Security Aspect in Android

P. Sahu[1], S.K.Balabantaray[2]
[1]Asst. Prof. Department of Computer Science and Engineering
[2]Asst. Prof. Department of Computer Science and Engineering
Raajdhani Engineering college, Bhubaneswar,odisha

Abstract

The smart phones usage has been increased rapidly over the last decade. Because of their mobility and connectivity, smart phones are growing thrice as compared to Personnel Computers. Android is a mobile device operating system platform for smart phones, which is growing very fast. There are many security concerns in the Android smart phones related to permissions in Apps. Android is having some negative gaps regarding security. One of the main security related gap is its Permission level through which Apps are gaining access to the devices hardware and software. The Apps access can sometimes make a security issue, which is not acceptable for the end-users and this security issue tends users' information leakage. Most of the time users are granting permissions while installing Apps but do not know about the permission requested by the Apps, which is a gap itself and this may lead for misusing user's personal information. In this paper, a number of vulnerabilities are explored in Android permission level and provide an approach for better security in Android Platform. An Attack Scenario is developed successfully for permission-based attacks in the android platform and provides the countermeasures for it.

Keywords: *Android, Security, Privacy, Smartphone*

1. Introduction

Since the first introduction in 2008, Android has gained a tremendous number of users over the last few years. Smartphones are the fastest growing technology market segment. According to Gartner [1], a technology research and advisory firm, 1:8 billion devices running on Google's Android OS were shipped in 2018 alone, marking its 80 percent mobile market share. Attributing to this fast-pace increament is the proliferation of Android apps, which provides an ever-growing application ecosystem. Officially reported by Android Google Play Store, the number of apps in the store has reached over 1:6 million in early 2015, surpassing its major competitor Apple Apps Store [2].

Mobile applications are essential to the smartphone experience. Mobile applications are getting increasingly sophisticated, robust, life-engaging, and privacy-intrusive. The market offers a wide variety of applications ranging from entertainment, productivity, health care, to online dating, home security, and business management [3]. Users depend more and more on mobile devices and applications.

As mobile applications are gaining increasing popularity among users, the privacy and security of smartphone users become a concern. Due to the large user base, smart devices are used to store sensitive personal information more frequently than laptops and desktops. As a consequence, a malicious thirdparty app can not only steal private information, such as the contact list, text messages, and location from its user, but can also cause financial loss of the users by making secretive premium-rate phone calls and text messages [4]. At the same time, the rapid growth of the number of applications on Android markets makes it hard for app market places, such as Google App Store for example, to thoroughly verify if an app is legitimate or malicious. As a result, mobile users are left to decide for themselves whether an app is safe to use. In addition, unlike iOS, Android device owners do not have to root or ”jailbreak” their devices to install apps from ”unknown sources”.

This gives Android users broad capability to install pirated, corrupted or banned apps from Google Play simply by changing a systems setting. This provides further incentive for the users to install third-party applications, but exposes their privacy to significant security risks [5].

The exponentially increasing number of Android applications, the unofficial apps developers, and the existing security vulnerabilities in Android OS encourage malware developers to take advantage of such vulnerable OS and apps and steal the private user information to inadvertently harms the apps markets and the developer reputation [6]. Moreover, since Android OS is an open source platform, it allows the installation of third-party market apps, stirring up dozens of regional and international app-stores such as PandaApp [7] and GetJar [8]. Android malware can gain control of device, steal private information from users, consume excessive battery, use telephony services to steal money from users' bank accounts, and even turn the device into a botnet zombie.

There are a large variety of Android vulnerabilities and they can occur in any layers of Android OS stack, such as application layer or framework layer. Vulnerabilities also appear in benign apps through the accidental inclusion of coding mistakes or design flaws.

As we described before, the flawed Android OS provides a fertile ground for attackers. There are a variety of security issues on Android phones, such as unauthorized access from one app to the others (information leakage), permission escalation, repackaging apps to inject malicious code, colluding, and Denial of Service (DoS) attacks

2. Android OS and Applications Architecture

In this section we describe the architecture of the Android OS and its applications. Android is being developed and maintained by Google and promoted by the Open Handset Alliance (OHA). Android OS is placed on top of the Linux kernel and it includes the middleware, libraries and APIs written in c language, and application software running on an application framework which includes Java-compatible libraries. Android's source code is released by Google under open source licenses

2.1 Framework Architecture

Android operating system is a stack of software components, which is roughly divided into five sections and four main layers as shown in the Figure 1. Android OS layers and components are explained as below

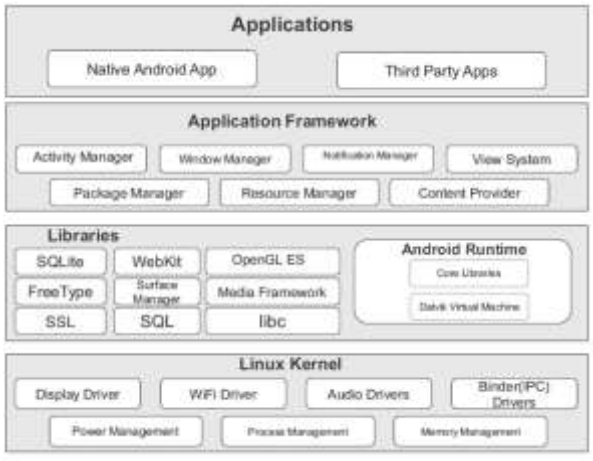

Figure 1: Android operating system architecture

2.1.1 Applications

Application layer is located at the top of the Android software stack. These comprise both the preinstalled apps provided with a particular Android implementation and third-party apps developed by individuals (unofficial) app developers. Examples of such apps are Browser, Contacts Manager, and Email apps.

2.1.2 Android Runtime:

This section describes a key component called Dalvik Virtual Machine (DVM), which is a Java Virtual Machine (JVM) specially designed and optimized for Android. Dalvik VM takes advantage of Linux core features such as multi-threading, multitasking execution environment and memory management, which is intrinsic in the Java language. Dalvik VM gives power to apps to run as a process directly on the Linux kernel and within its own VM (sandboxed). Since Dalvik is using JVM, it provides users with a set of libraries and APIs to develop Android apps predominantly using Java programming language. The standard Java development environment includes a vast array of classes that are contained in the core Java runtime libraries.

2.1.3 Libraries:

The Android's native libraries were developed on top of the Linux kernel. This layer enables the device to handle different types of data. It provides different libraries useful for the well-functioning of Android operating system. These libraries are written in C or C++ language and were developed for a particular hardware. Examples of some important native libraries include the open-source Web browser engine WebKit used to display HTML content, the well-known library libc, SQLite database engine used for data storage purposes, OpenGL used to render 2D or 3D graphics content to the screen, Media framework used to provide different media codecs, and SSL libraries for Internet security.

2.1.4 Kernel:

The Linux kernel is the fundamental layer of the entire system. This layer is customized specially for the embedded environment consisting of limited resources. The whole Android OS is built on top of the Linux kernel with some further architectural changes made by Google. This section also acts as an abstraction layer between the hardware and other software layers. Linux kernel provides the basic system functionality such as process management, memory management and device management. Linux kernel also provides an array of device drivers which make the task easier while interfacing the Android with peripheral devices.

3. Android Security Issues and Threats

Android security is built upon a permission-based mechanism which regulates the access of third-party Android applications to critical resources on an Android device. Such permission-based mechanism is widely criticized for its coarse-grained control of application permissions and the inefficient

permission management, by developers, marketers, and end-users. For example, users can either accept all permission requests from an app to install it, or not to install the app. This type of permission management is proved to be undesirable for the devices security. In this section, we discuss the main security issues of the Android, which leads to user information leakage and puts the user's privacy in jeopardy [9].

3.1 Information leakage

In current Android architecture design, apps are restricted from accessing resources or other apps unless it is authorized by the users. Users have to grant all resource access requests before installing and using an app. Information leakage occurs when users grant resources without any restriction from OS. However, Android's permission control mechanism has been proven ineffective to protect user's privacy and resource from malicious apps. Studies showed that more than 70% of smart phone apps request to collect data irrelevant to the main function of the app [10][11]. With more than 1:4 million available apps in Google Play, and a great number of apps from miscellaneous third-party markets, a significant number of malicious apps have been exposed to Android users for installation.

However, when installing a new app, only a small portion of users pay attention to the resource being requested, since they tend to rush through prompted permission request screens to get to use the application. Only a small portion (3%) of users are cautious and make correct answers to permission granting questions. In addition, the current Android permission warnings do not help most users make correct security decisions [12]. The "blaming the users" approach has become a failure to protect Android users

3.2 Privilege escalation

Privilege escalation or permission escalation attacks were leveraged by exploiting publicly available Android kernel vulnerabilities to gain elevated access to resources that are normally protected from an application or user. This type of attack can result in unauthorized actions from applications with more privileges than intended, which causes many sensitive information leakage. Android exported components can be exploited to gain access to the critical permissions [13]

3.3 Repackaging Apps

Repackaging is one of the most important and common security issues of the Android OS. Repackaging is the process of disassembling/decompiling of .apk files using reverse-engineering techniques and adding (injecting) malicious code into the main source code. Repackaging techniques that can be used on the Android platform allow malicious code to be disguised as a normal app. It is difficult to distinguish between a repackaged malicious code and a normal app because the repackaged app usually appears to function in the same way as the legitimate one.

3.4 Denial of Service (DoS) attack

The increasing number of smartphone users and prevalence of mobile devices (phones, tablets) which are connected to the Internet can be a platform for growth of DoS attacks. Since the majority of smartphones are not equipped with the same protections (i.e. anti-virus programs) as PCs, malicious apps find it as a proper platform for DoS attacks. Overusing limited CPU, memory, network bandwidth and battery power are the main goals of DoS attacks [14].

3.5 Colluding

Colluding threat is a client-side attack. In this attack, users install a set of apps developed by the same developer and same certificate and grant different types of permissions including sensitive and nonsensitive. After installing apps, these apps can take advantage of a shared UID and get access to all their permissions and resources [15].

3.6 Relay Attacks:

It involves only future applications on mobile phones. Elements and application access security relays APDU command interface / response network (GSM, UMTS, and Wi-Fi). Attackers can use victims' secured as if they have their physical possession. Relay application can access additional resources (address book, keyboard, etc.) [16]. In article [17], Peer-to-Peer communications in NFC (Near field communication) are being deliberated for a variety of applications with payment. Relay attacks are a threat and can circumvent security measures and encryption/decryption using temporary contracts.

The author's contributions in this work include the implementation of practical demonstrations of the first relay attack using NFC mobile platform technology. They show that the attacker using NFC can create a proxy for the development and introduction of the software (without hardware change) of the MID let appropriate for mobile devices. The attack does not involve any code validation and software to be installed on the insurance program. It also uses ordinary, readily accessible APIs such *as JSR 257 and JSR 82,* need for action measures. Such attacks can be controlled intensely using location-based solutions discussed in [17].

3.7 Cold Boot Attack:

Smartphones and tablets are easily stolen or lost. In paper [18], it is discussed that, this makes them vulnerable to low-grade memory attacks such as *cold-boot* attack using a bus, monitor to keep an eye on the memory bus and *DMA* attacks. The article further describes the *Sentry*, a system that permits applications and operating system modules to stock their code and data on the *System-on-Chip (SoC)* instead of DRAM. They propose the use a special mechanism of ARM-specific was specially intended for embedded systems, but it is still in existing mobile phones, to defend applications and OS in contradiction to a memory subsystem.

3.8 Brute Force Attack:

Kim [19], proposed a keypad to make the brute force and smudge attacks difficult. This type of keypad increases the time that is required by both brute force and smudge attacks. Keypad time is increased by the formation of random buttons and display delay time.

3.9 Smudge Attack*:*

Gibson [20], explored smudge attacks using oil on the mobile touch screen and captured the smudges. They emphasized on the effect on password pattern of smartphone. They provide a primary study of applying the information learned in a smudge attack to predicting a pattern password.

3.10 Cross-Site Scripting (XSS) Attacks*:*

Jin and Hu run the risk of systematic reviews in HTML5 - based mobile application, discovered a new injection code attack, which inherited a cross-site scripting (XSS) attacks (basic cause), but several channels used to insert XSS code. These channels exclusively for mobile devices, including contacts, SMS, bar codes, and MP3 to assess the occurrence of addition code susceptibility in mobile application based on HTMLThe problem is that HTML5-based malicious code can be inserted into any automated software or application and run. This is the cause of cross-site scripting (XSS) attacks are one of the most common attacks on Web-based applications or programs. Cross-site scripting can only target web application [21].

3.11 USSD Attacks:

USSD (Unstructured Supplementary Service Data) is a protocol used by operators of www(world wide web) to run specific functionality between users and operators [22], examples such as functions including credit check and credit of USSD, USSD can send a prepaid callback, Mobile-Money services. The USSD contains following components: Main Activity, USSD interceptor Service, Boot service and Permission testing. Hamdani, and Elhajj [23] identified and evaluated two types of Android smartphone based attacks. The first is done by sending an SMS in the background and push notifications network to steal customer credit. Also, they show how the SPM security structure in Android has grown, but they showed how the attack can still be performed. The second attack using the mobile dialler application using the USSD protocol on the target user background.

3.12 Camera based Vulnerabilities and Attacks:

Currently, almost all smartphones have features like camera and touchscreen. These functions can lead to attacks on smartphones. Users change device through third party applications from the "app stores" or traditional websites. Source application is a problem, so users are constantly at risk of installing malicious programs that steal personal information or gain root access to their device [24].

4. Conclusion

Smartphones are the multipurpose handheld devices that contain a lot of third-party applications that extend the functionality of the device. With the quick production of smartphones prepared with many features such as several connectivity links and sensors, the mobile malware are growing. The smartphone environment is different from the PC environment. Similarly, the solutions to prevent the infections and diffusion of malicious code in smartphone are different from PC or other computer devices. Smartphones have insufficient resources, including power (battery) and processing unit. Increasing the capabilities of the smartphone, these features can be misused by attackers, as different types of links, sensors, services and user's secrecy. In this work, at first we discussed about the architecture of android system ,
We investigated the vulnerabilities in smartphones and attacks that can occur in smartphones. Secondly, we have characterized identified attacks in contradiction of smartphones, concentrating on why attacks occur and what are their effects on smartphones.

Reference

[1] Gartner, "Gartner: 1.1 billion android smartphones, tablets expected to ship in 2018," Online; accessed at Jan 5, 2018, http://tinyurl.com/n8t3h9y.
[2] Victor, "Android's google play beats app store with over 1 million apps, now officially largest," Online; accessed at May 12, 2017, http://www.phonearena.com/news/Androids-Google-Play-beats-App-Store-withover- 1-million-apps-now-officially-largest id45680.
[3] "Number of available applications in the google play store," 2018, http://www.statista.com/.
[4] W. Rothman, "Smart phone malware: The six worst offenders," Online; accessed at April 17, 2015, http: //www.nbcnews.com/tech/mobile/smart-phone-malware-six-worst-offenders-f125248.
[5] "Bit9 report: Pausing google play: More than 100,000 android apps may pose security risks," 2012, https://www.bit9.com/files/1/Pausing-Google-Play-October2012.pdf.
[6] "Number of android applications," Online; accessed at August 7, 2017, http://www.appbrain.com/stats/number-of-android-apps.
[7] "Pandaapp," Online; accessed at August 7, 2015, http://www.pandaapp.com.