Live Attacks Simulation

This project simulates a brute-force attack on a Virtual Machine (VM) provided by TryHackMe, focusing on network security analysis and threat detection. The primary objective is to monitor and analyze network traffic using Snort, a powerful Intrusion Detection System (IDS) and Intrusion Prevention System (IPS). By examining packet logs, we aim to identify security anomalies, detect brute-force attack patterns, and implement effective countermeasures.

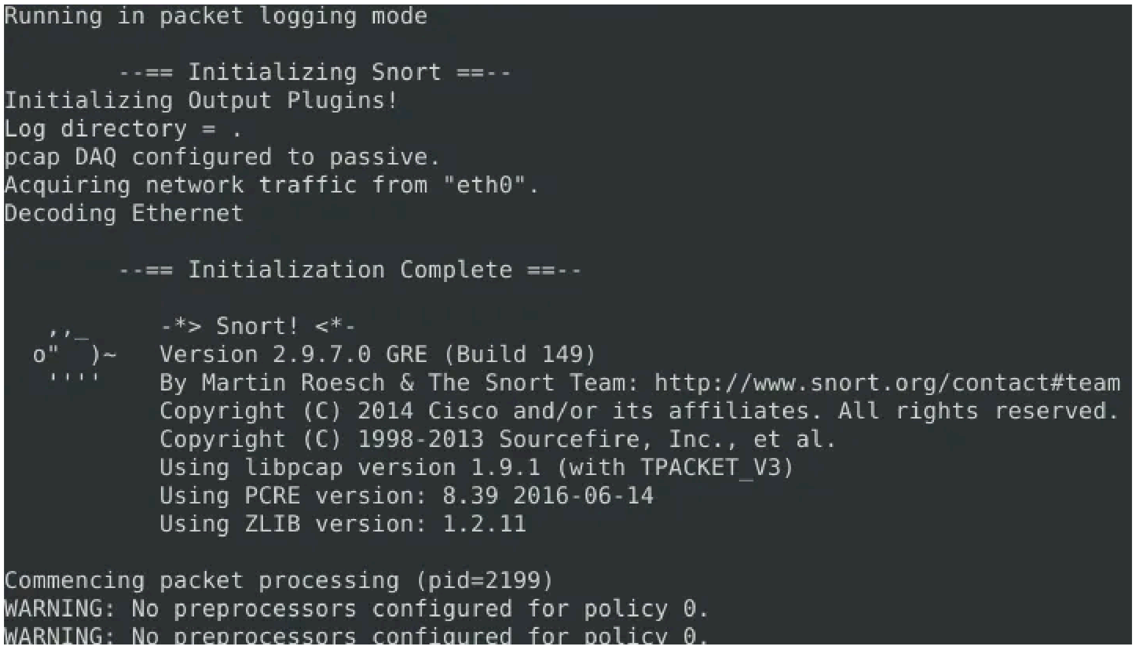
Key Objectives:

- **Traffic Analysis:** Utilize Snort to capture and inspect network traffic for suspicious activity.
- **Threat Detection:** Identify anomalies indicative of brute-force attacks on the system.
- **Mitigation Strategy:** Develop and deploy a custom Snort rule to detect and prevent unauthorized access attempts.

This project strengthens expertise in network security, intrusion detection, and proactive threat prevention, making it a valuable addition to a cybersecurity portfolio.

Let's start to run Snort in sniffer mode. We will use the command `sudo snort -v -l .`, we use the `-l` to log and the `.` to log it in our current directory. With `-v` verbose, display the TCP/IP output in the console.

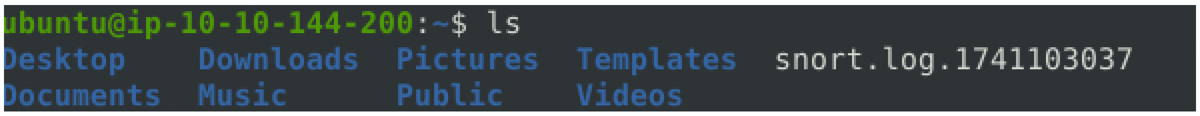
```
sudo snort -v -l .
```



```
Running in packet logging mode

        --== Initializing Snort ==--
Initializing Output Plugins!
Log directory = .
pcap DAQ configured to passive.
Acquiring network traffic from "eth0".
Decoding Ethernet

        --== Initialization Complete ==--

   ,,_     -*> Snort! <*-
  o"  )~   Version 2.9.7.0 GRE (Build 149)
   ''''    By Martin Roesch & The Snort Team: http://www.snort.org/contact#team
           Copyright (C) 2014 Cisco and/or its affiliates. All rights reserved.
           Copyright (C) 1998-2013 Sourcefire, Inc., et al.
           Using libpcap version 1.9.1 (with TPACKET_V3)
           Using PCRE version: 8.39 2016-06-14
           Using ZLIB version: 1.2.11

Commencing packet processing (pid=2199)
WARNING: No preprocessors configured for policy 0.
WARNING: No preprocessors configured for policy 0.
```

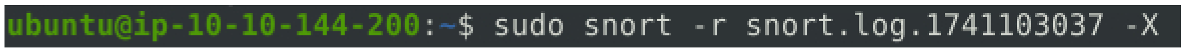
Let that run for 10–15 seconds, then press the keyboard ctrl + c to stop Snort. Let snort finish, when it is done, the terminal will be waiting to accept another command.

Check the log file by using command
`ls`



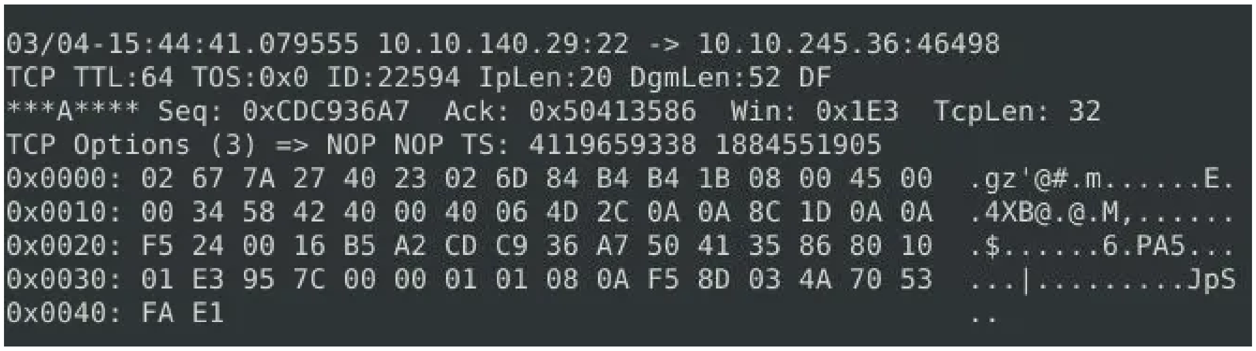
```
ubuntu@ip-10-10-144-200:~$ ls
Desktop    Downloads  Pictures  Templates  snort.log.1741103037
Documents  Music      Public    Videos
```

Investigate the log with `-r` as read and `-X` as display full packet details in HEX

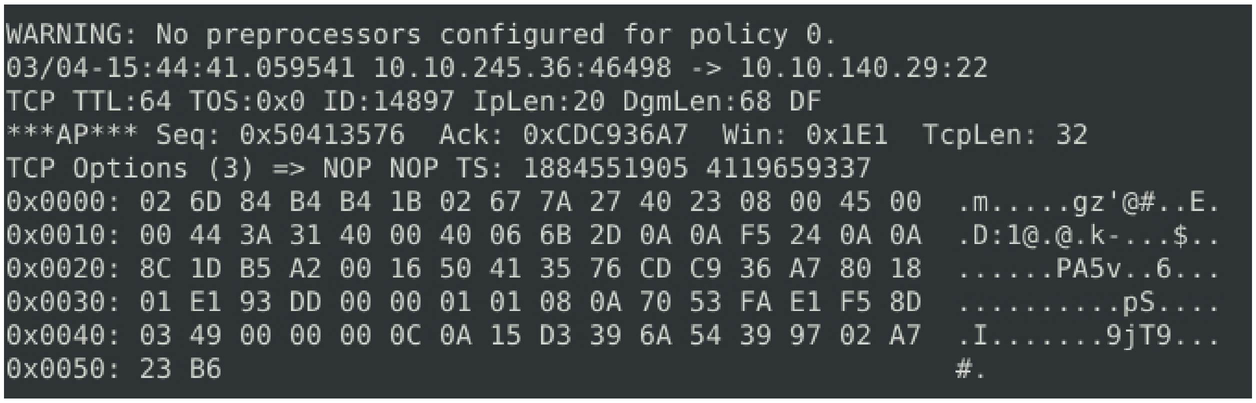


```
ubuntu@ip-10-10-144-200:~$ sudo snort -r snort.log.1741103037 -X
```

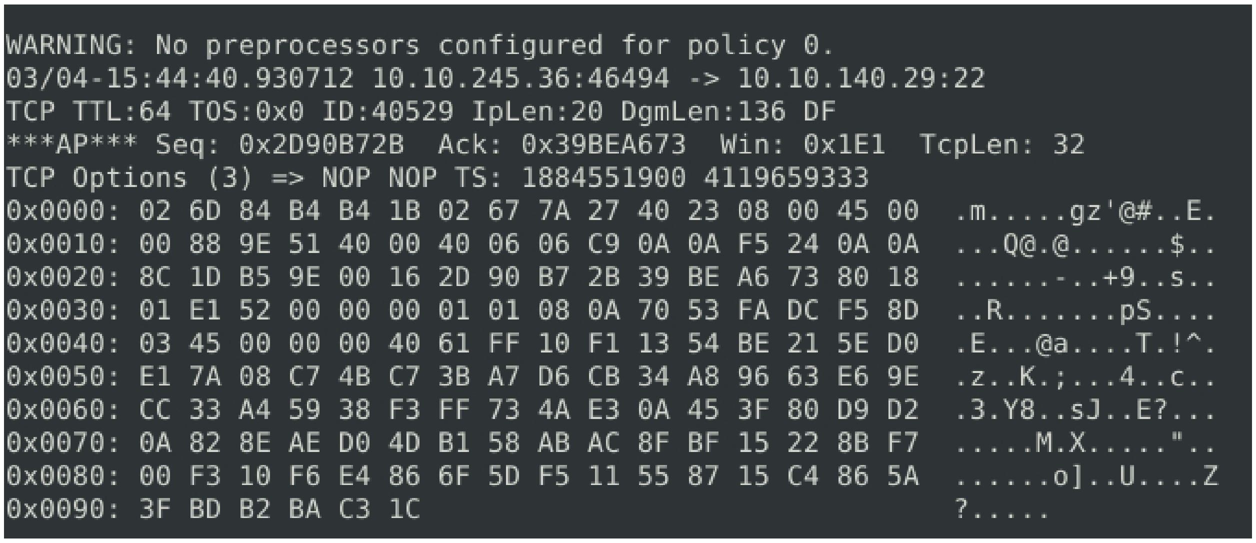
I see that there are some SSH connection request (port 22) keep coming up. Both from destination side and source side. Which there should be no SSH connection within organization.



```
03/04-15:44:41.079555 10.10.140.29:22 -> 10.10.245.36:46498
TCP TTL:64 TOS:0x0 ID:22594 IpLen:20 DgmLen:52 DF
***A**** Seq: 0xCDC936A7  Ack: 0x50413586  Win: 0x1E3  TcpLen: 32
TCP Options (3) => NOP NOP TS: 4119659338 1884551905
0x0000: 02 67 7A 27 40 23 02 6D 84 B4 B4 1B 08 00 45 00  .gz'@#.m......E.
0x0010: 00 34 58 42 40 00 40 06 4D 2C 0A 0A 8C 1D 0A 0A  .4XB@.@.M,......
0x0020: F5 24 00 16 B5 A2 CD C9 36 A7 50 41 35 86 80 10  .$......6.PA5...
0x0030: 01 E3 95 7C 00 00 01 01 08 0A F5 8D 03 4A 70 53  ...|.........JpS
0x0040: FA E1                                            ..
```



```
WARNING: No preprocessors configured for policy 0.
03/04-15:44:41.059541 10.10.245.36:46498 -> 10.10.140.29:22
TCP TTL:64 TOS:0x0 ID:14897 IpLen:20 DgmLen:68 DF
***AP*** Seq: 0x50413576  Ack: 0xCDC936A7  Win: 0x1E1  TcpLen: 32
TCP Options (3) => NOP NOP TS: 1884551905 4119659337
0x0000: 02 6D 84 B4 B4 1B 02 67 7A 27 40 23 08 00 45 00  .m.....gz'@#..E.
0x0010: 00 44 3A 31 40 00 40 06 6B 2D 0A 0A F5 24 0A 0A  .D:1@.@.k-...$..
0x0020: 8C 1D B5 A2 00 16 50 41 35 76 CD C9 36 A7 80 18  ......PA5v..6...
0x0030: 01 E1 93 DD 00 00 01 01 08 0A 70 53 FA E1 F5 8D  ..........pS....
0x0040: 03 49 00 00 00 0C 0A 15 D3 39 6A 54 39 97 02 A7  .I.......9jT9...
0x0050: 23 B6                                            #.
```

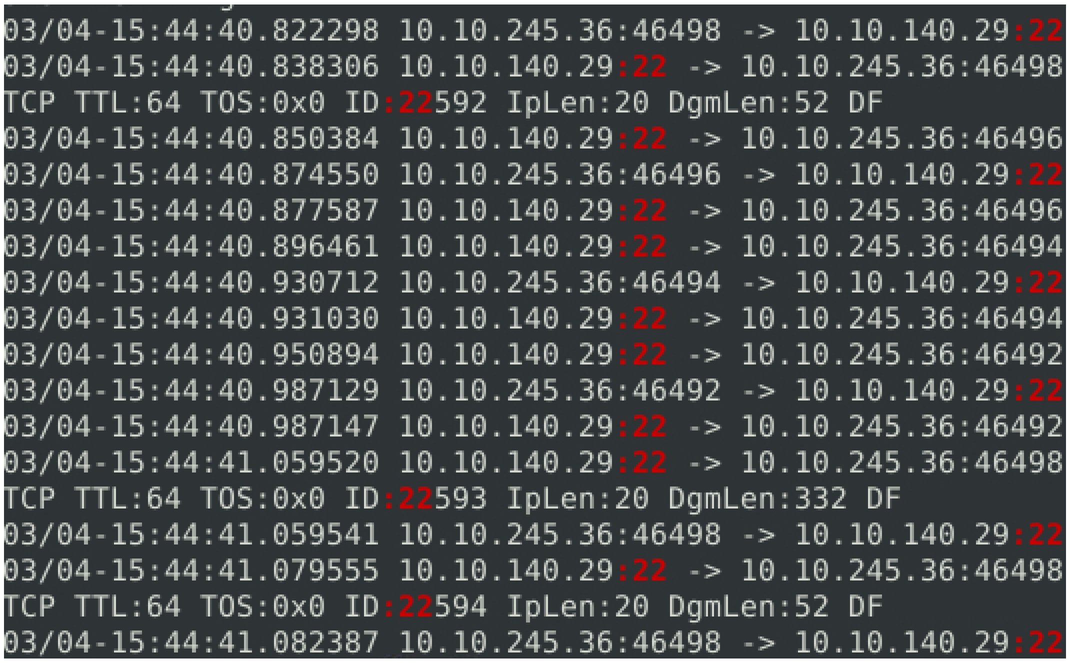


```
WARNING: No preprocessors configured for policy 0.
03/04-15:44:40.930712 10.10.245.36:46494 -> 10.10.140.29:22
TCP TTL:64 TOS:0x0 ID:40529 IpLen:20 DgmLen:136 DF
***AP*** Seq: 0x2D90B72B  Ack: 0x39BEA673  Win: 0x1E1  TcpLen: 32
TCP Options (3) => NOP NOP TS: 1884551900 4119659333
0x0000: 02 6D 84 B4 B4 1B 02 67 7A 27 40 23 08 00 45 00  .m.....gz'@#..E.
0x0010: 00 88 9E 51 40 00 40 06 06 C9 0A 0A F5 24 0A 0A  ...Q@.@......$..
0x0020: 8C 1D B5 9E 00 16 2D 90 B7 2B 39 BE A6 73 80 18  ......-..+9..s..
0x0030: 01 E1 52 00 00 00 01 01 08 0A 70 53 FA DC F5 8D  ..R.......pS....
0x0040: 03 45 00 00 00 40 61 FF 10 F1 13 54 BE 21 5E D0  .E...@a....T.!^.
0x0050: E1 7A 08 C7 4B C7 3B A7 D6 CB 34 A8 96 63 E6 9E  .z..K.;...4..c..
0x0060: CC 33 A4 59 38 F3 FF 73 4A E3 0A 45 3F 80 D9 D2  .3.Y8..sJ..E?...
0x0070: 0A 82 8E AE D0 4D B1 58 AB AC 8F BF 15 22 8B F7  .....M.X....."..
0x0080: 00 F3 10 F6 E4 86 6F 5D F5 11 55 87 15 C4 86 5A  ......o]..U....Z
0x0090: 3F BD B2 BA C3 1C                                ?.....
```

Let filter the log out to see all port 22 log

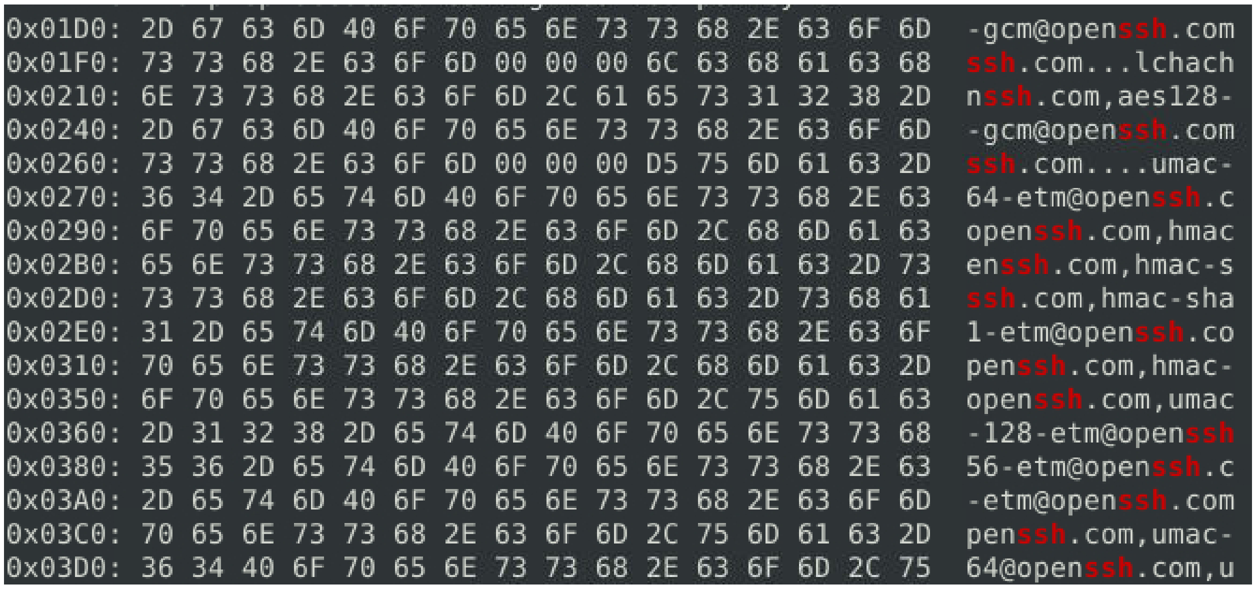
```
sudo snort -r snort.log.1741103037 -X | grep :22
```

The result:



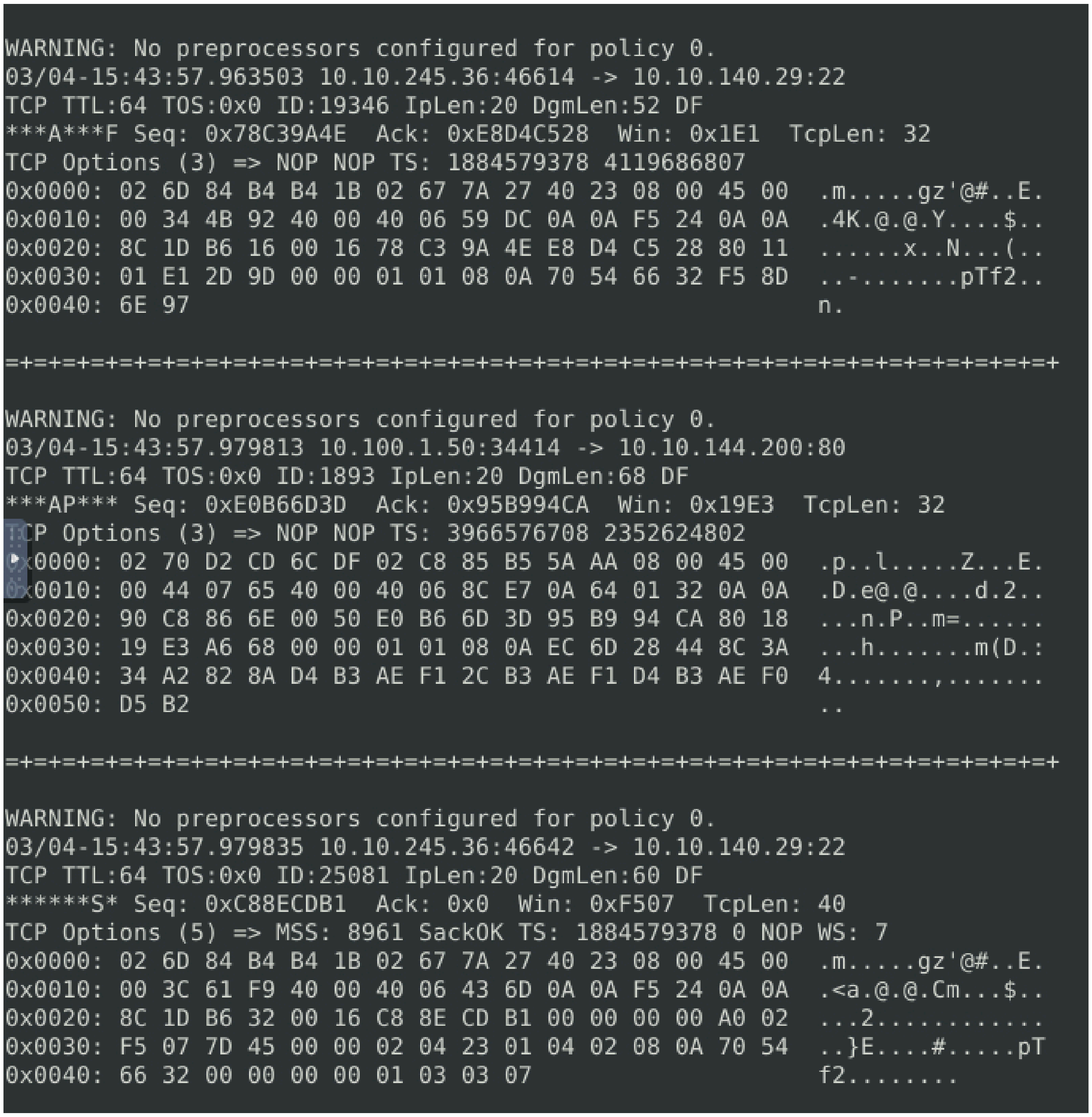
```
03/04-15:44:40.822298 10.10.245.36:46498 -> 10.10.140.29:22
03/04-15:44:40.838306 10.10.140.29:22 -> 10.10.245.36:46498
TCP TTL:64 TOS:0x0 ID:22592 IpLen:20 DgmLen:52 DF
03/04-15:44:40.850384 10.10.140.29:22 -> 10.10.245.36:46496
03/04-15:44:40.874550 10.10.245.36:46496 -> 10.10.140.29:22
03/04-15:44:40.877587 10.10.140.29:22 -> 10.10.245.36:46496
03/04-15:44:40.896461 10.10.140.29:22 -> 10.10.245.36:46494
03/04-15:44:40.930712 10.10.245.36:46494 -> 10.10.140.29:22
03/04-15:44:40.931030 10.10.140.29:22 -> 10.10.245.36:46494
03/04-15:44:40.950894 10.10.140.29:22 -> 10.10.245.36:46492
03/04-15:44:40.987129 10.10.245.36:46492 -> 10.10.140.29:22
03/04-15:44:40.987147 10.10.140.29:22 -> 10.10.245.36:46492
03/04-15:44:41.059520 10.10.140.29:22 -> 10.10.245.36:46498
TCP TTL:64 TOS:0x0 ID:22593 IpLen:20 DgmLen:332 DF
03/04-15:44:41.059541 10.10.245.36:46498 -> 10.10.140.29:22
03/04-15:44:41.079555 10.10.140.29:22 -> 10.10.245.36:46498
TCP TTL:64 TOS:0x0 ID:22594 IpLen:20 DgmLen:52 DF
03/04-15:44:41.082387 10.10.245.36:46498 -> 10.10.140.29:22
```

I then used grep to search for ssh in the packets with the command `sudo snort -r snort.log.1741103037 -X | grep "ssh"`.



```
0x01D0: 2D 67 63 6D 40 6F 70 65 6E 73 73 68 2E 63 6F 6D  -gcm@openssh.com
0x01F0: 73 73 68 2E 63 6F 6D 00 00 00 6C 63 68 61 63 68  ssh.com...lchach
0x0210: 6E 73 73 68 2E 63 6F 6D 2C 61 65 73 31 32 38 2D  nssh.com,aes128-
0x0240: 2D 67 63 6D 40 6F 70 65 6E 73 73 68 2E 63 6F 6D  -gcm@openssh.com
0x0260: 73 73 68 2E 63 6F 6D 00 00 00 D5 75 6D 61 63 2D  ssh.com....umac-
0x0270: 36 34 2D 65 74 6D 40 6F 70 65 6E 73 73 68 2E 63  64-etm@openssh.c
0x0290: 6F 70 65 6E 73 73 68 2E 63 6F 6D 2C 68 6D 61 63  openssh.com,hmac
0x02B0: 65 6E 73 73 68 2E 63 6F 6D 2C 68 6D 61 63 2D 73  enssh.com,hmac-s
0x02D0: 73 73 68 2E 63 6F 6D 2C 68 6D 61 63 2D 73 68 61  ssh.com,hmac-sha
0x02E0: 31 2D 65 74 6D 40 6F 70 65 6E 73 73 68 2E 63 6F  1-etm@openssh.co
0x0310: 70 65 6E 73 73 68 2E 63 6F 6D 2C 68 6D 61 63 2D  penssh.com,hmac-
0x0350: 6F 70 65 6E 73 73 68 2E 63 6F 6D 2C 75 6D 61 63  openssh.com,umac
0x0360: 2D 31 32 38 2D 65 74 6D 40 6F 70 65 6E 73 73 68  -128-etm@openssh
0x0380: 35 36 2D 65 74 6D 40 6F 70 65 6E 73 73 68 2E 63  56-etm@openssh.c
0x03A0: 2D 65 74 6D 40 6F 70 65 6E 73 73 68 2E 63 6F 6D  -etm@openssh.com
0x03C0: 70 65 6E 73 73 68 2E 63 6F 6D 2C 75 6D 61 63 2D  penssh.com,umac-
0x03D0: 36 34 40 6F 70 65 6E 73 73 68 2E 63 6F 6D 2C 75  64@openssh.com,u
```

So let’s narrow it down and take a look at that packet. To do this I used the command
`sudo snort -r snort.log.1741103037 -X -n 30`, this will only output the first 30 packets to the terminal.



```
WARNING: No preprocessors configured for policy 0.
03/04-15:43:57.963503 10.10.245.36:46614 -> 10.10.140.29:22
TCP TTL:64 TOS:0x0 ID:19346 IpLen:20 DgmLen:52 DF
***A***F Seq: 0x78C39A4E  Ack: 0xE8D4C528  Win: 0x1E1  TcpLen: 32
TCP Options (3) => NOP NOP TS: 1884579378 4119686807
0x0000: 02 6D 84 B4 B4 1B 02 67 7A 27 40 23 08 00 45 00  .m.....gz'@#..E.
0x0010: 00 34 4B 92 40 00 40 06 59 DC 0A 0A F5 24 0A 0A  .4K.@.@.Y....$..
0x0020: 8C 1D B6 16 00 16 78 C3 9A 4E E8 D4 C5 28 80 11  ......x..N...(..
0x0030: 01 E1 2D 9D 00 00 01 01 08 0A 70 54 66 32 F5 8D  ..-.......pTf2..
0x0040: 6E 97                                            n.

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

WARNING: No preprocessors configured for policy 0.
03/04-15:43:57.979813 10.100.1.50:34414 -> 10.10.144.200:80
TCP TTL:64 TOS:0x0 ID:1893 IpLen:20 DgmLen:68 DF
***AP*** Seq: 0xE0B66D3D  Ack: 0x95B994CA  Win: 0x19E3  TcpLen: 32
TCP Options (3) => NOP NOP TS: 3966576708 2352624802
0x0000: 02 70 D2 CD 6C DF 02 C8 85 B5 5A AA 08 00 45 00  .p..l.....Z...E.
0x0010: 00 44 07 65 40 00 40 06 8C E7 0A 64 01 32 0A 0A  .D.e@.@....d.2..
0x0020: 90 C8 86 6E 00 50 E0 B6 6D 3D 95 B9 94 CA 80 18  ...n.P..m=......
0x0030: 19 E3 A6 68 00 00 01 01 08 0A EC 6D 28 44 8C 3A  ...h.......m(D.:
0x0040: 34 A2 82 8A D4 B3 AE F1 2C B3 AE F1 D4 B3 AE F0  4.......,.......
0x0050: D5 B2                                            ..

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

WARNING: No preprocessors configured for policy 0.
03/04-15:43:57.979835 10.10.245.36:46642 -> 10.10.140.29:22
TCP TTL:64 TOS:0x0 ID:25081 IpLen:20 DgmLen:60 DF
******S* Seq: 0xC88ECDB1  Ack: 0x0  Win: 0xF507  TcpLen: 40
TCP Options (5) => MSS: 8961 SackOK TS: 1884579378 0 NOP WS: 7
0x0000: 02 6D 84 B4 B4 1B 02 67 7A 27 40 23 08 00 45 00  .m.....gz'@#..E.
0x0010: 00 3C 61 F9 40 00 40 06 43 6D 0A 0A F5 24 0A 0A  .<a.@.@.Cm...$..
0x0020: 8C 1D B6 32 00 16 C8 8E CD B1 00 00 00 00 A0 02  ...2............
0x0030: F5 07 7D 45 00 00 02 04 23 01 04 02 08 0A 70 54  ..}E....#.....pT
0x0040: 66 32 00 00 00 00 01 03 03 07                    f2........
```

Let use power of AI to see what these connection type could be mean :)))) (P/s:

Best way to use AI in 2025 for cybersecurity)

Looking at this log entry:

`03/04-15:43:57.963503 10.10.245.36:46614 -> 10.10.140.29:22`

Breakdown:

- **Timestamp:** `03/04-15:43:57.963503` → The event happened on **March 4th at 15:43:57.963503 (3:43 PM UTC or local time).**
- **Source IP & Port:** `10.10.245.36:46614`
  - The request is coming from **IP 10.10.245.36**
  - The **source port** is `46614`, which is likely a **random ephemeral port** assigned by the client.
- **Destination IP & Port:** `10.10.140.29:22`
  - The traffic is going to **IP 10.10.140.29**
  - The **destination port** is `22`, which is the default **SSH port**.

Interpretation:

This log entry indicates that a device (`10.10.245.36`) is attempting to **connect to an SSH server** (`10.10.140.29`).

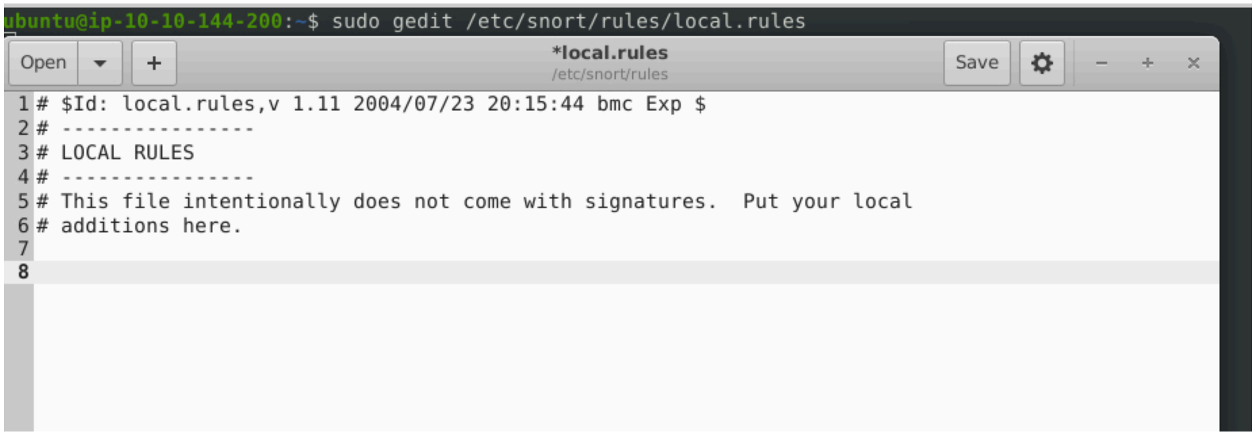
- This could be **a legitimate SSH login attempt** from a user.
- If you see **multiple repeated connections**, it could indicate **brute-force attacks** or unauthorized access attempts.
- If this is unexpected, you might want to check **who is making the connection** and whether it was **successful** (look for any SSH authentication logs on the server).

With these information we have, time to write some rule. Here are a few points to remember:

- Create the rule and test it with “-A console” mode.

- Use **"-A full"** mode and the **default log path** to stop the attack.
- Write the correct rule and run the Snort in IPS "-A full" mode.
- **Block the traffic at least for a minute** and then the flag file will appear on your desktop.

First, we need to open the local.rules file in a text editor. Using the command `sudo gedit /etc/snort/rules/local.rules`, and press enter



```
ubuntu@ip-10-10-144-200:~$ sudo gedit /etc/snort/rules/local.rules
```

```
# $Id: local.rules,v 1.11 2004/07/23 20:15:44 bmc Exp $
# ----------------
# LOCAL RULES
# ----------------
# This file intentionally does not come with signatures.  Put your local
# additions here.


```

Here will be our Snort rule according to information that we have so far:

```
drop tcp any 22 <> any any (msg: "SSH connection attempted"; sid:100001; rev:1
```

Save (ctrl + s) and X out of the text editor window, and your back in the terminal.

What This Rule Does

- It **blocks** (drops) **any TCP traffic** on **port 22** (SSH), **both inbound and outbound**.
- It **logs** an alert message: `"SSH connection attempted"`.

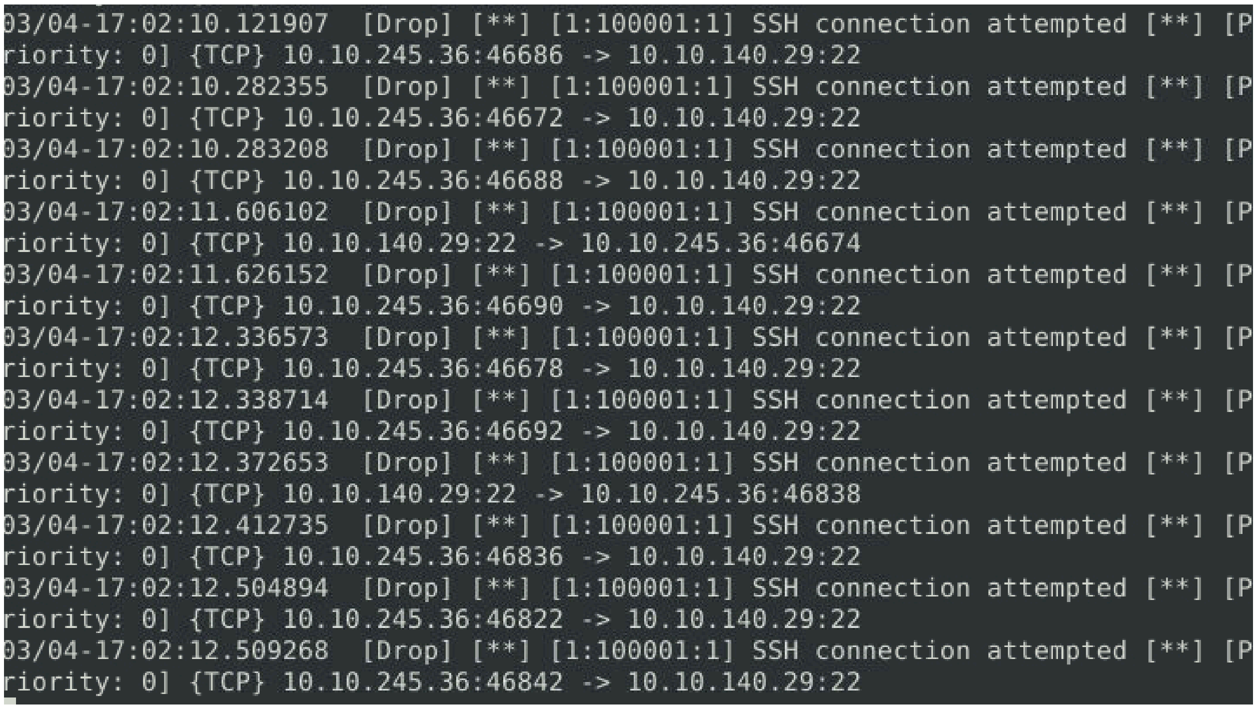
- Useful for environments where **SSH traffic needs to be blocked** to prevent unauthorized access or **exfiltration** via SSH.
- With a unique Snort ID (SID) is 100001 and revision (rev) 1 of the rule.

Let run the rule with follow command:

```
sudo snort -c /etc/snort/snort.conf -q -Q --daq afpacket -i eth0:eth1 -A console
```

Note:

- If you want to **log all details for later analysis**, use `full`.
- If you want to **monitor Snort alerts in real time**, use `console`.



```
03/04-17:02:10.121907  [Drop] [**] [1:100001:1] SSH connection attempted [**] [P
riority: 0] {TCP} 10.10.245.36:46686 -> 10.10.140.29:22
03/04-17:02:10.282355  [Drop] [**] [1:100001:1] SSH connection attempted [**] [P
riority: 0] {TCP} 10.10.245.36:46672 -> 10.10.140.29:22
03/04-17:02:10.283208  [Drop] [**] [1:100001:1] SSH connection attempted [**] [P
riority: 0] {TCP} 10.10.245.36:46688 -> 10.10.140.29:22
03/04-17:02:11.606102  [Drop] [**] [1:100001:1] SSH connection attempted [**] [P
riority: 0] {TCP} 10.10.140.29:22 -> 10.10.245.36:46674
03/04-17:02:11.626152  [Drop] [**] [1:100001:1] SSH connection attempted [**] [P
riority: 0] {TCP} 10.10.245.36:46690 -> 10.10.140.29:22
03/04-17:02:12.336573  [Drop] [**] [1:100001:1] SSH connection attempted [**] [P
riority: 0] {TCP} 10.10.245.36:46678 -> 10.10.140.29:22
03/04-17:02:12.338714  [Drop] [**] [1:100001:1] SSH connection attempted [**] [P
riority: 0] {TCP} 10.10.245.36:46692 -> 10.10.140.29:22
03/04-17:02:12.372653  [Drop] [**] [1:100001:1] SSH connection attempted [**] [P
riority: 0] {TCP} 10.10.140.29:22 -> 10.10.245.36:46838
03/04-17:02:12.412735  [Drop] [**] [1:100001:1] SSH connection attempted [**] [P
riority: 0] {TCP} 10.10.245.36:46836 -> 10.10.140.29:22
03/04-17:02:12.504894  [Drop] [**] [1:100001:1] SSH connection attempted [**] [P
riority: 0] {TCP} 10.10.245.36:46822 -> 10.10.140.29:22
03/04-17:02:12.509268  [Drop] [**] [1:100001:1] SSH connection attempted [**] [P
riority: 0] {TCP} 10.10.245.36:46842 -> 10.10.140.29:22
```

<u>Congratulation,</u> we have stop the attack and block all the Malicious IP.