Analysis And Research of System Security Based On Android

Binit Kumar Agarwal, Akshay Jain

CCSIT, TMU, Moradabad

binit.kgpand@gmail.com

***Abstract*— Android has the biggest market share among all Smartphone operating system. Security is one of the main concerns for Smartphone users today. As the power and features of Smartphones increase, so has their vulnerability for attacks by viruses etc. Perhaps android is more secured operating system than any other Smartphone operating system today. Android has very few restrictions for developer, increases the security risk for end users. In this paper we have reviewed android security model, application level security and security issues in the Android based Smartphone.**

***Keywords*— Android security, smartphone security, malware.**

I. Introduction

Android is a mobile operating system (OS) currently developed by Google, based on the Linux Kernal and designed primarily for touch screen mobile devices such as Smartphone's and tablets. Android's user interface is based on direct manipulation, using touch gestures that loosely correspond to real-world actions, such as swiping, tapping and pinching, to manipulate on-screen objects, along with a virtual keyboard for text input. In addition to touch screen devices, Google has further developed Android TV for televisions, Android Auto for cars, and Android Wear for wrist watches, each with a specialized user interface. Variants of Android are also used on notebooks, game consoles, digital cameras, and other electronics. Android's source code is released by Google under open source licenses, although most Android devices ultimately ship with a combination of open source and proprietary software, including proprietary software required for accessing Google services.Android is popular with technology companies that require a ready-made, low-cost and customizable operating system for high-tech devices. Its open nature has encouraged a large community of developers and enthusiasts to use the open-source code as a foundation for community-driven projects, which add new features for advanced users or bring Android to devices originally shipped with other operating systems. The success of Android has made it a target for patent litigation as part of the so-called "smart phone wars" between technology companies.

II. Interface

Android's default user interface is based on direct manipulation, using touch inputs, that loosely correspond to real-world actions, like swiping, tapping, pinching, and reverse pinching to manipulate on-screen objects, and a virtual keyboard. The response to user input is designed to be immediate and provides a fluid touch interface, often using the vibration capabilities of the device to provide haptic feedback to the user. Internal hardware such as accelerometers, gyroscopes and proximity sensors are used by some applications to respond to additional user actions, for example adjusting the screen from portrait to landscape depending on how the device is oriented, or allowing the user to steer a vehicle in a racing game by rotating the device, simulating control of a steering wheel.

Present along the top of the screen is a status bar, showing information about the device and its connectivity. This status bar can be "pulled" down to reveal a notification screen where apps display important information or updates, such as a newly received email or SMS text, in a way that does not immediately interrupt or inconvenience the user. Notifications are persistent until read (by tapping, which opens the relevant app) or dismissed by sliding it off the screen. Beginning on Android 4.1,

"expanded notifications" can display expanded details or additional functionality; for instance, a music player can display playback controls, and a "missed call" notification provides buttons for calling back or sending the caller an SMS message.

Android provides the ability to run applications which change the default launcher and hence the appearance and externally visible behaviour of Android. These appearance changes include a multi-page dock or no dock, and many more changes to fundamental features of the user interface.

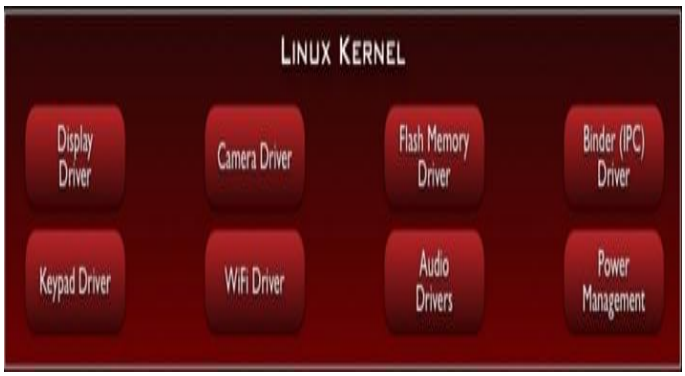
III. Applications

Applications which extend the functionality of devices, are written using the Android software development kit (SDK) and, often, the Java programming language that has complete access to the Android APIs. Java may be combined with C/C++ and the "Go" programming language is also supported since its version 1.4, which can also be used exclusively although with a restricted set of Android APIs. The SDK includes a comprehensive set of development tools, including a debugger, **software** libraries, a handset emulator based on QEMU, documentation, sample code, and tutorials. Initially, Google's supported integrated development environment (IDE) was Eclipse using the Android Development Tools (ADT) plugin; in December 2014, Google released Android Studio, based on IntelliJ IDEA, as its primary IDE for Android application development. Due to the open nature of Android, a number of third-party application marketplaces also exist for Android, either to provide a substitute for devices that are not allowed to ship with Google Play Store, provide applications that cannot be offered on Google Play Store due to policy violations, or for other reasons. Examples of these third-party stores have included the Amazon Appstore, GetJar, and SlideMe. F-Droid, another alternative marketplace, seeks to only provide applications that are distributed under free and open source licenses.

IV. Memory management

Since Android devices are usually battery-powered, Android is designed to manage memory (RAM) to keep power consumption at a minimum, in contrast to desktop operating systems which generally assume they are connected to unlimited mains electricity. When an Android application is no longer in use, the system will automatically suspend it in memory; while the application is still technically "open", suspended applications consume no resources (for example, battery power or processing power) and sit idly in the background until needed again. This brings a dual benefit by increasing the general responsiveness of Android devices, since applications do not need to be closed and reopened from scratch each time, and by ensuring that background applications do not consume power needlessly.

V. Linux Kernel



Android's kernel is based on one of the Linux kernel's long-term support (LTS) branches. Since April 2014, Android devices mainly use versions 3.4 or 3.10 of the Linux kernel. The specific kernel version depends on the actual Android device and its hardware platform; Android has used various kernel versions since the version 2.6.25 that was used in Android 1.0.

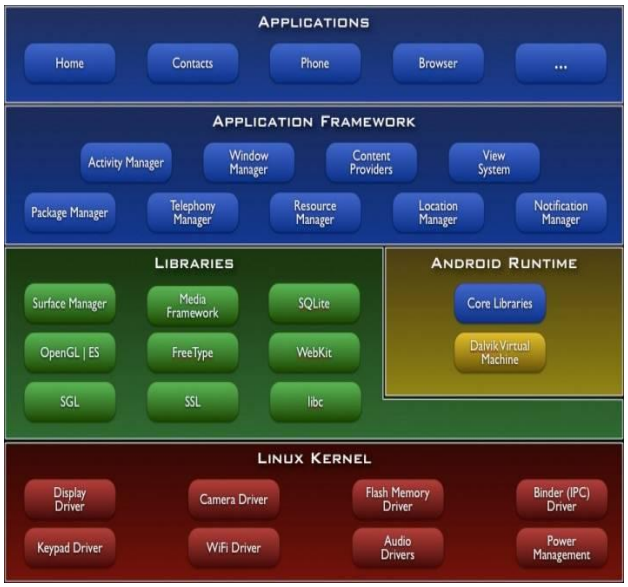
Android's variant of the Linux kernel has further architectural changes that are implemented by Google outside the typical Linux kernel development cycle, such as the inclusion of

components like Binder, Ashmem, Pmem, Logger, Wakelocks, and different out-of-memory (OOM) handling.Certain features that Google contributed back to the Linux kernel, notably a power management feature called "wakelocks", were rejected by mainline kernel developers partly because they felt that Google did not show any intent to maintain its own code.Google announced in April 2010 that they would hire two employees to work with the Linux kernel community, but Greg Kroah-Hartman, the current Linux kernel maintainer for the stable branch, said in December 2010 that he was concerned that Google was no longer trying to get their code changes included in mainstream Linux. Some Google Android developers hinted that "the Android team was getting fed up with the process," because they were a small team and had more urgent work to do on Android.

Android is a Linux distribution according to the Linux Foundation, Google's open-source chief Chris Debonair, and several journalists.Others, such as Google engineer Patrick Brady, say that Android is not Linux in the traditional Unix-like Linux distribution sense; Android does not include the GNU C Library (it uses Bionic as an alternative C library) and some of other components typically found in Linux distributions.

VI. Application Frame Work

On top of the Linux kernel, there are the middleware, libraries and APIs written in C, and application software running on an application framework which includes Java-compatible libraries based on Apache



Until version 5.0, Android used Dalvik as a process virtual machine with trace-based just-in-time (JIT) compilation to run Dalvik "dex-code" (Dalvik Executable), which is usually translated from the Java bytecode. Following the trace-based JIT principle, in addition to interpreting the majority of application code, Dalvik performs the compilation and native execution of select frequently executed code segments ("traces") each time an application is launched. Android 4.4 introduced Android Runtime (ART) as a new runtime environment, which uses ahead-of-time (AOT) compilation to entirely compile the application bytecode into machine code upon the installation of an application. In Android 4.4, ART was an experimental feature and not enabled by default; it became the only runtime option in the next major version of Android, 5.0. Android's standard C library, Bionic, was developed by Google specifically for Android, as a derivation of the BSD's standard C library code.

Aiming for a different licensing model, toward the end of 2012 Google switched the Bluetooth stack in Android from the GPL-licensed BlueZ to the Apache-licensed BlueDroid.

Android does not have a native X Window System by default, nor does it support the full set of standard GNU libraries. This made it difficult to port existing Linux applications or libraries to Android, until version r5 of the Android Native Development Kit brought support for applications written completely in C or C++.

VII. Versions

1. *Android 1.0 and 1.1: Unnamed*
2. Android *1.5: Cupcake*
3. *Android 1.6: Donut.*
4. *Android 2.0 and 2.1: Éclair*
5. *Android 2.2: Frodo.*
6. *Android 2.3, 2.4: Gingerbread*
7. *Android 3.0, 3.1, and 3.2: Honeycomb*
8. *Android 4.0: Ice Cream Sandwich*
9. *Android 4.1: Jelly Bean*
10. *Android 4.4: Kit Kat.*
11. *Android 5.X: Lollipop*
12. *Android 6.0, 6.1: Marshmallow*
13. *Android 7.0: Nougat*

A. Android 4.4: Kiata

Google announced that Android 4.4 would be named Kit Kat on September 3, 2013. Kit Kat's parent company, Nestlé, was fully on board with the naming of operating system and launched an advertising campaign during Kit Kat's release. As part of the campaign, specially marked packages of Kiata with Andy the Green Android on the package each contained a sweepstakes code that could win a new Nexus 7 Android tablet or Google Play store credit

B. Security

Android is a modern mobile platform that was designed to be truly open. Android applications make use of advanced hardware and software, as well as local and served data, exposed through the platform to bring innovation and value to consumers. To protect that value, the platform must offer an application environment that ensures the security of users, data, applications, the device, and the network.

Securing an open platform requires a robust security architecture and rigorous security programs. Android was designed with multi-layered security that provides the flexibility required for an open platform, while providing protection for all users of the platform. For information about reporting security issues and the update process, see Security Updates and Resources.

Android was designed with developers in mind. Security controls were designed to reduce the burden on developers. Security-savvy developers can easily work with and rely on flexible security controls. Developers less familiar with security will be protected by safe defaults

Android seeks to be the most secure and usable operating system for mobiles by re-purposing classical operating system security controls to protect user data, system resources and provide application isolation. Android provides following security features to achieve these objectives are first robust security at the operating system level through the Linux kernel, second compulsory application sandbox for all applications, third secure interposes communication, fourth application signing, and sixth application defined permission and user have to grant permissions.

C. Security Issues faced by Android

Android is not secure as it appear, even when such robust security measures. There are several security problems faced by the android, some of them are mentioned below.

i. Android has no security scan over the apps being uploaded on its market.

ii. There are some apps which can exploit the services of another app without permission request.
iii. Android permission security model provides power to user to make a decision whether an app should be trusted or not. This human power
Introduces a lot of risk in Android system.
iv. The Open Source is available to legitimate developers as well as hackers too. Thus
The Android framework cannot be trusted when it comes to develop critical systems.
v. The Android operating system
Developers clearly state that they are not responsible for the security of external storage.
vi. Any app on the android platform
Will access device data just like the GSM and SIM marketer Ids while not the permission of the user?
Android platform provides all security features, but there will always be a risk if the user will install suspicious apps or allow
permission to an app without paying attention.
Android is a victim of its own success, not just in the way it has attracted malicious attention, but in its very nature. One of the reasons the OS has succeeded in gaining market share so rapidly is that it is open source; it is essentially free for manufacturers to implement. Additionally this has led to substantial fragmentation of Android versions between devices and means that vendors have been reluctant to roll-out updates, presumably out of some concern regarding driving demand for future devices.

VIII. Service

A Service is code that is long-lived and runs without a UI. A good example of this is a media player playing songs from a play list. In a media player application, there would probably be one or more activities that allow the user to choose songs and start playing them. However, the music playback itself should not be handled by an activity because the user will expect the music to keep playing even after navigating to a new screen. When connected to a service, you can communicate with it through an interface exposed by the service. For the music service, this might allow you to pause, rewind, etc.

Features:

A. Storage: SQLite, a lightweight relational database, is used for data storage purposes.

B. Connectivity: Android supports connectivity technologies including GSM EDGE, IDEN, CDMA, EVDO, UMTS, Bluetooth, WI-Fi, LTE, NFC and WI MAX.

C. Messaging: SMS and MMS are available forms of messaging, including threaded text messaging and Android Cloud to Device Messaging (C2DM)
and now enhanced version of C2DM, Android Google Cloud Messaging (GCM) is also a part of Android Push Messaging service.

D. Multiple language support: Android supports multiple languages.

E. Web browser: The web browser available in Android is based on the open-source Web Kit layout engine, coupled with Chrome's V8 JavaScript engine. The browser scores 100/100 on the Acid3 test on Android 4.0.

F. Java support While most Android applications are written in Java, there is no Java Virtual Machine in the platform and Java byte code is not executed. Java classes are compiled into Dalvikexecutables and run on Dalvik, a specialized virtual machine designed specifically for Android and optimized for battery-powered mobile devices with limited memory and CPU. J2ME support can be provided via third party applications.

G. Multi-touch: Android has native support for multi-touch which was initially made available in handsets such as the HTC Hero. The feature was originally disabled at the kernel level (possibly to avoid infringing Apple's patents on touch-screen technology at the time). Google has since released an update for the Nexus One and the Motorola Droid which enables multi-touch natively.

8) Bluetooth: Supports A2DP, AVRCP, sending files (OPP), accessing the phone book (PBAP), voice dialing and sending contacts between phones. Keyboard, mouse and joystick (HID) support is available in Android 3.1+, and in earlier versions through manufacturer customizations and third-party applications.

9) Tethering: Android supports tethering, which allows a phone to be used as wireless/wired Wi-Fi hotspot. Before Android 2.2 this was supported by third-party applications or manufacturer customizations.

10) Screen capture: Android supports capturing a screenshot by pressing the power and volume-down buttons at the same time. Prior to Android 4.0, the only methods of capturing a screenshot were through manufacturer and third-party customizations or otherwise by using a PC connection (DDMS developer's tool). These alternative methods are still available with the latest Android.

Specific security features vary between devices and operating systems. Use whichever features your device offers that provide the best security for your needs:

Password, passcode, or PIN: Setting a password, passcode, or PIN to access your device is generally simple and effective. Use a code that is four digits or longer, and keep it secret, like you do for your email password or passphrase. See About your IU passphrase or At IU, how can I choose a secure PIN?

Unlock pattern: Some handheld devices let you set unlock patterns that function like PINs. Use a pattern with some complexity (e.g., with at least five points), keep it secret, and protect it from observers. Additionally, be aware that smudges on the face of your device may reveal your pattern to unauthorized users.

Device lockout: Most handheld devices provide a lockout option that locks the device if someone makes several consecutive unsuccessful attempts to enter the password, PIN, or pattern. Using the lockout option can thwart a brute-force attempt to guess your password, PIN, or pattern. Setting the lockout limit to 10 attempts is usually sufficient.

Auto-wipe: Auto-wipe is similar to the lockout option, but more secure. After several consecutive unsuccessful password, pattern, or PIN attempts, the device will automatically erase (i.e., wipe) all stored data and reset itself to the factory defaults.

Note: When you use the auto-wipe option, make sure to back up your data regularly (e.g., to a desktop computer or a cloud storage service). Consult your device's documentation for instructions on backing up data.

Encryption: Certain handheld devices are capable of employing data encryption. Consult your device's documentation or online support resources for information about available encryption option.

The following common features are frequently useful, but can also create security risks. You may want to consider disabling them:

Bluetooth: Consider disabling Bluetooth connectivity on your device unless you need it. Hackers and data thieves can use Bluetooth connections to "eavesdrop" on your device and access your sensitive data.

GPS: Consider disabling Global Positioning System (GPS) and other location services unless you need them. Your physical location (or the locations of your device) is a piece of sensitive data that you may not want stored or broadcast. Conversely, if your device is GPS-enabled, some apps and services (e.g., Find My iPhone) can help locate your device if it is lost or stolen.

IX. 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.

X. References

[1] Android Open Source Project. Android Security Overview.http://source.android.com/devices/tech/security/index.html.
(2013)

[2] Kaur S. and Kaur M., Review Paper on Implementing Security on Android Application, *Journal of Environmental Sciences, Computer Science and Engineering & Technology*, **2(3)**, **(2013)**

[3] Android Open Source Project. Security and permissions.http://developer.android.com/guide/topics/security/permissions.html. **(2013)**

[4] Android Open Source Project. Publishing on GooglePlay.http://developer.android.com/distribute/googleplay/publish/preparing.html. **(2013)**

[5] Enck W., Octeau D., McDaniel P. and Chaudhuri S., A Study of Android Application Security, *The 20th USENIX conference on Security*, 21-21, **(2011)**

[6] Powar S., Meshram B. B., Survey on Android Security Framework, *International Journal of Engineering Research and Applications*, **3(2)**, (**2013**)

[7] Smalley S. and Craig R., Security Enhanced (SE) Android:Bringing Flexible MAC to Android,www.internetsociety.org/sites/default/files/02_4.pdf . **(2012)**

[8] Enck W., Gilbert P., Chun B.G., Cox L.P., Jung J.,McDaniel P. and Sheth A.N., TaintDroid: An Information- Flow Tracking System for Realtime Privacy Monitoring on Smartphones, *9th USENIX Symposium on Operating Systems Design and Implementation*. **(2010)**

[9] Berger B.J., Bunke M., and Sohr K., An Android Security Case Study with Bauhaus, *Working Conference on Reverse Engineering*, 179–183 **(2011)**

[10] Ongtang M., McLaughlin S., Enck W. And McDaniel P., Semantically Rich Application-Centric Security in Android, *Computer Security Applications Conference*, 340–349 **(2009)**