Research on Android Access Control based on Isolation Mechanism*

Ying Peng
School of Computer Science and Technology
Soochow University
Suzhou, China
E-mail: KapruaPeng163@163.com

Mingxin Zhang†
School of Computer Science and Engineering
Changshu Institute of Technology
Changshu, China
E-mail:†mxzhang 163@163.com

Jinlong Zheng
School of Computer Science and Engineering
Changshu Institute of Technology
Changshu, China
E-mail: zjinlong@163.com

Zhenjiang Qian
School of Computer Science and Engineering
Changshu Institute of Technology
Changshu, China
E-mail:tony_h@sina.com

***Abstract*—With the rapid development of the Internet, the application of Android system is more and more widely. But then the user's privacy leaks, malicious software attacks and hacking and other security issues have become increasingly serious. Android privilege mechanism is an important security protection mechanism of Android. But the current authority mechanism cannot be a good solution to the problem of improper access and the lack of protection of the core thus brings potential safety hazard. A three-tier Android safety protection safety guarantee system is designed to protect the security control subsystem, which is composed of the application layer, the virtual monitoring layer and the trusted root layer. It then uses the security control subsystem to control the other main parts of the Android system. Experiments show that the system can effectively block applications with sensitive permissions, and put it into the isolation zone, finally integrated hardware mechanism to ensure the security of Android system.**

Keywords—***android permissions; APEX; isolation mechanism; system security***

I. INTRODUCTION

In recent years, with the continuous improvement of the software performance and the popularity of mobile Internet, smart phones and tablet PCs and other smart mobile devices are widely used. Among which Android platform has gained great market favor on account of its characteristics such as open source and easy operations. STATISTA—the world's leading information technology, communications industry and consumer technology market research firm, issued a survey report, representing that market share of Android has reached 81.6% in the fourth quarter of 2015[1-8],compared with approximately 50% in 2011.

However, the open source of Android system also pushes itself to be the target of malicious software. Subsequent security issues arise, such as data loss, malicious software attacks and hacker intrusion, etc. put forward a huge test on the Android system. At the same time, because of the rapid rise of remote Trojans, situations of spreading malicious software, malicious plug-ins through the back door in the Android platform also begin to increase rapidly. At present, the security threats against the Android system mainly include: system vulnerabilities, Rom kernel level malicious programs, and the application layer of malicious programs. It is worth mentioning that the threat of security is particularly serious brought by the application layer of malicious programs.

How to ensure the security and reliability of mobile terminals, How to implement mobile trusted platform on smart terminals under the standard of TCG (Trusted Computer Organization)[9],and how to ensure the Android applications, especially those involving sensitive confidential data far away from malicious programs have become issues, urgent to be resolved for security of mobile intelligent terminal. Therefore, the proposal and implementation of a trusted secure mobile terminal is becoming the focus of the current study without exception.

Privilege management is one of the important measures for the protection mechanism of Android system. In most cases, application programs apply to obtain the right to use the system resources, while improper permissions granted to the right would leave a security risk.

However, there are some defects in the Android access mechanism. In a general way, the security flaws of Android privilege mechanism are mainly manifested in the following aspects:

(1) Coarse-grained authorization mechanism

Android permission mechanism is coarse-grained. It proposes an all-or-nothing decision at install-time, which means for users, they have only two choices: either to accept all permissions, or to refuse all authorizations. Especially users have no choice but to allow access to all the requested permissions if they wish to use the applications, even though some permission is inappropriate. Then a slew of potential security issues will come afterwards.

(2) Coarse grained permissions

Not only the Android permissions management mechanism but also most of the Android permissions are coarse-grained, thus the application of these permissions can be arbitrary access to specific resources. For example, more than 60% of the Android application requires INTERNET permission [10]. Applications with the permission can send HTTP (S) request to all fields, and connect to any destination address and port. Therefore, malicious applications can be

Supported by the National Natural Science Foundation of China under Grant No.61173130,the Natural Science Foundation of China under grant No. 61402057,and the Natural Science Foundation of Jiangsu Province under grant No. BK20140418.Corresponding author: Mingxin Zhang †mxzhang163@163.com

978-1-5090-5437-4/16 $31.00 © 2016 IEEE
DOI 10.1109/WISA.2016.53

disguised as legal procedures needed to be accessed by Internet; abuse of Internet resources is resulted.

Many scholars have made improvements to authority problems.

The team of Felt [11] completed the construction of the mapping set of the API(Application Programming Interface) rights corresponding to the relationship, and on this basis compare permissions of applying for application and API invoking in order to check the overflow problem.

The team of Nauman[12] proposed APEX (Android Permission Extension) to realize dynamic user grant permissions. However, in some cases it ignores the actual functional requirements, which may result in an unavailable application. In addition, some users do not understand the rules of permissions, they cannot make the appropriate choice, which may bring authorization difficulties, even take the safety concerns.

In the aspect of the system kernel layer, the U.S. National Security Agency NSA introduces the SELinux [13, 14] security framework. It is mainly composed of two parts, including the strategy and the implementation. In short, the policy package works in the security server, and the implementation is taken by the object manager. However, merging SELinux technology into Android needs to face the challenges from the kernel, user space and policy configuration.

L4Linux based on L4Android [15] and Cells [15] proposed use of multiple operating systems on a mobile phone to achieve better isolation and security. Among them, the goal of L4Android is to run Android in a virtual machine on top of the micro kernel. ARM under the TrustZone architecture can implement an additional processor mode for building an isolation and security environment.

Some blemishes still exist in the following aspects:

(1) The authority control depends on users to complete the fine-grained authorization, but fail to take into account that ordinary users cannot understand the specific functions and potential risks of permissions, so references are not explicit enough for users to make a decision at install-time.

(2) The improvement of the current Android security protection system is relatively broad, and there are no requirements of the fine granularity restrictions according to actual demands of the requirements. Partial applications involve sensitive permissions that are not related to the main service provided by it.

The kernel is not really isolated in the present protection mechanism for the security of the Android system, so that attacks on the kernel will threaten the whole system.

Aiming at the above problems, this paper intends to build a three-tier security system. The three-tier security system is composed of Android Application layer, the VMM (virtual machine monitor) layer and the Trusted Root layer composition. In the operation space, the security domain and the common domain are partitioned, and the application program with sensitive permission is strictly separated from the common program to protect the system resources and the security of the user's personal information. And on the basis of the Apex mechanism, the permissions of the Android are further restricted, allowing the user to set permissions to enhance the self-adaptability.

II. Android Security Mechanism

A. Android Permission Mechanism

Android-specific security mechanisms include permission mechanism, component packaging, signature mechanism and Dalvik virtual machine. Permission management is one of the most important safety measures in the Android system, which is mainly aimed at the application by restricting access to system resources. Permission corresponds to an interface using a system resource or function.

Android application framework layer implements forced access control mechanism for access between components. The system defines a set of permission tags related to security operations, such as <permission>, <permission-group>, <permission-tree>, etc. Whenever an application needs to use certain permission, it is necessary to use these tags in the configuration file “Manifest.xml” to declare the permission to access resources. Otherwise, due to the lack of relevant authority, and the sandbox protection, the application will not be able to provide normal services and functions. Then the installation system “Package Installer” will display the permission of the application to the user. After the user agrees to authorize the application, it can be successfully installed. It is noteworthy that, once successfully authorized, all components of the application will inherit all the rights that the application declared. At the same time, the components can also limit the scope of the other components that can interact with them by using the tags.

Android supports four kinds of permission protection levels, normal, dangerous, signature and signature or system.

(1) Normal permissions: lower risk. They can be used without user’s specific authorization, such as the power of vibration function - "VIBRATE" permission;

(2) Dangerous permissions: higher risk. It needs to get the user's permission to apply for them during the installation time. For example, call permission - "PHONE_CALLS";

(3) Signature permissions: Only when the application which applies for them and the program which declares them use the same signature, they can be authorized;

(4) Signature Or Systems: :Only when the application which applies for them is located in the same Android system mirror or the application which applies for them and the program which declares them use the same signature, they can be authorized.

B. Android Isolation Mechanism

Android is based on the Linux kernel, so each application runs in its own Linux process. Typically, each application is assigned a unique Linux user ID, so the Linux's own access control mechanism guarantees that the application's files are invisible to other applications. In particular, the Android application runs in the Dalvik virtual machine, so its code is isolated from the other applications.

C. Android Permissions Authorization Extension

Permissions authorization extension is mainly aimed at

the coarse-grained authorization of all reject or all grant in Android permission control, improving the package manager in order that the user can use the part functions of the application in the case that only part permissions were granted. A typical representative of this aspect is APEX.APEX allows the user to grant or revoke permissions dynamically according to the specific time, place, the use of platform, SMS send and so on, thus enhancing user's personal decisions over these permissions and further improving the security and credibility of the system.

III. THREE-TIER SECURITY SYSTEM DESIGN

A. Three-tier Security System Architecture

Based on the Android hardware isolation mechanism and combined with the existing Android permission mechanism, we want to design a three-tier security system to protect the safety control subsystem, and then use it to control the behavior of every subject in the Android system to prevent the application breaking restrictions of the security policy. The three-tier security system is composed of Application layer, Virtual Machine Monitor layer and Trusted Root layer. It is as shown in Figure 1 below.

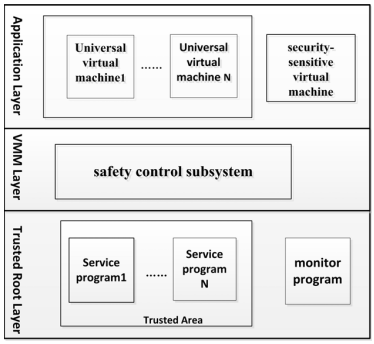

Fig. 1. Three-tier security system.

The specific functions of each layer are as follows:

(1) Application Layer

The main function is to isolate the running space of the common program and the sensitive program. It consists of several common virtual machines and security sensitive virtual machines which are used to run the normal application and the sensitive application, respectively.

(2) Virtual Machine Monitor Layer

The main function is as a link between the upper and lower layers. Judging by the security control subsystem permission security sensitivity of the application, sensitive program enters the next layer, non-sensitive program returns to the upper generic virtual machine lever. It has strict spatial separation control to prevent the mutual interference between virtual machines and provide safe and reliable virtual machine sandbox.

(3) Trusted Root Layer

The main function is to provide a secure and trusted area to protect VMM and a number of Android virtual machine areas and to the greatest extent ensure the safety of the system.

The above three layer security system of new intelligent mobile platform, using hardware security mechanism, on the one hand can solve the security issues of the intelligent mobile terminal; on the other hand also can promote security method research of mobile platform.

B. Application permissions design

First of all, analyze and judge the existing common permissions of the Android platform. Extract original permission set from the Android.xml file, in the installation phase of the application to intercept, decompile the APK (application installation package), form a jar source file, scan DEX, contrast API permission control table, remove unused permissions from the original permission set, and then form a practical fence permission set, namely minimum permission collection.

Categorize applications, such as shopping, games, communications and so on and then sum up the corresponding risk permission list which contains a combination of sensitive permissions.

According to the minimum permission collection and the list of dangerous rights, exclude non-essential sensitive permissions, sum up the security installation permission set, and then form the proposed installation push information table. Feedback to the user at the time of installation, allowing users to set permissions allowed.

During the installation process, it integrates Android permission extension, modifies the program PackageInstaller(), authorizes the user program partly in the installation time to implement access control of the running application according to the strategy.

After the application is installed successfully, the service "PackageManagerService" will check the permissions being used during the running time to detect whether these permissions are authorized when the application is installed. If has been authorized, these permissions will be agreed to be used, otherwise it will be refused.

Each user customizes the installation according to a brief introduction of permission and the proposed installation push information table. Add the "setting" button in the installation interface and limit the number of times, hours, days of the use of the inevitable sensitive privilege functions.

C. VMM Construction

TrustZone allows system virtualization based on hardware which provides common and credible/security regions. The multimedia operating system users often use runs in the common area and the safety critical software runs in the safety zone. Management software in the safety zone can access the memory of common area, and vice versa. Recently ARM announced to provide virtualization support for the Cortex A15 and release hypervisor Xen-ARM based on ARM platform architecture. This paper realizes hardware virtualization construction based on Xen-ARM. It is as shown in Figure 2 below.

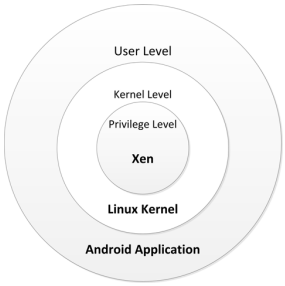

Fig. 2. Virtual structure design.

The kernel level and the user level are defined as non-sensitive areas.

Set access control module in VMM, and when the application falls into VMM because of access to sensitive resources and information, analyses it to determine whether it can be implemented according to the strategy.

Through the isolation mechanism based on Xen, a number of Android virtual machine instances of two types (security-sensitive virtual machine and universal virtual machine) are created and run in each virtual machine. Security virtual machine will implement mandatory access control and behavior testing by Xen, and only security applications can run in a security virtual machine.

IV. IMPIEMENTATION

After the text edit has been completed, the paper is ready for the template. Duplicate the template file by using the Save As command, and use the naming convention prescribed by your conference for the name of your paper. In this newly created file, highlight all of the contents and import your prepared text file. You are now ready to style your paper; use the scroll down window on the left of the MS Word Formatting toolbar.

When the application is installed, the security control subsystem plays an important role of judge, focus on the sensitive permissions. In contrast, the non-sensitive application program directly falls into the universal virtual machine, running autonomously and independently without interference. The applications with sensitive permissions will be handed over to the trusted root layer. Further assessment is executed by the monitoring program. Then the applications with sensitive permissions will be installed in security regional of the trusted root layer with authority mechanism authorized by the user, and returns to the security sensitive virtual machine to run on independently.

The Provider Content of the Android System allows applications to access personal information, such as contacts and diaries, etc. While some device of the Android that providing input data allows the application to interact with the surrounding environment. For example, the GPS, the user's location information is vulnerable to leakage. Some API (Application Programming Interface) calls used by the application may cause the user to incur charges, such as text messaging, telephone, network, etc.

As shown in Table 1 below, it lists some of the common ones.

TABLE I. COMMON PRIVACY SENSITIVE PERMISSIONS

Use of permissions	Name of permissions
get contacts	android.permission.READ_CONTACTS
read text messages	android.permission.READ_SMS
access to account information	android.permission.ACCOUNT_MANAGER
read system logs	android.permission.READ_LOGS
get accurate location information	android.permission.ACCESS_FINE_LOCATION
network access	android.permission.INTERNET
send text messages	android.permission.WRITE_SMS

In the virtual monitoring layer, the VMM mainly intercept applications with these commonly permissions. According to the principle of static analysis, static analysis is carried out with the aid of the static information flow tool "FlowDroid", and the sensitive rights of privacy disclosure are captured.

In the personal installation stage, with the help of APEX expansion mechanism, the user can complete the definition of personal installation. For example, as is shown in Figure3, we make some restrictions to the access time in the file "apex.xml".

```
<policy App ="com.Phone"
Permission="PHONE_CALLS" times="3"begin="8:00"
end=" 23:00" days="2"/>
```

Fig. 3. Restrictions in permission.

It represents that for the application "com.Phone", it can only have the access of permission "PHONE_CALLS" from 8 am to 11pm within 3times.And it will work for 2 days.

In this paper, We use the Android 4.4 version to take a test on the Ubuntu 16.04 LTS .We alter the installation framework "PackageInstaller" in the definition section, add a set button to limit the sensitivity of the permissions.

We downloaded 50 kinds of common applications from the Android Market, refer to the game applications, the video players, the weather forecast applications, the navigation applications, financial products and so on, and focus on sensitive permissions to conduct an intercept test. We enumerate some of the test results, as is shown in Table 2 below.

TABLE II. COMMON PRIVACY SENSITIVE PERMISSION TEST

applications	Numbers of intercepted permissions	Universal or security-sensitive virtual machine
Taobao.app	5	security-sensitive
Baidumap.app	1	universal
Kuwomusic.app	4	security-sensitive
Iqiyi.app	2	universal
Meitu.app	1	universal
Mojiweather.app	1	universal
Iireader.app	2	universal

We default applications containing more than 3 kinds of

sensitive permissions in range of the non-security applications, and push them into a security sensitive virtual machine.

It represents that from the analysis of the results, even the general virtual machine applications still contains sensitive permissions. The experimental results show that the design scheme of this paper can play a role in the safety of the Android system to a certain extent.

But for some tacit permission, because it does not show in the application files, there may be undetected.

V. CONCLUSIONS

In this paper, a three-tier security system is constructed by combining the hardware isolation mechanism and the APEX access control expansion, which helps the user to determine the granted permissions independently. This scheme takes both the Android system kernel layer and application layer security into account. It provides a considerable direction of exploration procedure for Android security protection. Android security is not an isolated problem, when considering the Android safety improvement and reinforcement of the program; we must take a comprehensive consideration of the performance, efficiency and ease of use. Just from a unilateral consideration the authority is not enough to intercept all security threats, how to run in the application at the same time, to better monitor the application, to improve the efficiency of the system is the direction of improvement in the future.

ACKNOWLEDGMENT

This paper is supported by the National Natural Science Foundation of China under grant No.61173130, and the Natural Science Foundation of China under grant No. 6140 2057, and the Natural Science Foundation of Jiangsu Province under grant No. BK20140418.

REFERENCES

[1] Zhao H, Chen M, Qiu M. A novel pre-cache schema for high performance Android system [J]. Future Generation Computer Systems, 2015, 70(56): 766-772.

[2] Sanchez M I, Oliva A D L, Bernardos C J. Experimental analysis of connectivity management in mobile operating systems[J]. Computer Networks, 2015, 80(2):345-349.

[3] Chen W, Xu L, Li G. A Lightweight Virtualization Solution for Android Devices [J]. IEEE Transactions on Computers, 2015(1):2741-2751.

[4] Jung J H, Ju Y K, Lee H C. Repackaging Attack on Android Banking Applications and Its Countermeasures[J].Wireless Personal Communications, 2013, 73(4):1421-1437.

[5] Wang W, Wang X, Feng D. Exploring Permission-Induced Risk in Android Applications for Malicious Application Detection [J]. IEEE Transactions on Information Forensics & Security, 2014, 9(11):1869-1882.

[6] Jing-Hua L I, De-Jun M U, Yang M K, et al. Design on Android Malware Behavior Analysis System [J]. Journal of Beijing University of Posts and Telecommunications, 2014, 37(s1):104-107.

[7] Zhang Y, Wang K, Yang H, et al. Survey of Android OS Security [J]. Journal of Computer Research and Development, 2014, 51(7):1385-1396.

[8] Jiang S. A Summary on Android Security [J]. Computer Applications and Software, 2012, 29(10):205-210.

[9] Aung Z, Zaw W. Permission-Based Android Malware Detection [J]. International Journal of Scientific and Technology Research, 2013, 2(3):228-234.

[10] Shabtai A, Fledel Y, Kanonov U. Google Android: A Comprehensive Security Assessment [J]. IEEE Security & Privacy, 2010, 8(2):35-44.

[11] Felt A P, Chin E, Hanna S, et al. Android permissions demystified[C]// ACM Conference on Computer and Communications Security. ACM, 2015:627-638.

[12] Nauman M, Khan S, Zhang X. Apex: Extending Android Permission Model and Enforcement with User-defined Runtime Constraints[C]// 5th International Symposium on ACM Symposium on Information, Computer and Communications Security. ACM, 2010:328-332.

[13] Enck W, Gilbert P, Han S. TaintDroid: An Information-Flow Tracking System for Realtime Privacy Monitoring on Smartphones [J]. Acm Transactions on Computer Systems, 2010, 57(3):393-407.

[14] Ahn G J, Xu W, Zhang X. Systematic Policy Analysis for High-Assurance Services in SELinux[C]// Policies for Distributed Systems and Networks, 2008. IEEE, 2008:3-10.

[15] Lange M, Liebergeld S, Lackorzynski A. L4Android: a generic operating system framework for secure smartphones[C]// Proceedings of the 1st ACM workshop on Security and privacy in smartphones and mobile devices. ACM, 2011:39-50.