The Design and Development of the Vulnerability Detection System of Android Platform

Huimin Dong¹, Wenxin Sun²

¹ Hebi Automotive Engineering Professional College, Henan Hebi 458030
² Hebi Polytechnic, Henan Hebi 458030

E-mail: sunwxin126@126.com

Abstract

With the share of Android market increasing continuously, the Android mobile terminal provides convenience for the users. At the same time, it is encountering more and more secure challenges and threats, which has a serious threat to the safe use of users. Although Android has already a good security mechanism to ensure the safe use of the users, in the interest of huge commercial interests, a large number of attackers attack Android users in a variety of ways because of vulnerabilities of system and software. Even though the system module and access control system has been designed by Android, the attacker can still fight with the vulnerabilities of system or third party bugs, which has good for making illegal use some of its application beyond the permission function. It not only has a greater security threat, but also has certain concealment. The present study analyzes and innovates previous researches, put forward a new communication hole detection method based on controlling flow detection and Android permission sensitive dictionary matching, and on this basis, completed the high degree of automation, high detection efficiency design and development of test software.

Keywords: Unauthorized attack, Automatic detection, Android platform

1. INTRODUCTION

The users’ information was leaked at the end of 2014 by 12306 railway customer service center, resulting in a large number of user's ID number, bank card number, bank card password, mobile phone number and other important sensitive information were leaked; after the Spring Festival in 2015, due to faking beauty group red Trojan events and great temptation of the red, after the installation of the counterfeit Trojan application, the users’ personal ID number, bank card number, bank card password, mobile phone number and other important information were uploaded to the specified server, and these information were stole by criminals; Ganji.com, 58 city and other Android client vulnerabilities caused the leakage of important information of a large number of users(Xiao et al., 2015). These events of major information security around us gave us a wake-up call of terminal information security. Android which was with the largest number of users in the world also faces serious security threats. Whether as Android Hair or Android users themselves, they are eager to find and resolve the security issues Android system earlier.

2. THE RISK PROFILE OF ANDROID SYSTEM IN 2015

2.1 Virus infection situation investigation
From 2015, Android security feedback survey statistics of the security software obtained the virus infection of Android system showed a rising trend, increasing to 8966602 in the fourth quarter from 3732034 of the first quarter. What is more, virus infection of the equipment also presented obvious rising trend, growing to 7370278 in the fourth quarter from 2377712 of the first quarter, in which average every 7.6 Android device would have 1 sets of equipment had been infected with the virus. Figure 1 shows the trend of the number of virus and devices infection of Android in the four quarter of 2014 (Hu et al., 2016).

![Figure 1. 2014 Android devices infected trends](image)

In samples of the virus, the malicious virus samples is still very high in proportion, which was up to 52%, more 23 per cent than last year. In the virus infection of users, rogue behavior of the virus infection users accounted for the highest proportion, amounting to 46%; as for the virus infection users in privacy theft, the proportion increased significantly, which is up to 24%, than last year rose about 85%; the proportion of the virus infection users of SMS hijacking decreased significantly, accounted for only 3%, which dropped 81% compared the last year (Hung et al, 2015). Figure 3 shows the investigation of virus infection.

![Figure 2. Infection situation](image)

The reason why there are more and more equipment being infected with a variety of virus is since Android itself exposed many vulnerabilities of system or software, which are used by many criminals (Part et al., 2015). So the primary task is to find out these vulnerabilities, circumvent these vulnerabilities primly and does not allow an attacker to exploit in solving the Android system security problem.

### 2.2 The survey of various sectors in applications types
In 2014, the feedback survey statistics of Android users showed that there was about 86% of the mobile application existing vulnerabilities, which accounted for 33% of high risk vulnerabilities and vulnerabilities in popular applications was more, in which the ranking top5 application had an average of 25 many loopholes. Figure 3 shows the top5 vulnerability in various industries of applications.

Figure 3. The top5 vulnerability of sectorial application

2.3 The investigation of vulnerability investigation in popular mobile applications

Store application data was in the highest flight in mobile applied vulnerabilities, which accounted for 48% of the proportion. Most famous application contained vulnerabilities, if these vulnerabilities were exploited by hackers, it would affect hundreds of millions of users. Figure 4 shows the top10 vulnerability of mobile applications in 11 industries.

Figure 4. The top 10 vulnerability case of mobile application

As for the largest proportion of the Android data storage based on the mobile security vulnerability research in the application of the top5 vulnerabilities in 2014, the present study gives the brief description and analysis, and puts forward the corresponding method to avoid risks(Oh et al., 2014). The Android data storage involves Shared Preferences security risk, Database configuration mode security risk, Content provider file directory traversal vulnerability and Internal Storage security risk.

3. THE CONCEPT OF UNAUTHORIZED ATTACK AND THE MODEL OF THREAT

3.1 The concept of unauthorized attack
Unauthorized attack is a kind of greater harm attack in all the ways of attacking the Android system. In the traditional sense, unauthorized attack refers to lower authorized and restricted users or program which control computer or implement other beyond permission through illegal means to enhance their permission (Lee et al., 2016). Once the attacker strikes successfully, original security settings of system will not be able to protect user privacy and security, posing a serious threat to the users.

Unauthorized attack in Android system has specific characteristics which are without in general authorized attack. Unauthorized attack in Android system refers to the original lower privilege or have no the permission which is through other malicious programs with access to third party applications and makes the vulnerability of third party applications function. It is the first time proposed by DAVI based on the aggressive behavior of the Android system, and in the literature, they define Unauthorized attack in Android system as follows: the application with low (less) access has a higher (more) application components permission (activities, services, content provider and broadcast receiver etc.) attacks without any limits. This attack is very subtle, and often the attacker does not destroy the users’ equipment, just to steal users’ information, so it is difficult for users to detect (Wen et al., 2014). In addition, the premise of this kind of attack is users’ system has a higher priority, but also there is vulnerability of third-party software, with certain limitations, therefore has not been given much attention. As the users of smart phone increase and the people more and more depend on mobile devices, more and more mobile phone preserved more and more users of personal information, such as bank account information, personal identity information, etc., which greatly increase the Unauthorized attack in Android.

In 2009, ENCK et al introduced Android security mechanism at first, announcing a Kirin application installation tools, and for the first time taking static testing strategy (Lee et al., 2012). That is to say, all the permissions were reviewed in time when install the application. If contrary to the allocation strategy, it would refuse to install. Although the tool could effectively prevent sensitive privileges to install applications, the false positive rate was still higher. The reason was that only according to the known risk software permission combination was not able to judge other applications of the same permissions for dangerous software.

After this, FELT et al developed Stowaway. Its principle was through decompile Android application code to detect the API system, and compared with the permission in the file manifest.xml does use to check whether the application as a dangerous procedure (Qing, 2013). The tool of false positives rate was also higher, because developers the required permissions of API system was lack of understanding, often applied beyond the necessary permissions, so in the configuration file for high permission cannot fully prove that is the malicious programs. Subsequently, researches on unauthorized attack were extended to multiple application of collusion attacks, but not were limited to study for a single application (Fang et al., 2014).

### 3.2 Processing procedure of risk model

(1) Android application and the interface of system service

It usually needs to use API function for Application to send request to the system service, such as if an application wanted to set the WiFi, it must call the two functions of “Context. Get System Service” and “Wifi Manager. Set Wifi Enabled” through two steps to complete. However, some system service functions did not provide the API function to applications (Lee et al., 2016). We found that the best method of sending request was the API which was named for transact. It only needed one step to complete the call and could call any public system service function.
Essentially, a request was a collection of several input parameters, in order to send a request, the application first called “transact”, which needed to take four parameters. The first parameter was an integer and the specified by system service function parameters called; the second was a “Parcel” object, it filled the service interface parameters and the required function of corresponding string system; the third parameter was a Parcel object, if there was returned data, it would be stored in the object; the last parameter was an integer that specified whether the system service function corresponding to the need of returned data. When this function was executed, the application process would be driven by “binder” to send these input parameters to the system service layer. The system service layer received parameters through on Transact. The “on Transact” function could handle all kinds of requests, it could do two things: firstly, it could analyze the required parameters from the second parameters of the object, and then could call the corresponding system service function according to the first parameter.

With WiFi service system as an example. Figure 5 shows the interface between application program and system services. Request sent by application was the API of “transact” sends to the function of on Transact through the drive of “binder”. The dotted line represents the parameters of the two same functions. After the corresponding parameters are parsed by “on Transact”, it calls the system service function “Wifi Service Impl. Set Frequency Band” and sets corresponding “integer-band” and “Boolean parameters-persist”.

![Diagram of interface between applications and system services](image)

**Figure 5.** Interface between applications and system services

We assumed that input validation type of attack is initiated by a malicious application, and malicious application had been installed in the target device. To achieve this attack, malicious applications might require some jurisdictions, such as attack WiFi system service might have access to change WiFi state permission. But for other types of attacks (such as a system crash, freeze screen), malicious applications only needed to send a special request to service system, did not need any permission. It is worth to note that malicious requests and legitimate requests looked very similar, which is why this attack was difficult to be detected.
(2) The classification of unauthorized attack

When DAVI et al proposed against the unauthorized attack in Android system in 2011 firstly, they gave the model as shown in Figure 6.

Unauthorized attack mainly included the operational type of attack (such as phone, texting, etc.) as well as the data type of attack (access to sensitive data).

(1) The operational type of attack

Attack source sent out the instruction to lead to some unauthorized operation eventually. This kind of attack did not require returning data to the source of the attack, so it was difficult to track. The model of operational type attack was as shown in Figure 7.

(2) The data type of attack
After the instruction was emitted from the source of attack, sensitive information were got from the database of sensitive and then returned, or sensitive information were returned to the specified leak point. This kind of attack must return sensitive data, so it was easy to track and monitor. The data type of attack model was as shown in Figure 8.

4. THE OVERALL DESIGN OF THE DETECTION SYSTEM

Figure 9 is the for the basic working principle diagram detection system of Unauthorized attack. As shown in Fig 4, the system mainly includes two core modules: applying for permission analysis module and controlling flow graph building blocks by application program.

![Detection System Principles](image)

**Figure 9.** Unauthorized attack detection system principles

4.1 The analysis module of application permission

The Manifest. xml is the configuration file for the entire application, in which all the privileges, custom user permissions, and each component needed to be protected by permissions of application are needed to be declared in the file. Therefore, it is the most basic part for the test measurement system to analyze for the Manifest. xml file. After the completion of this analysis, we will get a full permissions list of applications, and thus map the function list, to determine the system function which is cannot be accessed. That is to say, the analysis module of application permission analyzes for the Manifest. xml file.

4.2 The construction module of application program controlling flow diagram

As shown in Figure 9, the analysis is mainly based on Smali file parsing to achieve from submission to APK, and then from APK files decompile code to called system function. As for the application of the APK files decompile, we will get Smali file, the file preserves decompile code which is obtained after application translation. Analysis of the file can be obtained the function of applications, the called system function and parameters and other information. In addition, according to this, we can get the called list and order of system function, that is to say, the controlling flow diagram.

4.3 The parsing of Manifest.xml file

The Manifest. xml configuration file is a required file for each Android program. It is located in the root directory of the entire Android application development project, including the each component of Android applications (activities, services, and content provider and radio receiver, etc.). What is more, we can determine how the interaction between these components and these components and other applications by using intent filters and permissions.
System is mainly to make a preliminary judge for the permission of APK through the analysis of the Manifest. xml file. And according to the statement permissions, it can get a controlling list of output risk and a list of available, which are used to compare sensitive permissions and security permissions crossly. As shown in Figure 10: the decision tree used by analyzing the Manifest. xml file.

![Decision Tree](image)

**Figure 10.** Manifest. xml file analysis and decision tree

### 5. IMPLEMENTATION OF DETECTION SYSTEM

#### 5.1 The overview of system

This system is a static checking tool based on the unauthorized attack in Android system, which is mainly divided into two parts, the server and client.

**1. Server side**

The server side is mainly used for storing data in the system, updating and querying function. The server operates in the SAE of Sina cloud computing platform, which is responsible for providing the required data for clients, and updates the data according to the results returned to the client server. The server operates and maintains two database table of the Android system permissions function: “map_lib” and “permission_map”.

- **map_lib** contains 90848 data, including the next step of the mapping function of each function, if the mapping function is a sensitive function of permission, it will include the sensitive infection involved in all the paths.
- **permission_map** is permission mapping table, which is mainly responsible for obtaining sensitive permissions mapping into corresponding exercises by looking up the dictionary of sensitive permissions for each one.

**2. Client side**

The client can run on windows vista/7/8 system, which mainly contains two modules: the analytical module of Manifest. xml file and the parsing module of Smali file. The parsing module Smali is responsible for distinguishing Smali file and the Manifest. xml file from the input of Smali folder. By analyzing the Smali file, it can get the list of called function, then interact the called function with the server side data to distinguish security functions and the sensitivity function. If it is sensitive function, it will query the sensitive infection path to the server. By analyzing the Manifest. xml file, it can get the permissions of Android application, if it is not declared this permission in Manifest.xml
explicitly, but to attack in the Smali file, a warning will be happened. The analysis process of client side is shown in Figure 11.

![Figure 11. The client analysis flow chart](image)

### 5.2 The test of detection system

In order to test the actual operating results of the detection system against unauthorized attack, and the most important to test run efficiency and false alarm rate, we use the detection system of present study to detect 1179 applications in Google platform, and make manual analysis of sampling.

After detection, 613 applications contain vulnerabilities for the sensitive permissions in Manifest. xml file. For these applications, further detection of 123 applications that contain too much permission, these permissions might be used by attackers to launch unauthorized attack. From 123 alarm applications randomly selected 20 applications to make manual analysis, which is mainly to analyze the alarm path to verify the system detection accuracy. The results in this 20 alarm applications, in which there is 9 real unauthorized attacks, they have security threats and other 11 applications are false positives.

Table 1 shows the comparison of the experimental results among three kinds of testing tools.

<table>
<thead>
<tr>
<th>Names</th>
<th>Application number</th>
<th>Number of warning</th>
<th>Number of false alarm</th>
<th>Misstatement rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>ComDroid</td>
<td>50</td>
<td>179</td>
<td>165</td>
<td>92.2%</td>
</tr>
<tr>
<td>DroidChecker</td>
<td>1179</td>
<td>30</td>
<td>15</td>
<td>50%</td>
</tr>
<tr>
<td>coChecker</td>
<td>1179</td>
<td>131</td>
<td>22</td>
<td>16.4%</td>
</tr>
</tbody>
</table>

Through the analysis, the main causes of false alarm system include as follows: 1) Some sensitivity of API function of Android cannot be uniquely determined. For example, the print function () is considered to be a function which might compromise users’ privacy data, but in most cases it still shows normal, only is exploited by attackers in a handful of cases.
2) The sensitive permission dictionary of Android is queried by the Android web document, its maps the sensitive function list on the basis of it. Because of the workload, the 1 edition of the sensitive permissions only has 403, which also has certain promotion space.

6. CONCLUSIONS

Firstly, the Android system architecture and security mechanism were detailed analyzed in this present study. This paper leads to the Android system exist many security vulnerabilities and security risks through carries on the elaboration to the Android system components, security mechanism and its limitations.

Combined with a large number of researches and practices, this paper puts forward a bran-new lightweight unauthorized attacks detection method based on controlling flow detection and matching with Android permission sensitive dictionary. It gives a detailed construction method of controlling flow graph for the first time, and on this basis, it achieved a high degree of automation, high detection efficiency of detection software. At last, through the detection analyzed results of 1179 Android applications and then verifies the compliance of the software design index. Experiments show that, no matter in reducing the rate of false alarm or automatic, compared with the traditional Android application of unauthorized attacks, the new detection tools have a certain upgrade.

7. REFERENCES


Xiao Y., Xu J., Xing Y., Shen Y., Zhuang W. (2015). Design and implementation of mobile tobacco marketing system based on android platform, Tobacco Science and Technology, 48,98-102,