International Journal of Applied Mathematics Electronics and Computers

ISSN:2147-8228 http://dergipark.gov.tr/ijamec Original Research Paper

A Practical Approach to Android Mobile Application Security

Ahmet Caliskan [1], Sakir Tasdemir[2,*]

Accepted : 18/11/2018 Published: 27/12/2018 DOI: 10.18100/ ijamec.2018447400

Abstract: In parallel to rapid developments in computer technology, the number of mobile applications developed for the devices also increases. Mobile applications make life easier, but also bring some risks. These applications may create some weaknesses due to mistakes in the app development or use phase. In this study, a sample security test was performed for mobile application security awareness. This paper related to phising attacks to Android mobile users and data storage security on Android device. The sample mobile application has been decompiled. The malicious code was injected into the sample app. After the code was injected into the sample banking application developed, the user interface was modified. In addition, when the application is open, the user's credit card information is requested. After the user fills information, the credit card information is sent to a different phone number (attacker's phone number) through an SMS. The mobile user is at risk of stealing sensitive information. This study also shows that the data stored in the device can be accessed through the Android Debug Bridge (ADB) shell commands. As a result, this paper shows that the application developer should be more careful during the development phase and the device user should be more careful during the use phase.

Keywords: *Android, Spyware, Reverse Engineering, Vulnerability, Phishing, Security*

1. Introduction

Depending on the advancement of technology, the use of smart devices such as phones and tablets is rapidly increasing. Digital marketing agency We Are Social provides detailed information about mobile device usage. The report states that more than five billion people use mobile devices in the world [1]. The usage rates of operating systems can be displayed interactively on the official website of Kantar Worldpanel [2]. As seen in the reports, the most used operating system is the Android operating system. Therefore, in this study, the Android mobile operating system was preferred.
In the study "Android Applications and Security Breach" [3] some types of cyber attacks and mobile threats that attackers perform to steal user information are described. In addition, the Remote Access Trojan example that called Dendroid was analyzed.
In the study "Android Malware Detection & Protection: A Survey" [4], malware applications on Android are described.
In the paper that called "An Enhanced Security Framework for Reliable Android Operating System" [5], an increase in the number of applications installed as malware in the Android operating system has been mentioned.
The study, titled "An Android-based Trojan Spyware to Study the NotificationListener Service Vulnerability" [6], a trojan application, known as SMS backup, is developed to spy the notifications of other applications.
The study, "Mobil Kötücül Yazılımlar ve Güvenlik Çözümleri Üzerine Bir İnceleme" [7], it was targeted some researches on security solutions for mobile devices to present a comprehensive view.
The study, titled "Android Security" [8], this thesis shows the security-relevant structures of Android's system and application architecture. In order to provide infrastructure-independent education, the exercises are based on Android Virtual Devices (AVDs).
The paper, titled "Mobile Security Testing Approaches and Challenges" [9] presents four security testing approaches for mobile security.
The study, "An effective behavior-based Android malware detection system" [10] proposes a behavior-based malware detection system.
The study titled "Keyboard or Keylogger?: a security analysis of third-party keyboards on Android" [11] is shown that third-party keyboard applications can work as keylogger.
The study titled "Android Zararlı Yazılımlarını Tespit Etme, İmza Oluşturma ve Sınıflandırma" [12], a new malware detection infrastructure, developed for the Android operating system, signature algorithms, correlation with other malware families and evaluation of proposed system are discussed.
In the studies, titled "Mobil Yaşamda Siber Güvenlik Yaklaşımı" [13] and titled "Android Keylogging Threat" [14], mobile security risks and measures are described.
The study, "Mobil Bankacılıkta Güvenlik Sorunlarının Analizi" [15] mobile application security methods are described and presented.
In the studies, "Android Kötücül Yazılım Tespit Sistemleri İncelemesi" [16], "An Android Malware Detection Method Based on AndroidManifest File" [17] "Permission-Based Android Malware Detection" [18] and "MAMA: Manifest Analysis for Malware Detection in Android" [19], AndroidManifest.xml file has been seen that how important it is for application security.
As seen in the literature, if there is not enough security on mobile devices, personal information of the users can be stolen, their privacy can be violated and the applications of the users can be disabled.
In this study has been shown that malicious code can be injected into some applications because of the sensitive data obtained after reviewing the AndroidManifest.xml file. Some mobile applications on the non-rooted device have been reverse

[1] *Kuveyt Türk Participation Bank Research & Development Center, Kocaeli – 41420, TURKEY*
[2]*Computer Engg., Selcuk University, Konya – 42002, TURKEY*
** Corresponding Author: Email: stasdemir@selcuk.edu.tr*

engineered. In addition, some applications were analyzed through ADB terminal commands and it was shown that important information stored on the device can be access.
After injecting new codes into the application in this study;

- Some UI changes (color changes, text changes etc.).
- The Toast message will be displayed every time the application is opened.
- Dialog screen will be displayed. After users fill their credit cards information and the users click the OK button, user's credit cards information will be sent as SMS to the attacker's phone.

As a result of the changes, there is no malfunction in the application.
Some risks arise as a result of carelessness during application development and use. In order to draw attention to these risks, this study was carried out entirely ethically. Also in the figures in this study, information that is thought to remind any original application has been modified and censored. Very importantly, this study does not harm any application developer or company. The pentest (Penetration Test) was carried out entirely on the user's device.

2. Materials and Method

The smart phone using android operating system was preferred in this paper.
Android is an open source operating system developed for mobile devices [20, 21]. Android is Linux based and developed by Google. Nowadays, Android has the largest share in mobile application stores because it is the most widely used mobile operating system.
The tools and system requirements used in the study are as follows;

- Android Studio
- JDK – SDK
- ADB
- Advanced APKTool
- dex2jar
- JD-GUI
- Genymotion
- Android (smart phone) device

3. Reverse Engineering

3.1. Code Injection

The Lenovo P2a42 (Android version 6.0.1) mobile device and Samsung Galaxy Tab A (Android version 5.1.1) were used for code injection. The operations performed on this device can also be performed on the Genymotion virtual device. Sample mobile banking application on the device has been reverse engineered.
First, the application APK (Android Package) name should be detected by using ADB shell commands. Then the APK file must be transferred to the computer.
A connection is established between the computer and the device. After the connection, the APK name of the application should be known. Therefore, the list of all application packages in the connected device is displayed by using ADB commands. After the name of the application's APK file is detected in the list, it is transferred to the computer. As shown in Figure 1, after the APK file has been decompiled, important files such as the AndroidManifest.xml file is appearing.

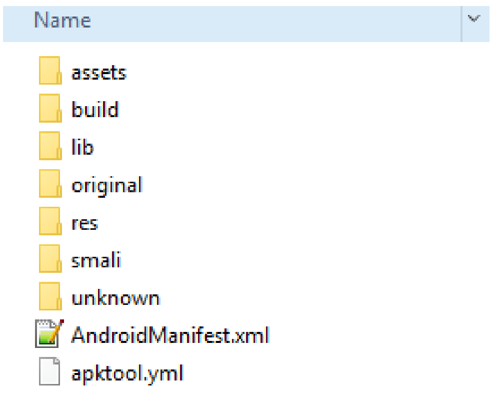

```
Name
assets
build
lib
original
res
smali
unknown
AndroidManifest.xml
apktool.yml
```

Fig. 1. After decompiling APK file.

The Android Smali Assembly Source Code (SMALI) files of the classes used in the application are located in the smali folder. Important files to be used for code injection are in this folder. The AndroidManifest.xml file has been examined to detect the launcher Activity class of the application. As shown in Figure 2, the launcher Activity class was detected. In this way, a new Activity class with the same name will be created and then embedded in the original application.

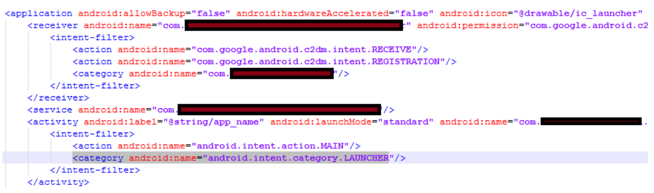

```
<application android:allowBackup="false" android:hardwareAccelerated="false" android:icon="@drawable/ic_launcher"
    <receiver android:name="com.[redacted]" android:permission="com.google.android.c2
        <intent-filter>
            <action android:name="com.google.android.c2dm.intent.RECEIVE"/>
            <action android:name="com.google.android.c2dm.intent.REGISTRATION"/>
            <category android:name="com.[redacted]"/>
        </intent-filter>
    </receiver>
    <service android:name="com.[redacted]"/>
    <activity android:label="@string/app_name" android:launchMode="standard" android:name="com.[redacted].
        <intent-filter>
            <action android:name="android.intent.action.MAIN"/>
            <category android:name="android.intent.category.LAUNCHER"/>
        </intent-filter>
    </activity>
```

Fig. 2. AndroidManifest.xml file.

Another way to detect the launcher activity is to use the ADB logcat command. However, this method is difficult and laborious. After the activity class has been detected, a new project is created in Android Studio. The path of the new launcher Activity class in the new project must be the same as the path of the launcher Activity class in the original application. The malicious code that will run when the application is opened is written in this Activity class. A new Activity class is created in the new project with the same name as the launcher Activity class in the original application. A new application is developed to inject code by using this method.
The new application's APK file is decompiled. After Decompile, the SMALI file of the new Activity class is copied to the location of the SMALI files of the original application. Recompile is performed after making the necessary changes in the AndroidManifest.xml file of the original application. After the recompiling, this application is installed on the device. Figure 3 shows the original screen of the application and the screen that after code injection. In this study, the user interfaces of the application are censored.

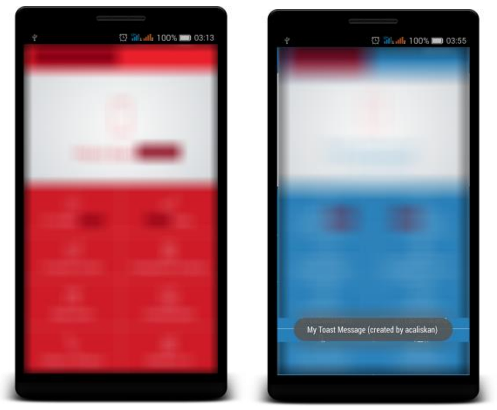


Fig. 3. Original screen and screen that after code injection.

In this study, two different examples were made for mobile application security awareness. Injected codes may vary depending on the intent and purpose of the attacker. For example, an ADB logcat or a JD-GUI tool can be used to find other Activity and classes. Important information on the device can be obtained by a Service that is running when the application is opened.
Also, a sample phishing attack was carried out in the same mobile application. The user interface of the mobile banking application will be used in the user interface of the developed new application. In addition to this, a dialog screen will appear for user's credit card information. After completing the information, the user will proceed with the button "OK". However, the information entered in the dialog screen will be sent as SMS to the number specified (attacker's phone number) in the injected code after the button is pressed.
For this code injection, it is necessary to access the Hexadecimal (HEX) codes of the layout file within the SMALI files. The HEX code to be injected is replaced by the layout HEX code in the original SMALI file. The icon of the mobile application is used in the dialog screen shown to the user for the phishing attack. The AndroidManifest.xml file is also modified to send SMS.
Figure 4 shows the user interface of the original application and the user interface of the phishing attack application. As seen in the figure, phishing attack was carried out with the same screen design.

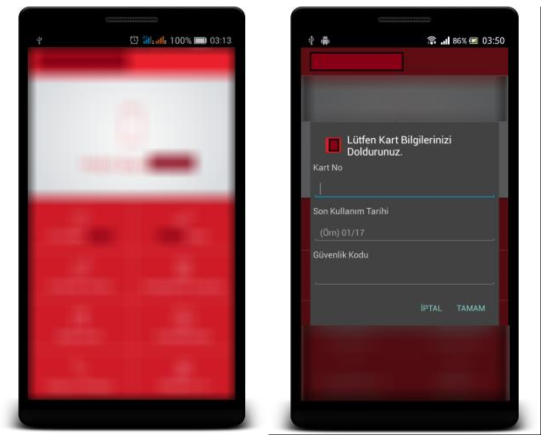
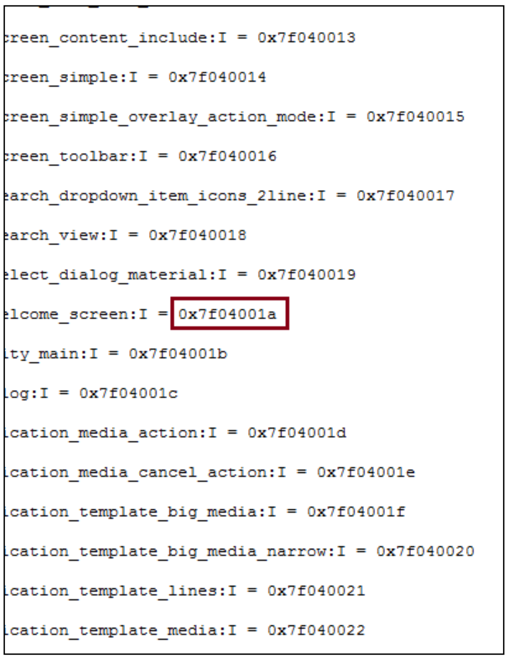


Fig 4. Original application UI and Phishing application UI.

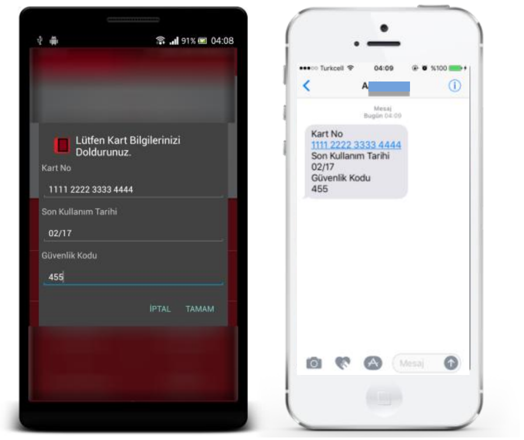
The interface of the phishing app is similar to the interface of the original application. The user will think that this UI belongs to the original application. So the mobile user is at risk of stealing sensitive information. In the previous example, it has been shown that the user interface can be modified by code injection. In the second example, malware is embedded in the original application to obtain the user's information.
Figure 5 shows the layout HEX codes in the SMALI file. The code shown in the marked field will be replaced by the layout HEX code in the SMALI files of the Activity classes.



```
reen_content_include:I = 0x7f040013
reen_simple:I = 0x7f040014
reen_simple_overlay_action_mode:I = 0x7f040015
reen_toolbar:I = 0x7f040016
arch_dropdown_item_icons_2line:I = 0x7f040017
arch_view:I = 0x7f040018
lect_dialog_material:I = 0x7f040019
lcome_screen:I = 0x7f04001a
ty_main:I = 0x7f04001b
og:I = 0x7f04001c
cation_media_action:I = 0x7f04001d
cation_media_cancel_action:I = 0x7f04001e
cation_template_big_media:I = 0x7f04001f
cation_template_big_media_narrow:I = 0x7f040020
cation_template_lines:I = 0x7f040021
cation_template_media:I = 0x7f040022
```

Fig. 5. HEX codes in SMALI file.

After the user card information is entered, card information will be sent to the phone number that embedded in the malware code. Figure 6 shows the dialog screen and a screenshot of the attacker's phone. As shown in these figures, the entered card information was sent to another device through an SMS. The user will switch to the user interface of the original application after pressing the button "OK". No errors were encountered in the application. The application continues to run successfully. There is no attack against the developer or application owner. All these operations are in the device and only affect the owner of the device.



Lütfen Kart Bilgilerinizi Doldurunuz.
Kart No
1111 2222 3333 4444
Son Kullanım Tarihi
02/17
Güvenlik Kodu
455

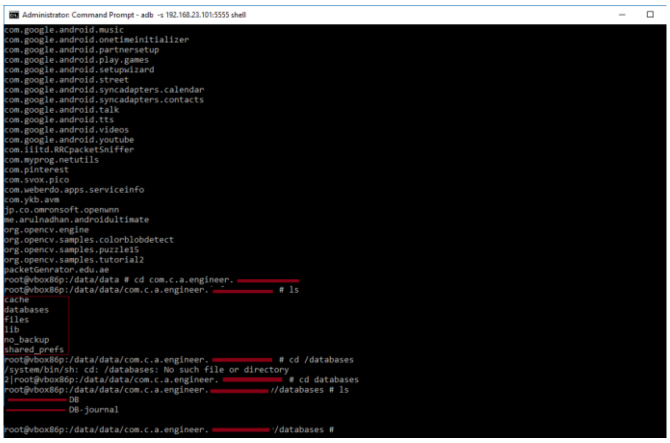
Kart No
1111 2222 3333 4444
Son Kullanım Tarihi
02/17
Güvenlik Kodu
455

Fig. 6. User's credit card information and attacker's phone.

3.2. Data Storage Security in Android Mobile Application

This section shows that some application data stored in the device can be access. The purpose of this section is that application developers should be careful when storing data into the mobile device. Especially, important information such as password, ID, etc. should be stored after it is encrypted. Firstly, a test was performed on the application in the Genymotion virtual device. The connection was established after the Internet Protocol (IP) address of the virtual device was detected by using the ADB shell command.

/data/data/<appname> directory is accessed by using ADB commands. This directory contains SQLite database tables and a SharedPreferences storage file. There is a database in the */data/data/databases* directory. In Figure 7, the database name has been detected by using ADB commands. Database contents will be accessed using sqlite3 commands.

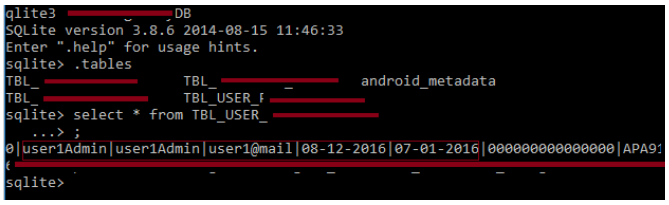


```
Administrator: Command Prompt - adb -s 192.168.23.101:5555 shell
com.google.android.music
com.google.android.onetimeinitializer
com.google.android.partnersetup
com.google.android.play.games
com.google.android.setupwizard
com.google.android.street
com.google.android.syncadapters.calendar
com.google.android.syncadapters.contacts
com.google.android.talk
com.google.android.tts
com.google.android.videos
com.google.android.youtube
com.iiitd.RRCpacketSniffer
com.myprog.netutils
com.pinterest
com.svox.pico
com.weberdo.apps.serviceinfo
com.ykb.avm
jp.co.omronsoft.openwnn
me.arulnadhan.androidultimate
org.opencv.engine
org.opencv.samples.colorblobdetect
org.opencv.samples.puzzle15
org.opencv.samples.tutorial2
packetGenrator.edu.ae
root@vbox86p:/data/data # cd com.c.a.engineer.[redacted]
root@vbox86p:/data/data/com.c.a.engineer.[redacted] # ls
cache
databases
files
lib
no_backup
shared_prefs
root@vbox86p:/data/data/com.c.a.engineer.[redacted] # cd /databases
/system/bin/sh: cd: /databases: No such file or directory
2|root@vbox86p:/data/data/com.c.a.engineer.[redacted] # cd databases
root@vbox86p:/data/data/com.c.a.engineer.[redacted]/databases # ls
[redacted]DB
[redacted]DB-journal

root@vbox86p:/data/data/com.c.a.engineer.[redacted]/databases #
```

Fig. 7. Database in Genymotion.

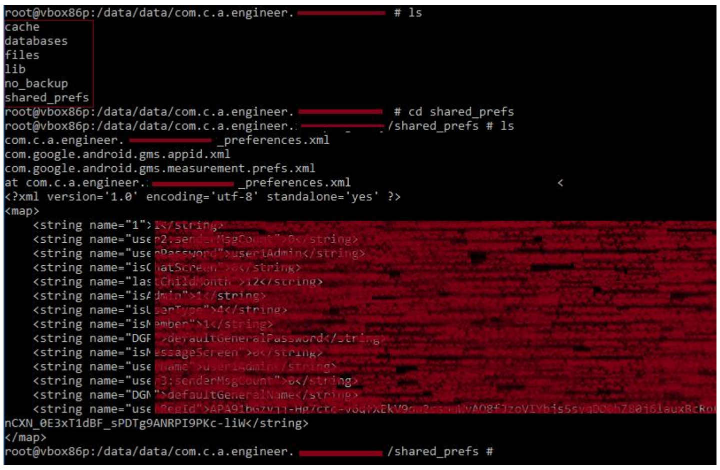
After learning database names, database table names are learned by using sqlite3 commands. After the table names are detected, the table can be examined with Structured Query Language (SQL) "*select"* query. As shown in Figure 8, the data in the application database is stored unencrypted. Thus sensitive data can be accessed.



```
qlite3 [redacted]DB
SQLite version 3.8.6 2014-08-15 11:46:33
Enter ".help" for usage hints.
sqlite> .tables
TBL_[redacted]        TBL_[redacted]     android_metadata
TBL_[redacted]        TBL_USER_F[redacted]
sqlite> select * from TBL_USER_[redacted]
   ...> ;
0|user1Admin|user1Admin|user1@mail|08-12-2016|07-01-2016|000000000000000|APA91
([redacted]
sqlite>
```

Fig. 8. Sensitive data in database.

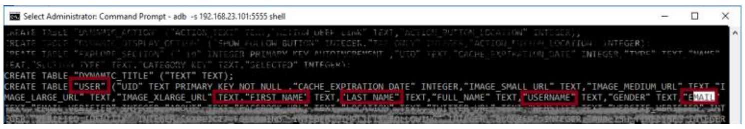
Extensible Markup Language (XML) files are available in the shared_prefs located in the application directory. In the application, if there is data stored by using SharedPreferences, it can be accessed. Figure 9 shows data stored by using SharedPreferences. <*map*> tags contain application data stored using the "key-value" principle.



```
root@vbox86p:/data/data/com.c.a.engineer.[redacted] # ls
cache
databases
files
lib
no_backup
shared_prefs
root@vbox86p:/data/data/com.c.a.engineer.[redacted] # cd shared_prefs
root@vbox86p:/data/data/com.c.a.engineer.[redacted]/shared_prefs # ls
com.c.a.engineer.[redacted]_preferences.xml
com.google.android.gms.appid.xml
com.google.android.gms.measurement.prefs.xml
at com.c.a.engineer.[redacted]_preferences.xml                   <
<?xml version='1.0' encoding='utf-8' standalone='yes' ?>
<map>
    <string name="1">[redacted]
    <string name="use
    <string name="use
    <string name="isC
    <string name="las
    <string name="isA
    <string name="isU
    <string name="isM
    <string name="DGP
    <string name="isM
    <string name="use
    <string name="use
    <string name="DGN
    <string name="use
nCXN_0E3xT1dBF_sPDTg9ANRPI9PKc-liW</string>
</map>
root@vbox86p:/data/data/com.c.a.engineer.[redacted]/shared_prefs #
```

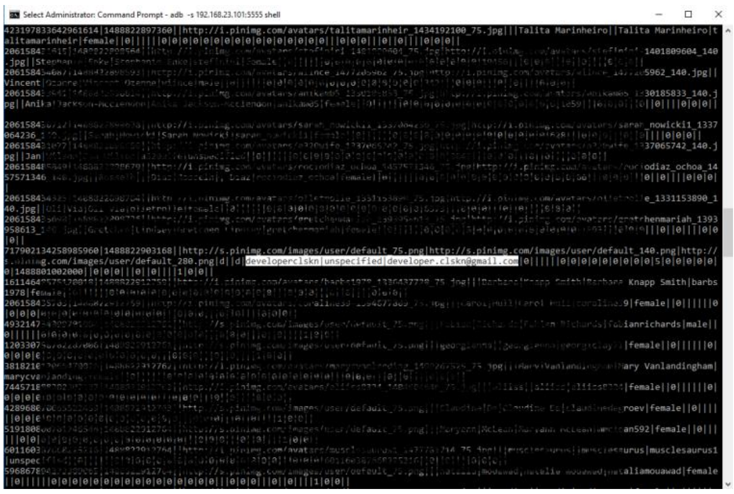
Fig. 9. Sensitive data in SharedPreferences.xml file.

Figure 10 and Figure 11 show that data of a social media application installed on the virtual device can be accessed. The user's e-mail address and the user's name etc. information has been accessed.



```
Select Administrator: Command Prompt - adb -s 192.168.23.101:5555 shell
[redacted]
CREATE TABLE "DYNAMIC_TITLE" ("TEXT" TEXT);
CREATE TABLE "USER" ("UID" TEXT PRIMARY KEY NOT NULL ,"CACHE_EXPIRATION_DATE" INTEGER,"IMAGE_SMALL_URL" TEXT,"IMAGE_MEDIUM_URL" TEXT,"I
MAGE_LARGE_URL" TEXT,"IMAGE_XLARGE_URL" TEXT,"FIRST_NAME" TEXT,"LAST_NAME" TEXT,"FULL_NAME" TEXT,"USERNAME" TEXT,"GENDER" TEXT,"EMAIL"
[redacted]
```

Fig. 10. Social media application data (Table and Column name) in database.

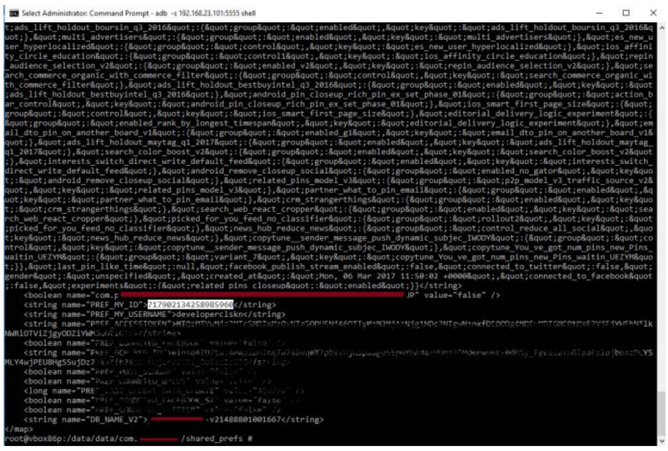


```
Select Administrator: Command Prompt - adb -s 192.168.23.101:5555 shell
423197833642961614|1488822897360||http://i.pinimg.com/avatars/talitamarinheir_1434192100_75.jpg|||Talita Marinheiro||Talita Marinheiro|t
alitamarinheir|female||0|||||0|0|0|0|0|0|0|0|0|0|0|0|0|0|0|0||0|0|0|||0||0|||0|0|0||
2061584[redacted]1401809604_140
.jpg||Stephan[redacted]
2061584[redacted]5962_140.jpg||
Vincent[redacted]
2061584[redacted]30185833_140.j
pg||Anika[redacted]

2061584[redacted]_nowicki1_1337
064236_[redacted]||||0|0|0||
2061584[redacted]37065742_140.j
pg||Jan[redacted]
2061584[redacted]odiaz_ochoa_14
57571346[redacted]||0||0||||0|0|0|
|
2061584[redacted]e_1331153890_1
40.jpg[redacted]
2061584[redacted]henmariah_1393
958613_[redacted]|||0||0|||0|0
|0||
717902134258985960|1488822903168||http://s.pinimg.com/images/user/default_75.png|http://s.pinimg.com/images/user/default_140.png|http://
s.[redacted]g.com/images/user/default_280.png|d||d|developerclskn|unspecified|developer.clskn@gmail.com|0|||||0|0|0|0|0|0|0|0|0|5|0|0|0|0|0|
0|1488801002000||0|0|0|||0||0||||1|0|0||
1611464[redacted] Knapp Smith|barbs
1978|fe[redacted]
2061584[redacted]9|female||0||||||0
|0|0|0|[redacted]
4932147[redacted]ianrichards|male||
0|[redacted]
1203307[redacted]|female||0||||||0|
0|0|0|[redacted]
3818216[redacted]ary Vanlandingham|
marycva[redacted]
7445718[redacted]|female||0||||||0|
0|0|0|[redacted]
4289686[redacted]roev|female||0||||
||0|0|[redacted]
5191808[redacted]an592|female||0|||
|||0|0|[redacted]
6011603[redacted]urus|musclesaurus1
|unspec[redacted]
5968678[redacted]aliamouawad|female
||0||||||0|0|0|0|0|0|0|0|0|0|0|0|0|0|0|0|0||0|0|0|||0||0||||1|0|0||
```

Fig. 11. Social media application data (e-mail) in database.

In this social media application, data stored by using SharedPreferences can be accessed. Information such as username, ID has been accessed. There are many records in this XML file. The file in which the user's name is stored is shown in figure 12.



```
Select Administrator: Command Prompt - adb  -s 192.168.23.101:5555 shell
t;ads_lift_holdout_boursin_q3_2016&quot;:{&quot;group&quot;:&quot;enabled&quot;,&quot;key&quot;:&quot;ads_lift_holdout_boursin_q3_2016&q
uot;},&quot;multi_advertisers&quot;:{&quot;group&quot;:&quot;enabled&quot;,&quot;key&quot;:&quot;multi_advertisers&quot;},&quot;es_new_u
ser_hyperlocalized&quot;:{&quot;group&quot;:&quot;control&quot;,&quot;key&quot;:&quot;es_new_user_hyperlocalized&quot;},&quot;ios_affini
ty_circle_education&quot;:{&quot;group&quot;:&quot;control1&quot;,&quot;key&quot;:&quot;ios_affinity_circle_education&quot;},&quot;repin
_audience_selection_v2&quot;:{&quot;group&quot;:&quot;enabled_v2&quot;,&quot;key&quot;:&quot;repin_audience_selection_v2&quot;},&quot;se
arch_commerce_organic_with_commerce_filter&quot;:{&quot;group&quot;:&quot;control&quot;,&quot;key&quot;:&quot;search_commerce_organic_wi
th_commerce_filter&quot;},&quot;ads_lift_holdout_bestbuyintel_q3_2016&quot;:{&quot;group&quot;:&quot;enabled&quot;,&quot;key&quot;:&quot
;ads_lift_holdout_bestbuyintel_q3_2016&quot;},&quot;android_pin_closeup_rich_pin_ex_set_phase_01&quot;:{&quot;group&quot;:&quot;action_b
ar_control&quot;,&quot;key&quot;:&quot;android_pin_closeup_rich_pin_ex_set_phase_01&quot;},&quot;ios_smart_first_page_size&quot;:{&quot;
group&quot;:&quot;control&quot;,&quot;key&quot;:&quot;ios_smart_first_page_size&quot;},&quot;editorial_delivery_logic_experiment&quot;:{
&quot;group&quot;:&quot;enabled_rank_by_longest_timespan&quot;,&quot;key&quot;:&quot;editorial_delivery_logic_experiment&quot;},&quot;em
ail_dto_pin_on_another_board_v1&quot;:{&quot;group&quot;:&quot;enabled_g1&quot;,&quot;key&quot;:&quot;email_dto_pin_on_another_board_v1&
quot;},&quot;ads_lift_holdout_maytag_q1_2017&quot;:{&quot;group&quot;:&quot;enabled&quot;,&quot;key&quot;:&quot;ads_lift_holdout_maytag_
q1_2017&quot;},&quot;search_color_boost_v2&quot;:{&quot;group&quot;:&quot;enabled&quot;,&quot;key&quot;:&quot;search_color_boost_v2&quot
;},&quot;interests_switch_direct_write_default_feed&quot;:{&quot;group&quot;:&quot;enabled&quot;,&quot;key&quot;:&quot;interests_switch_
direct_write_default_feed&quot;},&quot;android_remove_closeup_social&quot;:{&quot;group&quot;:&quot;enabled_no_gator&quot;,&quot;key&quo
t;:&quot;android_remove_closeup_social&quot;},&quot;related_pins_model_v3&quot;:{&quot;group&quot;:&quot;p2p_model_v3_traffic_source_v2&
quot;,&quot;key&quot;:&quot;related_pins_model_v3&quot;},&quot;partner_what_to_pin_email&quot;:{&quot;group&quot;:&quot;enabled&quot;,&q
uot;key&quot;:&quot;partner_what_to_pin_email&quot;},&quot;crm_strangerthings&quot;:{&quot;group&quot;:&quot;enabled&quot;,&quot;key&quo
t;:&quot;crm_strangerthings&quot;},&quot;search_web_react_cropper&quot;:{&quot;group&quot;:&quot;enabled&quot;,&quot;key&quot;:&quot;sea
rch_web_react_cropper&quot;},&quot;picked_for_you_feed_no_classifier&quot;:{&quot;group&quot;:&quot;rollout2&quot;,&quot;key&quot;:&quot
;picked_for_you_feed_no_classifier&quot;},&quot;news_hub_reduce_news&quot;:{&quot;group&quot;:&quot;control_reduce_all_social&quot;,&quo
t;key&quot;:&quot;news_hub_reduce_news&quot;},&quot;copytune__sender_message_push_dynamic_subjec_IWODY&quot;:{&quot;group&quot;:&quot;co
ntrol&quot;,&quot;key&quot;:&quot;copytune__sender_message_push_dynamic_subjec_IWODY&quot;},&quot;copytune_You_ve_got_num_pins_new_Pins_
waitin_UEZYM&quot;:{&quot;group&quot;:&quot;variant_7&quot;,&quot;key&quot;:&quot;copytune_You_ve_got_num_pins_new_Pins_waitin_UEZYM&quo
t;}},&quot;last_pin_like_time&quot;:null,&quot;facebook_publish_stream_enabled&quot;:false,&quot;connected_to_twitter&quot;:false,&quot;
gender&quot;:&quot;unspecified&quot;,&quot;created_at&quot;:&quot;Mon, 06 Mar 2017 11:50:02 +0000&quot;,&quot;connected_to_facebook&quot
;:false,&quot;experiments&quot;:{&quot;related_pins_closeup&quot;:&quot;enabled&quot;}}</string>
    <boolean name="com.p[redacted]JP" value="false" />
    <string name="PREF_MY_ID">717902134258985960</string>
    <string name="PREF_MY_USERNAME">developerclskn</string>
    <string name="PREF_ACCESSTOKEN">[redacted]
NWRlOTViZjgyODZiYWM[redacted]
    <boolean name="[redacted]
    <string name="P[redacted]
MLY4wjPEU8HgSSujDz7[redacted]
    <boolean name="[redacted]
    <boolean name="[redacted]
    <long name="PRE[redacted]
    <boolean name="[redacted]
    <boolean name="[redacted]
    <string name="DB_NAME_V2">[redacted]-v21488801001667</string>
</map>
root@vbox86p:/data/data/com.[redacted]/shared_prefs #
```

Fig. 12. Social media application data (ID, user name etc.) in SharedPreference.xml file.

4. Conclusion and Recommedations

This work was carried out for mobile application security awareness in the Android mobile operating system. In this study, the Android operating system was preferred because has the largest share in the mobile application store.

As seen in the literature, studies are carried out for the detection of malware applications. The number of malware applications is increasing every day. There are many academic studies in order to detect malware applications. There are also academic studies in which new approaches are presented. In the academic studies where malware detection is detected by a static analysis approach, the AndroidManifest.xml file is examined in detail. In addition to these approaches, malware detection has been realized by classification algorithms. Also in the literature, malware applications developed to demonstrate security risks have been tested in devices. Android malware applications can violate user privacy by using Android Service classes in order to obtain many important data on the device. Service is the classes that perform the background operations in Android applications. It does not provide a UI. Service may not have any interaction with the user. In this way, the attacker will steal the important information of the user without interaction. For this reason, it is useful for users to periodically check the applications in the device and the services running in the background.

Another way to ensure security is to perform scans on the device with known reliable antivirus programs.

In the application development phase, the application developer should consider the APK can be decompiled by attackers.

In this study, a security test was performed on the sample mobile applications to show the risks that application developers and users may face. In addition, this study was carried out ethically. The security testing steps in this study may vary depending on the developer's knowledge, purpose. This study was carried out to show the basic concepts of reverse engineering and mobile security. In addition, this study was carried out to guide application developers and researchers who will work ethically in this regard.

At the end of this study, it is described how the attackers can access the data stored on the device. It is also shown that the data stored using by the SQLite database and SharedPreferences can be accessed. The developer must consider such risks when storing data to the device during the application development phase. For example, sensitive data can be encrypted.

The sample mobile application on the non-root device was reverse engineered. In this study, malicious codes were embedded in the application. Two different examples were shown after embedding codesIn the first example, there are some design changes. For example, in this malware application, the UI color was changed and the Toast message was shown. In the second example, a phishing attack was performed against the user.

The device user should be careful about Android permissions. In the application development phase, especially application developers should develop applications by considering the weaknesses that attackers can use. A vulnerability of the application may cause a great risk for personal safety.

As a result, the application developer should be more careful during the development phase and the device user should be more careful during the use phase. Users should not install applications that require unnecessary permissions on the device. Otherwise, user privacy may be violated.

References

[1] WeAreSocial. 2018 02.12.2018]; Available from: https://wearesocial.com/blog/2018/01/global-digital-report-2018.

[2] KantarWorldPanel. 2018 02.012.2018]; Available from: https://www.kantarworldpanel.com/global/smartphone-os-market-share/.

[3] Benítez-Mejía DGN, Sánchez-Pérez G, and Toscano-Medina LK. Android Applications and Security Breach. in 2016 Third International Conference on Digital Information Processing, Data Mining and Wireless Communications (DIPDMWC). 2016.

[4] Arshad S, et al., Android Malware Detection & Protection: A Survey. International Journal of Advanced Computer Science and Applications, 2016. 7(2): p. 463-475.

[5] Park JH, et al., An Enhanced Security Framework for Reliable Android Operating System. Security Comm. Networks, 2016. 9: p. 528-234.

[6] Abualola H, et al., An Android-based Trojan Spyware to Study the NotificationListener Service Vulnerability. Procedia Computer Science, 2016. 83: p. 465-471.

[7] Utku A and Doğru İA, Mobil Kötücül Yazılımlar ve Güvenlik Çözümleri Üzerine Bir İnceleme. Gazi University Journal of Science, 2016. 4(2): p. 49-64.

[8] Heinl M, Android Security, in Department of Media and Information Technology. 2015, Offenburg University of Applied Sciences: Almanya. p. 92.

[9] Wang Y and Alshboul Y, Mobile Security Testing Approaches and Challenges, in First Conference On Mobile And Secure Services. 2015: Gainesville, Florida/USA.

[10] Zou S, Zhang J, and Lin X, An effective behavior-based Android malware detection system. Security and Communication Networks, 2015. 8(12): p. 2079-2089.

[11] Cho J, Cho G, and Kim H. Keyboard or Keylogger?: a security analysis of third-party keyboards on Android. in 13th Annual Conference on Privacy, Security and Trust (PST). 2015. İzmir.

[12] Acar ÖF. Android Zararlı Yazılımlarını Tespit Etme, İmza Oluşturma ve Sınıflandırma. in 7. Uluslararası Bilgi Güvenliği ve Kriptoloji Konferansı. 2014. İstanbul/Türkiye.

[13] Gökçe KG, Şahinaslan E, and Dincel S, Mobil Yaşamda Siber Güvenlik Yaklaşımı, in 7. Uluslararası Bilgi Güvenliği ve Kriptoloji Konferansı. 2014: İstanbul/Türki. p. 214-221.

[14] Mohsen F and Shehab M. Android Keylogging Threat. in 9th International Conference Conference Collaborative Computing: Networking, Applications and Worksharing (Collaboratecom). 2013.

[15] Kazancı T, Mobil Bankacılıkta Güvenlik Sorunlarının Analizi, in İstanbul Üniversitesi Fen Bilimleri Enstitüsü. 2013, İstanbul Üniversitesi: İstanbul/Türkiye. p. 111.

[16] Kiraz Ö and Doğru İA, Android Kötücül Yazılım Tespit Sistemleri

İncelemesi. Düzce Üniversitesi Bilim ve Teknoloji Dergisi, 2017. 5(1): p. 281-298.

[17] Li X, et al. An Android Malware Detection Method Based on AndroidManifest File. in Proceedings of CCIS2016. 2016. China.

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

[19] Sanz B, et al., MAMA: Manifest Analysis for Malware Detection in Android. Cybernetics and Systems, 2013. 44(6-7): p. 469-488.

[20] Narman AE, Android Programlama. 2013, İstanbul: Kodlab Yayın Dağıtım Yazılım ve Eğitim Hizmetleri San. ve Tic. Ltd. Şti.

[21] Android. 2018 05.12.2018]; Available from: https://www.android.com/everyone/.