by mapping of required Permission of the application with Static Permission Database [16].

4.2 Graphical Layout

A layout defines the visual structure for a user interface. Android Development Tool provides a lot of features to permit us to design and make our application's user interface. A lot of functionalities are accessed by opening one of our application's XML layout files in Eclipse which is in the graphical layout editor [13].

An Android layout is a class that controls organizing the way its children show on the screen. There are different types of standard layouts like RelativeLayout, LinearLayout, FrameLayout, AbsoluteLayout, TableLayout. In this Paper, we describe only FrameLayout which is related to our work [14]. Frame Layout is designed in such a manner so that it can show a single item to block out an area on the screen. It is applied to grasp a single child view, because it can be difficult to arrange child views in a way that's scalable to different screen sizes without the children overlapping each other [15]. FrameLayout becomes more useful when elements are unseen and displayed programmatically. Attribute android:visibility is used in the AndroidManifest.XML to hide particular elements. It can call by using the code setVisibility to achieve the same thing. There are three types of visibility values are visible, invisible (does not display, but still takes up space in the layout), and gone (does not display, and does not take space in the layout) [14].

Due to its useful feature i.e. elements are hidden and displayed programmatically, unauthorized user take the advantage of this feature and steal the personal information of user. But some malicious applications uses this feature and place the Camera preview object in a Frame Layout and place another child view to block the visibility of camera preview which could hence use either the front or rear camera to take user images. This may be compromising to the user at large number of situations. In this paper our second proposed way in which we check the frame layout and all of its hidden element.

4.3 Source Code Analysis

In source code analysis, we check all objects i.e. declared in its java file are properly used according to their description or not. If not then there is chances that malware is existed in the application and use the personal information for their own purpose.

5. CONCLUSION

In this paper, we are proposing the three way method by permission mapping, Frame layout detection and source code analysis to indicate presence of spyware in an application. A weight is attached to the application at each level based upon the severity and if the application crosses the threshold limit an notification is thrown to the user. This enhanced user understanding for making right decision about an application and enhanced the security of user's personal data.

6. REFERENCES

[1] Egele, Manuel, Theodoor Scholte, Engin Kirda, and Christopher Kruegel. "A survey on automated dynamic malware-analysis techniques and tools." *ACM Computing Surveys (CSUR)* 44, no. 2 (2012): 6.

[2] Blasing, Thomas, Leonid Batyuk, A-D. Schmidt, Seyit Ahmet Camtepe, and Sahin Albayrak. "An android application sandbox system for suspicious software detection." In *Malicious and Unwanted Software (MALWARE), 2010 5th International Conference on*, pp. 55-62, IEEE, 2010.

[3] A.pinto, "Android Malware 400% increase" [Available online]: http://cybersecurity.mit.edu/2012/11/android-malware-400-increase/.

[4] Vinod, P., R. Jaipur, V. Laxmi, and M. Gaur. "Survey on malware detection methods." In *Proceedings of the 3rd Hackers' Workshop on Computer and Internet Security (IITKHACK'09)*, pp. . 74-79. 2009

[5] Mohd Afizi, Mohd Shukran, Wan Sharil and Sham Bin Sharif, "Android Augmented Reality System In Malaysia Military Operations – Unit Positions." in *Australian Journal of Basic and Applied Sciences*, Vol. 6, Issue 8, p79, Aug2012.

[6] Sharma Sumit, Rohitt Sharma, Paramjit Singh, and Aditya Mahajan. "Age Based User Interface in Mobile Operating System." *arXiv preprint arXiv:1205.1687* (2012).

[7] Agematsu, Harunobu, Junya Kani, Kohei Nasaka, Hideaki Kawabata, Takamasa Isohara, Keisuke Takemori, and Masakatsu Nishigaki, "A Proposal to Realize the Provision of Secure Android Applications--ADMS: An Application Development and Management System." In *Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS), 2012 Sixth International Conference on*, pp. 677-682, IEEE, 2012.

[8] Nema Behutiye, Woubshet. "Android ECG Application Development." (2012).

[9] Features of Samsung Galaxy S5 [Available Online http://www.gsmarena.com/samsung_galaxy_s5-6033.php

[10] Satpal, Nitin B. "Enhancing Permission Model of Android." PhD diss., Indian Institute of Technology Bombay, 2013.

[11] Shin, Wook, Shinsaku Kiyomoto, Kazuhide Fukushima, and Toshiaki Tanaka. "A formal model to analyze the permission authorization and enforcement in the android framework." In *Social Computing (SocialCom), 2010 IEEE Second International Conference on*, pp. 944-951. IEEE, 2010.

[12] Boksasp, Trond, and Eivind Utnes. "Android apps and permissions: Security and privacy risks." (2012).

[13] The Android Developer's Guide Graphical Layout [Available Online] http://developer.android.com/tools/help/adt.html#graphical-editor/

[14] Learn Android- Tutorials For Developing With Android [Available Online] http://www.learn-android.com/2010/01/05/android-layout-tutorial/

[15] The Android Developer's Guide – Frame Layout [Available Online] http://developer.android.com/reference/android/widget/FrameLayout.html

[16] The Android Developer's Guide - Android Manifest Permissions [Available Online] http://developer.android.com/reference/android/Manifest.permission.html

[17] Au, Kathy Wain Yee, Yi Fan Zhou, Zhen Huang, Phillipa Gill, and David Lie. "Short paper: a look at smartphone permission models." In *Proceedings of the 1st ACM workshop on Security and privacy in smartphones and mobile devices*, pp. 63-68. ACM, 2011.

[18] Android Development Tutorial [Available Online] http://www.vogella.com/tutorials/Android/article.html

[19] Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2013-2018, White Paper, February 5, 2014.

[20] Kaur, Parmjit, and Sumit Sharma. "Google Android a mobile platform: A review." In *Engineering and Computational Sciences (RAECS), 2014 Recent Advances in*, pp. 1-5. IEEE, 2014.