A literature Review on Android Permission System

Ahmed Ben Ayed

***Abstract* - Android uses a permission-based model to protect its users' information and system resources. This permission-based system has been the center of many researchers' interest; they have been used to identify malicious behaviors and ultimately could help identify malicious applications. This study is not intended to create an anti-malware solution or method; instead it offers a literature review on Android permissions system and illustrates previous work that has been studied using permissions to identify harmful applications. This study could be used as a source to better understand the Android architecture and its permission-based system.**

***Index Terms*—Android Security, Permission-based systems, malware detection.**

I. INTRODUCTION

Android is leading the market share in mobile applications downloads since 2011 [1]. Android is an open source platform, it supports third party applications development that technically could have full access to data and hardware. To secure users' information and resources, the platform uses a permission-based model. This paper is offering a brief explanation of the Android system and its permission model, and it will illustrate the work that has been done in the past using permissions to detect malicious applications.

II. BACKGROUND ON SMARTPHONES

Reference [2], defines smartphones as a predominantly communication devices, with additional computing power built in.

In The United states only over eighty-seven percent of Americans own a cell phone and more than fifty percent use their phones to access the internet [3]. Smartphone usage is experimenting growth. Smartphone ownership consists of forty-five percent in 2012 [3], and rose to sixty-four percent as of October 2014 [4]. The computing power building into smartphones enables it to offer a big range of services as accessing the internet, playing games, and using the phone to store personal data as photo, videos, calendars events….etc. Those ranges of features offered by smartphones make it very attractive to users and a good and cheap alternative to personal computers [4], found that ten percent of Americans that own a smartphone do not have internet at home, and fifteen percent of smartphones owners depends mainly on their phones to access the internet. The accessibility, convenience, and the wide range of features smartphones offer have made it appealing not only to personal use, but it was extended to be used in the business environment.

III. THE ANDROID PLATFORM

According to [5], Android is an open source mobile device platform-based on the Linux operating system, it was developed by Google and the Open Handset Alliance. Android is considered to be the largest installed base of any mobile platform, and it is growing tremendously, every day a new million users are activating their Android devices for the first time [6]. It provides an API that give programmers the authority of developing third party applications to be used on the platform. Every application has its own Linux user ID and its own virtual machine, so applications' code runs in isolation from other applications [7].

A. Android Architecture

Android is architected in the form of a software stack of four main layers: Applications, Application Framework, Libraries & Android Runtime, and Linux Kernel [8], the following diagram shows the four layers of the Android system.

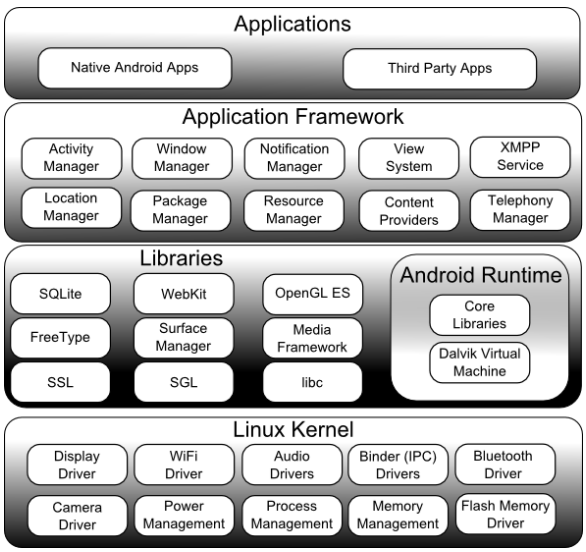

Figure 1: Android's Conceptual Architecture

B. Android Permission Model

Android uses a permission model that requires an application to request the permissions it needs to perform its activities before it gets installed as shown below in figure 2.

***Ahmed Ben Ayed**, Department of Engineering and Computer Science, Colorado Technical University, Colorado Springs, Colorado, USA, Phone +1-510-200-2167*

Therefore, the user is notified during installation of the permissions the application will receive. If the user doesn't feel that the application should receive such permission s/he can stop the installation. When the application is installed it cannot be able to request any more permission. See Figure 2 for an example of a permission request screen.

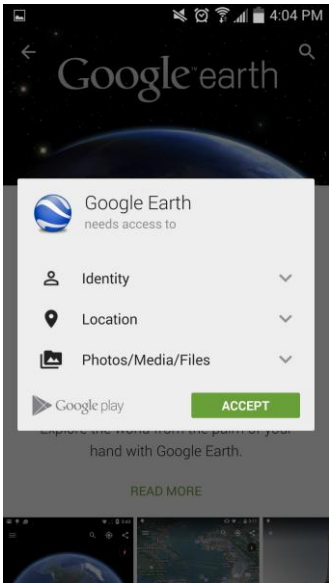

Figure 2: Permission disclosure prior to installation

An Example of permissions is accessing the internet, accessing the location, sending and receiving SMS, accessing files and pictures stored in the device, among many others. The Permission Based Model provides a controlled access to various system resources and restricts access on others.

Android 4.4 defines 152 permissions categorized into three protection level that determines how difficult the permission is to get. The basic Android permission access control mechanism could be illustrated in figure 3.

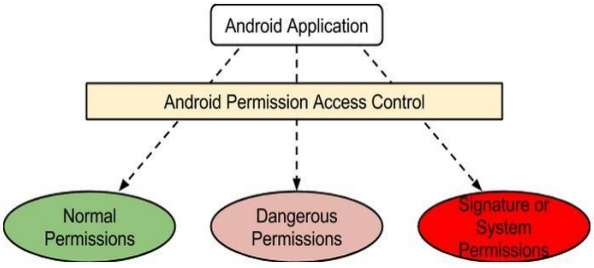

Figure 3: Basic Android Permission Access Control Mechanism

- Normal Permissions are granted to any application that requests them, those kinds of permissions are not considered harmful to the user, however they could annoy them [9]. An example of a normal permission is the permission SET_WALLPAPER_HINTS that allows applications to set the wallpaper hints.
- Dangerous Permissions are granted only after the user confirmation during the installation, if the user denies the permission the application wouldn't be installed. Dangerous Permissions have access to potentially harmful API calls, as those that cost the user money like SMS and Calls service as the SEND_SMS and CALL_Phone permissions.
- Signature or System permissions are granted if the application requesting those permissions meets certain criterions, those permissions are hard to obtain, and they are given only to applications that are signed by the same developer that defined the permission. Signature of System permissions are considered to be the most dangerous permissions, and access to it is regulated. Some example of dangerous or Signature Permissions are able to backup or delete data as DELETE_CACHE_FILES, and DELETE_CACHE_PACKAGES.

C. Android Manifest File

The Manifest is the file where permissions are declared. Every application must have a manifest file in its root directory. Those permissions declared as strings in the application's manifest file [7], and cannot be changed or modified after installation. The manifest contains application's specific information such as name, icon, and application compatible setting…etc. an example of a manifest file is showing in figure 4.

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
        package="com.rafaelkhan.android.download" android:versionCode="1"
        android:versionName="1.0">

        <uses-sdk android:minSdkVersion="7" />
        <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
        <uses-permission android:name="android.permission.INTERNET" />
        <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />

        <application android:icon="@drawable/icon" android:label="@string/app_name">
                <activity android:name=".DownloadMain" android:label="@string/app_name"
                        android:theme="@android:style/Theme.Light">
                        <intent-filter>
                                <action android:name="android.intent.action.MAIN" />
                                <category android:name="android.intent.category.LAUNCHER" />
                        </intent-filter>
                </activity>

        </application>
</manifest>
```

Figure 4: Example of an Android manifest file

D. Permission Enforcement in Android Applications

Most of the permissions are enforced at the Android Framework level in the android software stack. Some of those permissions are enforced at the kernel level using supplementary group ID [10]. Supplementary group ID allows an application if it's a part of a group to receive all the privileges of that specific group. The groups are assigned accordingly using the permissions declared in the application's manifest file. As shown in figure 5, Camera, Bluetooth, and Internet are some examples of permissions that are enforced in the kernel Level using group ID, other permissions as SMS, and accessing the contact list are enforced in the Android Framework level. Figure 5 shows the enforcement mechanism as explained earlier.

ISSN: 2278 – 1323

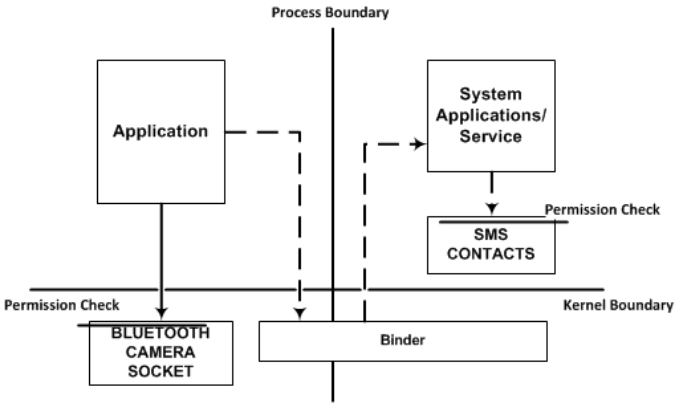

Figure 5: Permissions' Enforcement Mechanism

IV. Previous Work done using Android Permissions To identify Malicious Applications

Table 1: Noticeable malware detection studies using Permissions

Analysis method	Research Paper	Description
Static Analysis	[11]	Presented VetDroid a dynamic analysis platform for reconstructing sensitive behaviors from a permission use prospective. The tool analyzes how the applications use permissions to access sensitive information. A real android malware was used to test Vetdroid, and it showed that it can reconstruct fine-gained malicious behavior to ease malware analysis.
	[12]	Introduced a new framework, which enforces a fine-grained permissions model. A new service layer in between Android applications and underlying service managers was created and assigned the responsibility of enforcing access to permissions. The study found that 62% of the occurrence of the permission that were studied could be replaced by a fine gained permission without affecting the application's ability to perform its work.
	[13]	Proposed a permission based foot printing schema to identify certain malware applications of known families, Then Zhou *et al*, applied a heuristics-based filtering scheme to identify malware or malicious applications that belong to an unknown family strain.
Dynamic Analysis	[14]	Classified Android malicious applications using a machined learning algorithm. The data-set used contained 47 applications, were 24 are considered benign applications. The result obtained was 81.25% malware detection rate, but the false positive rate was reported as very high. Permissions, Binder properties, CPU usage, and more features were used to create feature vectors.

V. Conclusion

The Android platform model gives a lot of privileges to applications; however, the permission model could reduce the risk of unauthorized access if the user is aware of the security threat. This paper could be an attempt to improve the users' understanding of the Android permission system and what it allows the applications to access and control.

REFERENCES

[1] J. Kendrick. "Latest smartphone market share numbers: Apple is flat, google going strong." Retrieved from *http://www.zdnet.com/blog/mobile-news/latest-smartphone-market-hare-numbers-apple-is-flat-google-going-strong/2387, May 2011.*

[2] Beale, R. (2005). "Supporting social interaction with smartphones." *IEEE Pervasive Computing.*4(2), 35-41.

[3] J. Brenner, "PEW Internet Mobile. PEW Internet & American Life Project" Retrieved from *http://pewinternet.org/Commentary/2012/February/Pew-Internet-Mobile.aspx*

[4] A. Smith, "US Smartphone use in 2015" *Pew Research Center.* Retrieved from http://www.pewinternet.org/files/2015/03/PI_Smartphones_0401151.pdf

[5] C. Nimodia, and H. Deshmukh. "ANDROID OPERATING SYSTEM" *Software Engineering*, 3(1), 10. 2012

[6] Android. "Android, the World's most popular mobile platform" Retrieved from *http://developer.android.com/about/index.html*

[7] Android. "Application Fundamentals" Retrieved from *http://developer.android.com/guide/components/fundamentals.html*

[8] W. Hu, D. Han , A. Hindle, and K. Wong. "The build dependency perspective of android's concrete architecture" *The 9th Working Conference on Mining Software Repositories*, page to appear, 2012.

[9] A. Felt, E. Chin, S. Hanna, D. Song, and D. Wagner. "Android permissions demystified." *In Proceedings of ACM Conference on Computer and Communications Security* (CCS), 2011.

[10] A. Stephen, W. Rago, and R. Stevens. "Advanced programming in the unix environment: Second edition." *Addison Wesley Professional*, 2005

[11] Y. Zhang, M. Yang, B. Xu, Z. Yang, G. Gu, P. Ning, X. S. Wang, and B. Zang. "Vetting undesirable behaviors in android apps with permission use analysis." *In Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security*, pages 611–622. ACM, 2013.

[12] J. Jinseong, K. Kristopher, V. Jeffrey, F. Ari, J. Foster, and M. Todd. "Dr . Android and Mr . Hide : Fine-grained Permissions in Android Applications" *Categories and Subject Descriptors*. pages 3–14. 2012.

[13] Y. Zhou and X. Jiang. "Dissecting Android Malware: Characterization and Evolution." *2012 IEEE Symposium on Security and Privacy*, (4):95–109, May 2012.

[14] B. Amos, H. Turner, and J. White. "Applying machine learning classifiers to dynamic Android malware detection at scale." *2013 9th International Wireless Communications and Mobile Computing Conference (IWCMC)*, pages 1666–1671, July 2013.

Ahmed Ben Ayed has received his Bachelor of Science in Computer Information Systems, Master of Science in Cyber Security and Information Assurance, and currently pursuing a doctorate degree in Computer Science at Colorado Technical University, his research interest are Android Security, Pattern recognition of Malicious Applications, Machine Learning, Cryptography, Information & System Security, and Computer networks.

ISSN: 2278 – 1323