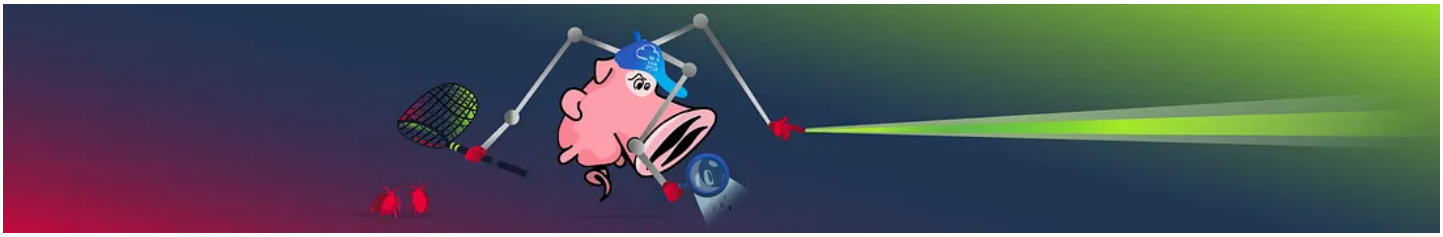


Snort | TryHackMe — Write-up

igor_sec · Follow
22 min read · Jul 20, 2023

SNORT is an **open-source, rule-based** Network Intrusion Detection and Prevention System **(NIDS/NIPS).** It was developed and still maintained by Martin Roesch, open-source contributors, and the Cisco Talos team.

The official description: *"Snort is the foremost Open Source Intrusion Prevention System (IPS) in the world. Snort IPS uses a series of rules that help define malicious network activity and uses those rules to find packets that match against them and generate alerts for users."*

In this room, we will learn how to use Snort to detect real-time threats, analyse recorded traffic files and identify anomalies.

Link: https://tryhackme.com/room/snort

Task 2: Interactive Material and VM

Navigate to the Task-Exercises folder and run the command “./.easy.sh” and write the output

Answer: Too Easy!

```
ubuntu@ip-[redacted]:~/Desktop/Task-Exercises$ ./.easy.sh
Too Easy!
```

Task 3: Introduction to IDS/IPS

There are two main types of IDS systems;

- Network Intrusion Detection System (NIDS) — NIDS monitors the traffic flow from various areas of the network. The aim is to investigate the traffic on the entire subnet. If a signature is identified, an alert is created.
- Host-based Intrusion Detection System (HIDS) — HIDS monitors the traffic flow from a single endpoint device. The aim is to investigate the traffic on a particular device. If a signature is identified, an alert is created.

There are four main types of IPS systems;

- Network Intrusion Prevention System (NIPS) — NIPS monitors the traffic flow from various areas of the network. The aim is to protect the traffic on the entire subnet. If a signature isidentified, the connection is terminated.

- Behaviour-based Intrusion Prevention System (Network Behaviour Analysis — NBA) — Behaviour-based systems monitor the traffic flow from various areas of the network. The aim is to protect the traffic on the entire subnet. If a signature is identified, **the connection is terminated.**

Network Behaviour Analysis System works similar to NIPS. The difference between NIPS and Behaviour-based is; behaviour based systems require a training period (also known as “baselining”) to learn the normal traffic and differentiate the malicious traffic and threats. This model provides more efficient results against new threats.

The system is trained to know the “normal” to detect “abnormal”. The training period is crucial to avoid any false positives. In case of any security breach during the training period, the results will be highly problematic. Another critical point is to ensure that the system is well trained to recognise benign activities.

- Wireless Intrusion Prevention System (WIPS) — WIPS monitors the traffic flow from of wireless network. The aim is to protect the wireless traffic and stop possible attacks launched from there. If a signature is identified, the connection is terminated.

- Host-based Intrusion Prevention System (HIPS) — HIPS actively protects the traffic flow from a single endpoint device. The aim is to investigate the traffic on a particular device. If a signature is identified, **the connection is terminated.**

HIPS working mechanism is similar to HIDS. The difference between them is that **while HIDS creates alerts for threats, HIPS stops the threats by terminating the connection.**

Detection/Prevention Techniques

There are three main detection and prevention techniques used in IDS and IPS solutions;

Signature-Based- This technique relies on rules that identify the specific patterns of the known malicious behaviour. This model helps detect known threats.

Behaviour-Based- This technique identifies new threats with new patterns that pass through signatures. The model compares the known/normal with unknown/abnormal behaviours. This model helps detect previously unknown or new threats.

Policy-Based-This technique compares detected activities with system configuration and security policies. This model helps detect policy violations.

"Snort can be deployed inline to stop these packets, as well. Snort has three primary uses: As a packet sniffer like tcpdump, as a packet logger — which is useful for network traffic debugging, or it can be used as a full-blown network intrusion prevention system. Snort can be downloaded and configured for personal and business use alike."

SNORT is an open-source, rule-based Network Intrusion Detection and Prevention System (NIDS/NIPS). It was developed and still maintained by Martin Roesch, open-source contributors, and the Cisco Talos team.

Capabilities of Snort;

- Live traffic analysis
- Attack and probe detection
- Packet logging
- Protocol analysis
- Real-time alerting
- Modules & plugins
- Pre-processors
- Cross-platform support! (Linux & Windows)

Snort has three main use models;

- Sniffer Mode — Read IP packets and prompt them in the console application.
- Packet Logger Mode — Log all IP packets (inbound and outbound) that visit the network.

- **NIDS (**Network Intrusion Detection System) and NIPS (Network Intrusion Prevention System) Modes — Log/drop the packets that are deemed as malicious according to the user-defined rules.

Answer the questions below

Which snort mode can help you stop the threats on a local machine?

Answer: HIPS

Which snort mode can help you detect threats on a local network?

Answer: NIDS

Which snort mode can help you detect the threats on a local machine?

Answer: HIDS

Which snort mode can help you stop the threats on a local network?

Answer: NIPS

Which snort mode works similar to NIPS mode?

Answer: NBA

According to the official description of the snort, what kind of NIPS is it?

Answer: full-blown

NBA training period is also known as ...

Answer: baselining

Task 4: First Interaction with Snort

Here are some of the parameters to help us solve the questions.

- **V / — version** -This parameter provides information about your instance version.

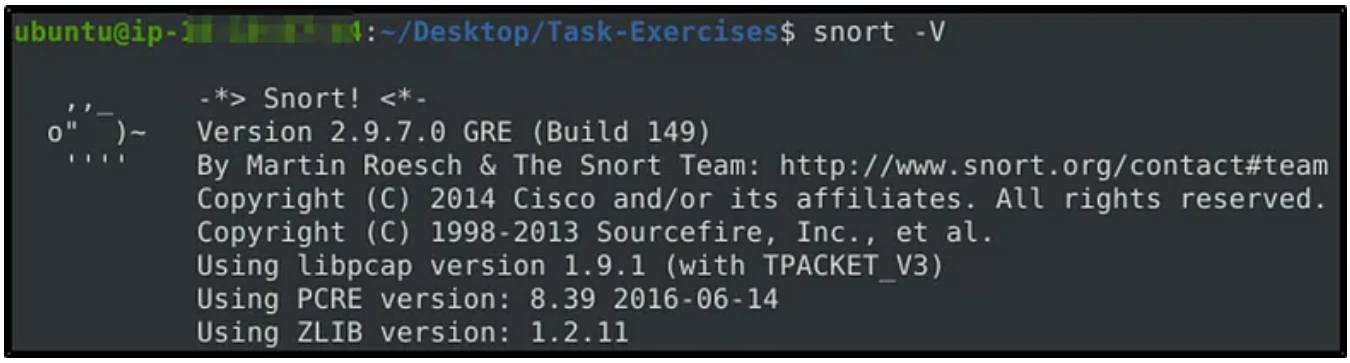
- **-c** -Identifying the configuration file
- **-T** -Snort's self-test parameter, you can test your setup with this parameter.
- **-q** -Quiet mode prevents snort from displaying the default banner and initial information about your setup.

Run the Snort instance and check the build number.

Answer: 149

Refer above for the parameter to get the version information about the installed snort in the machine.

```
snort -V
```

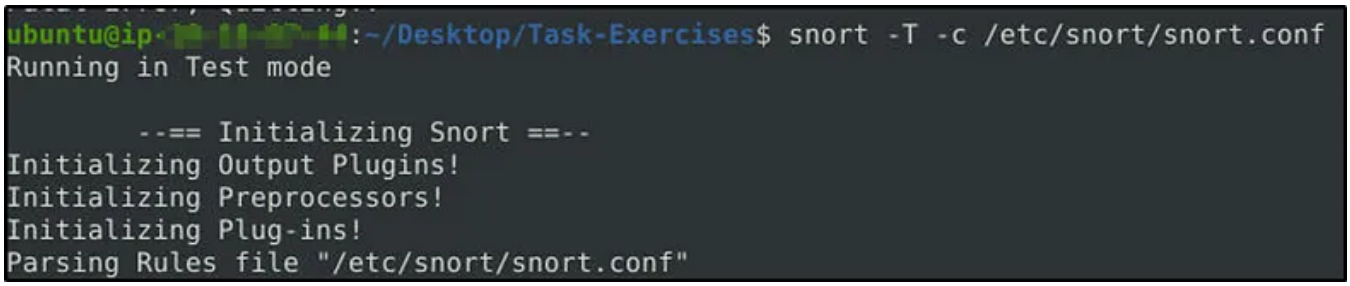


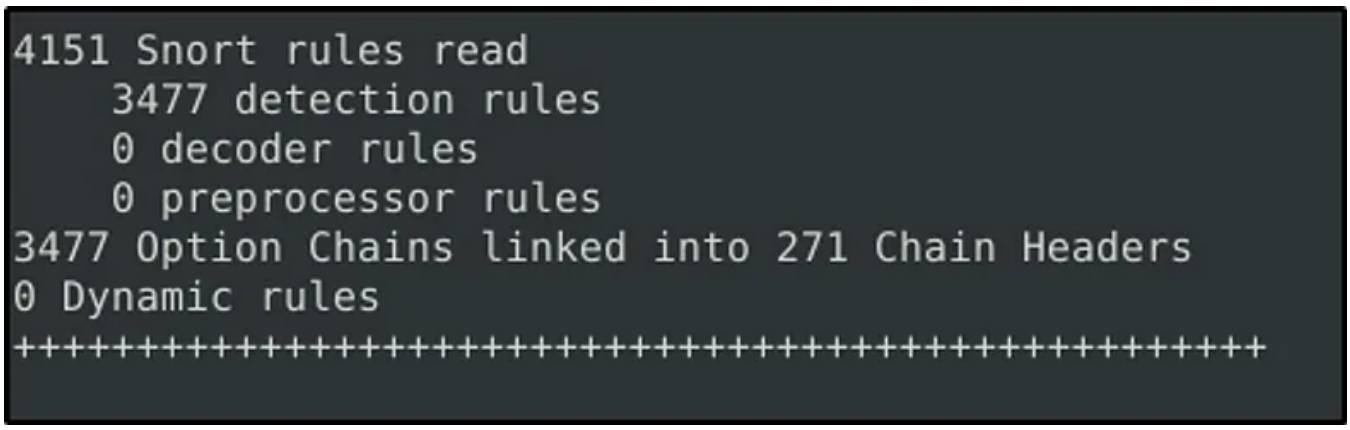
Test the current instance with "**/etc/snort/snort.conf**" file and check how many rules are loaded with the current build.

Answer: 4151

Refer from above the used parameters in the command.

```
snort -T -c /etc/snort/snort.conf
```

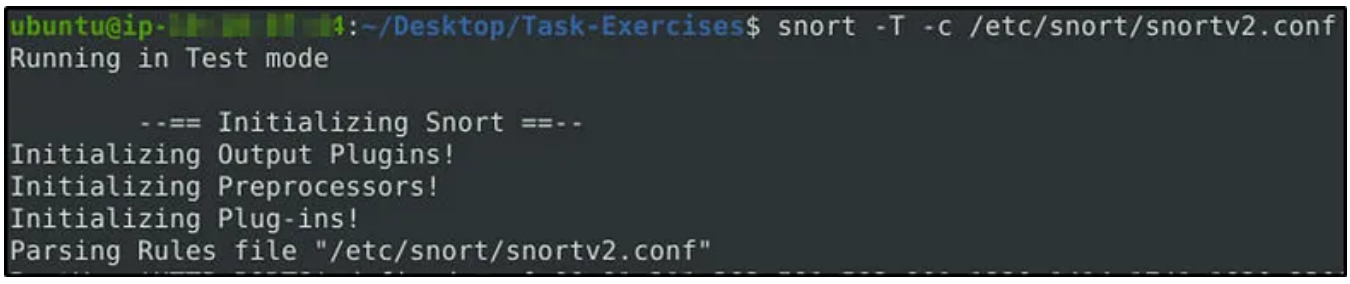


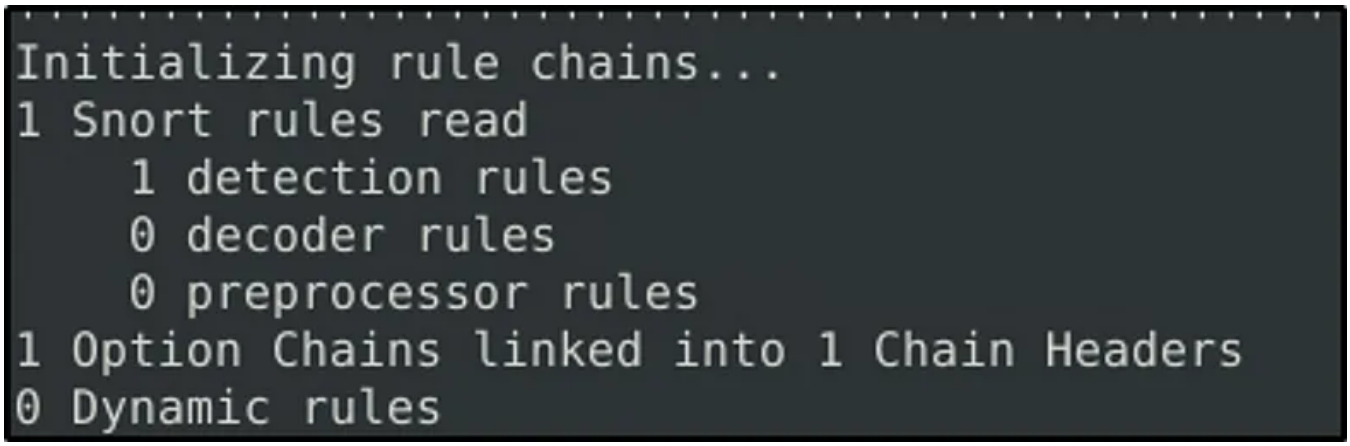


Test the current instance with "**/etc/snort/snortv2.conf**" file and check how many rules are loaded with the current build.

Answer: 1

```
snort -T -c /etc/snort/snortv2.conf
```





Task 5: Operation Mode 1: Sniffer Mode

Like tcpdump, Snort has various flags capable of viewing various data about the packet it is ingesting.

Sniffer mode parameters:

- **-v** -Verbose. Display the TCP/IP output in the console.
- **-d** -Display the packet data (payload).
- **-e** -Display the link-layer (TCP/IP/UDP/ICMP) headers.
- **-X** -Display the full packet details in HEX.
- **-i** -This parameter helps to define a specific network interface to listen/sniff. Once you have multiple interfaces, you can choose a specific interface to sniff.

Sniffing with parameter “-i”

```
udo snort -v-i eth0
```

Sniffing with -v

```
sudo snort -v
```

Sniffing with parameter “-d”

```
sudo snort -d
```

Sniffing with parameter “-de”

```
sudo snort -de
```

Sniffing with parameter “-X”

```
sudo snort -X
```

Note that you can use the parameters both in combined and separated form as follows;

- **snort -v**
- **snort -vd**
- **snort -de**
- **snort -v -d -e**
- **snort -X**

Task 6: Operation Mode 2: Packet Logger Mode

Packet logger parameters:

- **-l** -Logger mode, target log and alert output directory. Default output folder is **/var/log/snort.** The default action is to dump as tcpdump format in **/var/log/snort**
- **-K ASCII**- Log packets in ASCII format.
- **-r** -Reading option, read the dumped logs in Snort.
- **-n** -Specify the number of packets that will process/read. Snort will stop after reading the specified number of packets.

Investigate the traffic with the default configuration file **with ASCII mode.**

```
sudo snort -dev -K ASCII -l .
```

Execute the traffic generator script and choose **“TASK-6 Exercise”**. Wait until the traffic ends, then stop the Snort instance. Now analyse the output summary and answer the question.

```
sudo ./traffic-generator.sh
```

Now, you should have the logs in the current directory. Navigate to folder "**145.254.160.237**". What is the source port used to connect port 53?

Answer: 3009

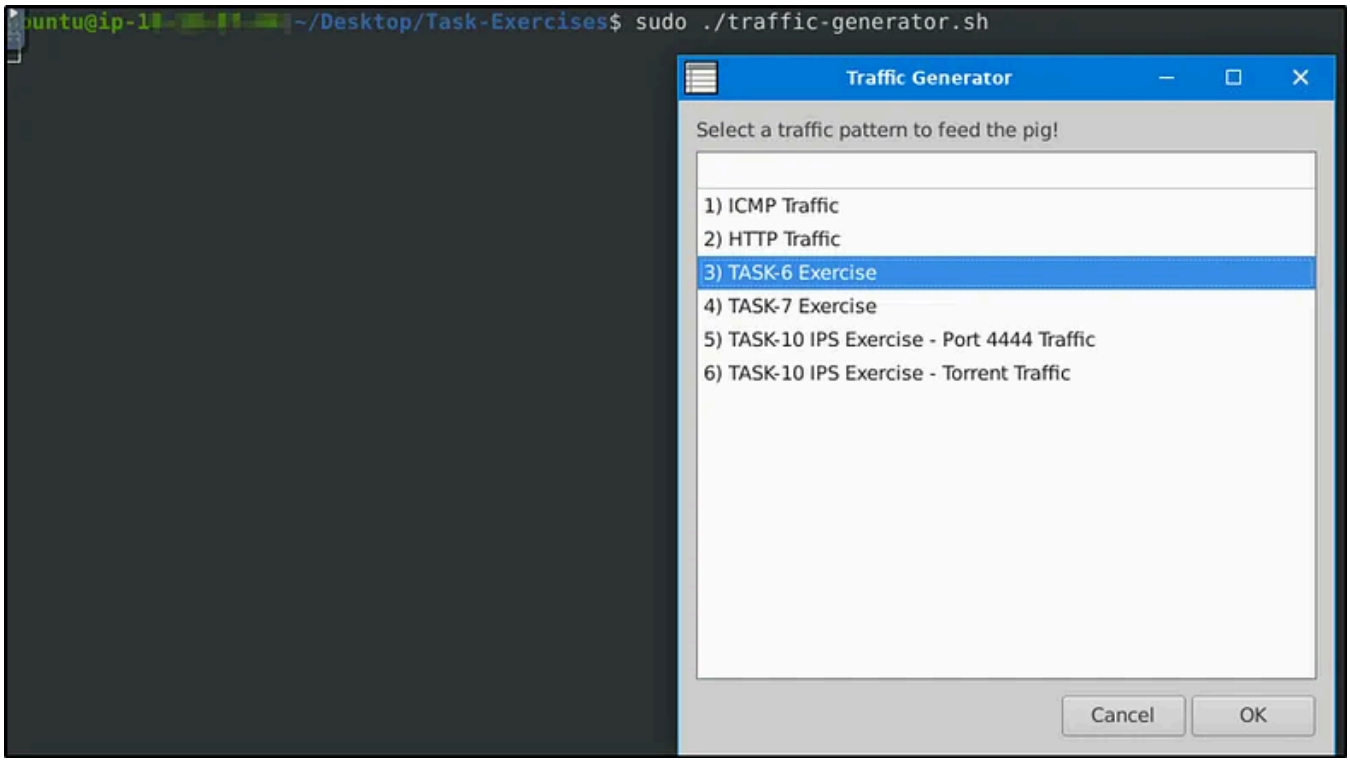
Run first snort in logger mode.

```
sudo snort -dev -K ASCII -l .
```

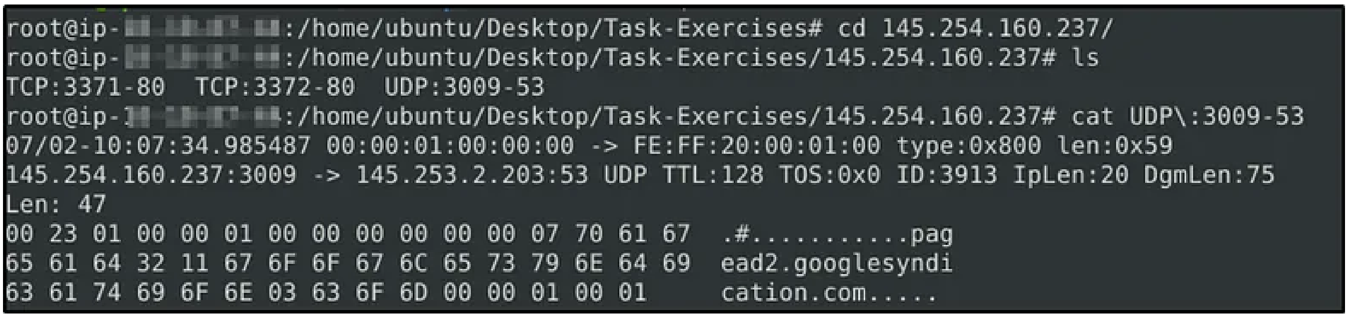
Let's run the traffic generator script.

```
sudo ./traffic-generator.sh
```

We are going to select task #3.



Let's cd to the folder created. We see 3 log files were created, which also denotes the port numbers the machine used in the traffic generated

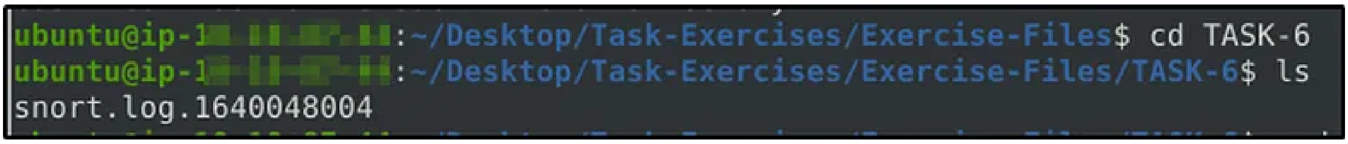


Use **snort.log.1640048004**

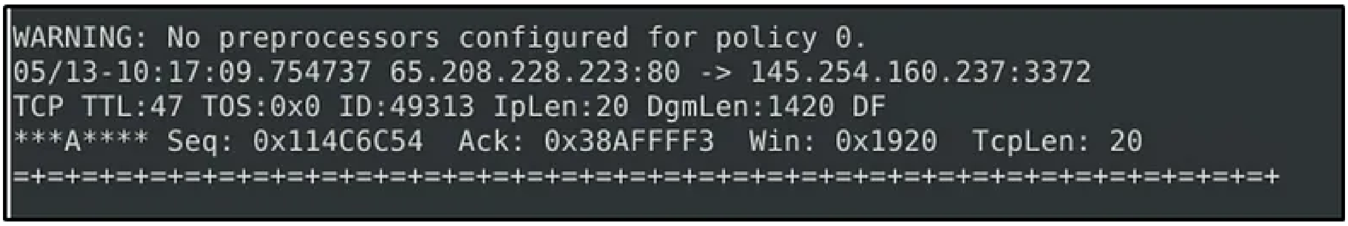
Read the snort.log file with Snort; what is the IP ID of the 10th packet?

Answer: 49313

The log file created should be in the current directory.




```
snort -r snort.log.1640048004 -n 10
```



Read the "**snort.log.1640048004**" file with Snort; what is the referer of the 4th packet?

Answer: http://www.ethereal.com/development.html

Add "-X" to display results in ASCII format.

```
sudo snort -Xr snort.log.1640048004  -n 4
```

```
ubuntu@ip [redacted]:~/Desktop/Task-Exercises/Exercise-Files/TASK-6$ sudo snort -Xr snort.log.1640048004  -n 4
Exiting after 4 packets
Running in packet dump mode
```

Right down at the bottom is the Referer.

```
WARNING: No preprocessors configured for policy 0.
05/13-10:17:08.222534 145.254.160.237:3372 -> 65.208.228.223:80
TCP TTL:128 TOS:0x0 ID:3909 IpLen:20 DgmLen:519 DF
***AP*** Seq: 0x38AFFE14  Ack: 0x114C618C  Win: 0x25BC  TcpLen: 20
0x0000: FE FF 20 00 01 00 00 00 01 00 00 00 08 00 45 00  .. ..........E.
0x0010: 02 07 0F 45 40 00 80 06 90 10 91 FE A0 ED 41 D0  ...E@.........A.
0x0020: E4 DF 0D 2C 00 50 38 AF FE 14 11 4C 61 8C 50 18  ...,.P8....La.P.
0x0030: 25 BC A9 58 00 00 47 45 54 20 2F 64 6F 77 6E 6C  %..X..GET /downl
0x0040: 6F 61 64 2E 68 74 6D 6C 20 48 54 54 50 2F 31 2E  oad.html HTTP/1.
0x0050: 31 0D 0A 48 6F 73 74 3A 20 77 77 77 2E 65 74 68  1..Host: www.eth
0x0060: 65 72 65 61 6C 2E 63 6F 6D 0D 0A 55 73 65 72 2D  ereal.com..User-
0x0070: 41 67 65 6E 74 3A 20 4D 6F 7A 69 6C 6C 61 2F 35  Agent: Mozilla/5
0x0080: 2E 30 20 28 57 69 6E 64 6F 77 73 3B 20 55 3B 20  .0 (Windows; U;
0x0090: 57 69 6E 64 6F 77 73 20 4E 54 20 35 2E 31 3B 20  Windows NT 5.1;
0x00A0: 65 6E 2D 55 53 3B 20 72 76 3A 31 2E 36 29 20 47  en-US; rv:1.6) G
0x00B0: 65 63 6B 6F 2F 32 30 30 34 30 31 31 33 0D 0A 41  ecko/20040113..A
0x00C0: 63 63 65 70 74 3A 20 74 65 78 74 2F 78 6D 6C 2C  ccept: text/xml,
0x00D0: 61 70 70 6C 69 63 61 74 69 6F 6E 2F 78 6D 6C 2C  application/xml,
0x00E0: 61 70 70 6C 69 63 61 74 69 6F 6E 2F 78 68 74 6D  application/xhtm
0x00F0: 6C 2B 78 6D 6C 2C 74 65 78 74 2F 68 74 6D 6C 3B  l+xml,text/html;
0x0100: 71 3D 30 2E 39 2C 74 65 78 74 2F 70 6C 61 69 6E  q=0.9,text/plain
0x0110: 3B 71 3D 30 2E 38 2C 69 6D 61 67 65 2F 70 6E 67  ;q=0.8,image/png
0x0120: 2C 69 6D 61 67 65 2F 6A 70 65 67 2C 69 6D 61 67  ,image/jpeg,imag
0x0130: 65 2F 67 69 66 3B 71 3D 30 2E 32 2C 2A 2F 2A 3B  e/gif;q=0.2,*/*;
0x0140: 71 3D 30 2E 31 0D 0A 41 63 63 65 70 74 2D 4C 61  q=0.1..Accept-La
0x0150: 6E 67 75 61 67 65 3A 20 65 6E 2D 75 73 2C 65 6E  nguage: en-us,en
0x0160: 3B 71 3D 30 2E 35 0D 0A 41 63 63 65 70 74 2D 45  ;q=0.5..Accept-E
0x0170: 6E 63 6F 64 69 6E 67 3A 20 67 7A 69 70 2C 64 65  ncoding: gzip,de
0x0180: 66 6C 61 74 65 0D 0A 41 63 63 65 70 74 2D 43 68  flate..Accept-Ch
0x0190: 61 72 73 65 74 3A 20 49 53 4F 2D 38 38 35 39 2D  arset: ISO-8859-
0x01A0: 31 2C 75 74 66 2D 38 3B 71 3D 30 2E 37 2C 2A 3B  1,utf-8;q=0.7,*;
0x01B0: 71 3D 30 2E 37 0D 0A 4B 65 65 70 2D 41 6C 69 76  q=0.7..Keep-Aliv
0x01C0: 65 3A 20 33 30 30 0D 0A 43 6F 6E 6E 65 63 74 69  e: 300..Connecti
0x01D0: 6F 6E 3A 20 6B 65 65 70 2D 61 6C 69 76 65 0D 0A  on: keep-alive..
0x01E0: 52 65 66 65 72 65 72 3A 20 68 74 74 70 3A 2F 2F  Referer: http://
0x01F0: 77 77 77 2E 65 74 68 65 72 65 61 6C 2E 63 6F 6D  www.ethereal.com
0x0200: 2F 64 65 76 65 6C 6F 70 6D 65 6E 74 2E 68 74 6D  /development.htm
0x0210: 6C 0D 0A 0D 0A                                   l....
```

Read the “**snort.log.1640048004**” file with Snort; what is the Ack number of the 8th packet?

Answer: 0x38AFFFF3

```
sudo snort -r snort.log.1640048004 -n 8
```

Note to read the 8th packet of the results.

```
WARNING: No preprocessors configured for policy 0.
05/13-10:17:09.123830 65.208.228.223:80 -> 145.254.160.237:3372
TCP TTL:47 TOS:0x0 ID:49312 IpLen:20 DgmLen:1420 DF
***A**** Seq: 0x114C66F0  Ack: 0x38AFFFF3  Win: 0x1920  TcpLen: 20
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
```

Read the "**snort.log.1640048004**" file with Snort; what is the number of the **"TCP port 80"** packets?

Answer: 41

For this task, we will be utilizing "BPF". According to Wikipedia, "The **Berkeley Packet Filter** (**BPF**) is a technology used in certain computer operating systems for programs that need to, among other things, analyze network traffic. It provides a raw interface to data link layers, permitting raw link-layer packets to be sent and received."

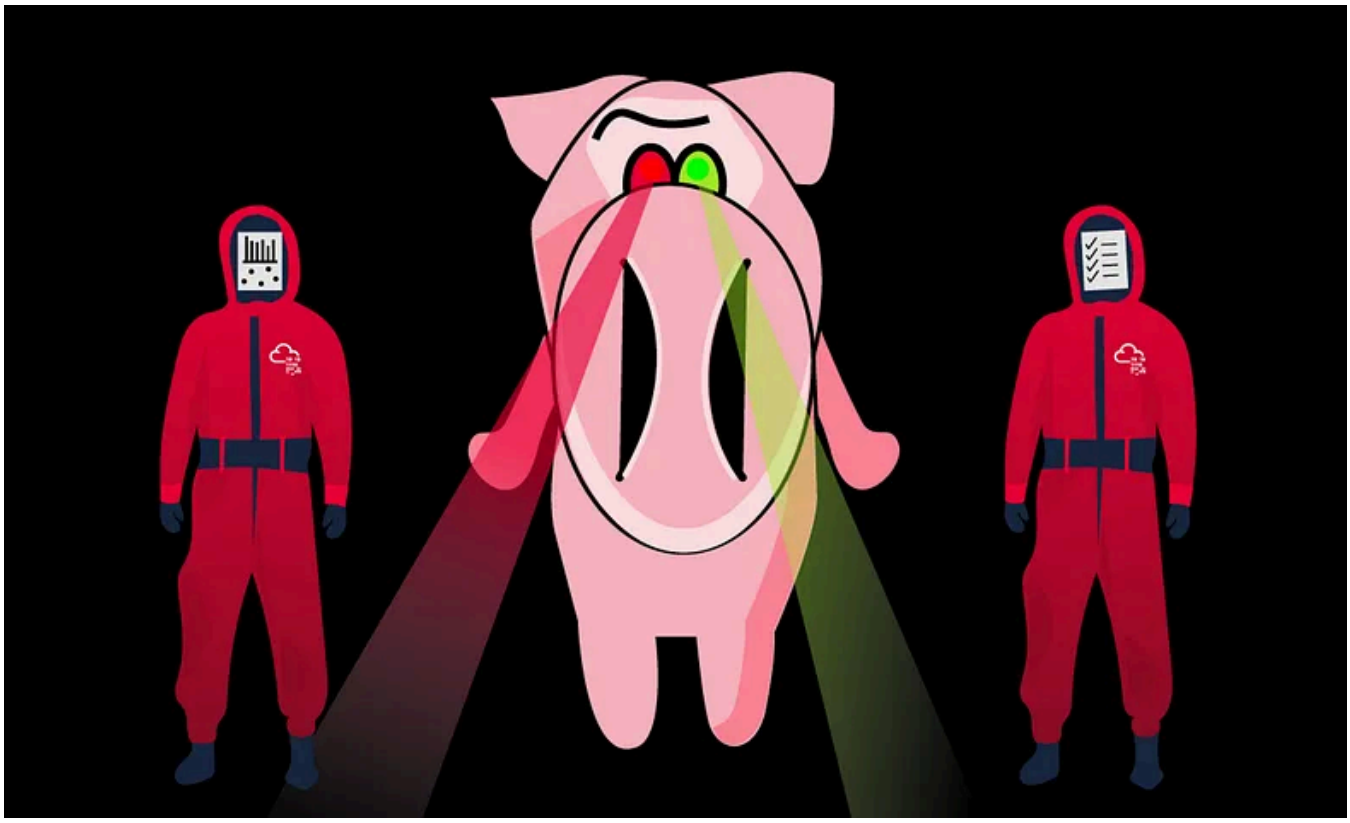
Check out the syntax for BPF here: https://biot.com/capstats/bpf.html

```
sudo snort -r snort.log.1640048004  'tcp port 80'
```

The result will only display traffic captured from port 80.

```
Packet I/O Totals:
   Received:           41
   Analyzed:           41 (100.000%)
    Dropped:            0 (  0.000%)
   Filtered:            0 (  0.000%)
Outstanding:            0 (  0.000%)
   Injected:            0
```

Task 7: Operation Mode 3: IDS/IPS



Note that (N)IDS/IPS mode depends on the rules and configuration. TASK-10 summarises the essential paths, files and variables. Also, TASK-3 covers configuration testing. Here, we need to understand the operating logic first, and then we will be going into rules in TASK-9

NIDS mode parameters:

- -c :Defining the configuration file.
- -T :Testing the configuration file.
- **-N :**Disable logging.
- **-D :**Background mode.
- **-A:** Alert modes;
- **full**: Full alert mode, providing all possible information about the alert. This one also is the default mode; once you use -A and don't specify any mode, snort uses this mode.

- **fast**: Fast mode shows the alert message, timestamp, source and destination IP, along with port numbers.
- **console**: Provides fast style alerts on the console screen.
- **cmg**: CMG style, basic header details with payload in hex and text format.
- **none**: Disabling alerting

Once you start running IDS/IPS mode, you need to use rules. We will use a pre-defined ICMP rule as an example. The defined rule will only generate alerts in any direction of ICMP packet activity.

```
alert icmp any any <> any any  (msg: "ICMP Packet Found"; sid: 100001; rev:1;)
```

IDS/IPS mode with the different parameters:

```
sudo snort -c /etc/snort/snort.conf -T
sudo snort -c /etc/snort/snort.conf -N
sudo snort -c /etc/snort/snort.conf -D
sudo snort -c /etc/snort/snort.conf -D -X -l .
sudo snort -c /etc/snort/snort.conf -A console
sudo snort -c /etc/snort/snort.conf -A cmg
sudo snort -c /etc/snort/snort.conf -A fast
sudo snort -c /etc/snort/snort.conf -A full
sudo snort -c /etc/snort/snort.conf -A none
```

With parameter "-D", we can activate **verbosity (-v)** or **full packet dump (-X)** with **packet logger mode (-l)** and we will still have the logs in the logs folder, but there will be no output in the console.

Once you start the background mode and want to check the corresponding process, you can easily use the "ps" command as shown below;

```
ps -ef | grep snort
```

If you want to stop the daemon, you can easily use the “kill” command to stop the process.

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

Using rule file without configuration file

```
sudo snort -c /etc/snort/rules/local.rules -A console
```

IPS mode and dropping packets

Snort IPS mode activated with **-Q — daq afpacket** parameters. You can also activate this mode by editing snort.conf file.

Activate the Data Acquisition (DAQ) modules and use the afpacket module to use snort as an IPS: **-i eth0:eth1**

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

Investigate the traffic with the default configuration file.

```
sudo snort -c /etc/snort/snort.conf -A full -l .
```

Execute the traffic generator script and choose **“TASK-7 Exercise”**. Wait until the traffic stops, then stop the Snort instance. Now analyse the output summary and

answer the question.

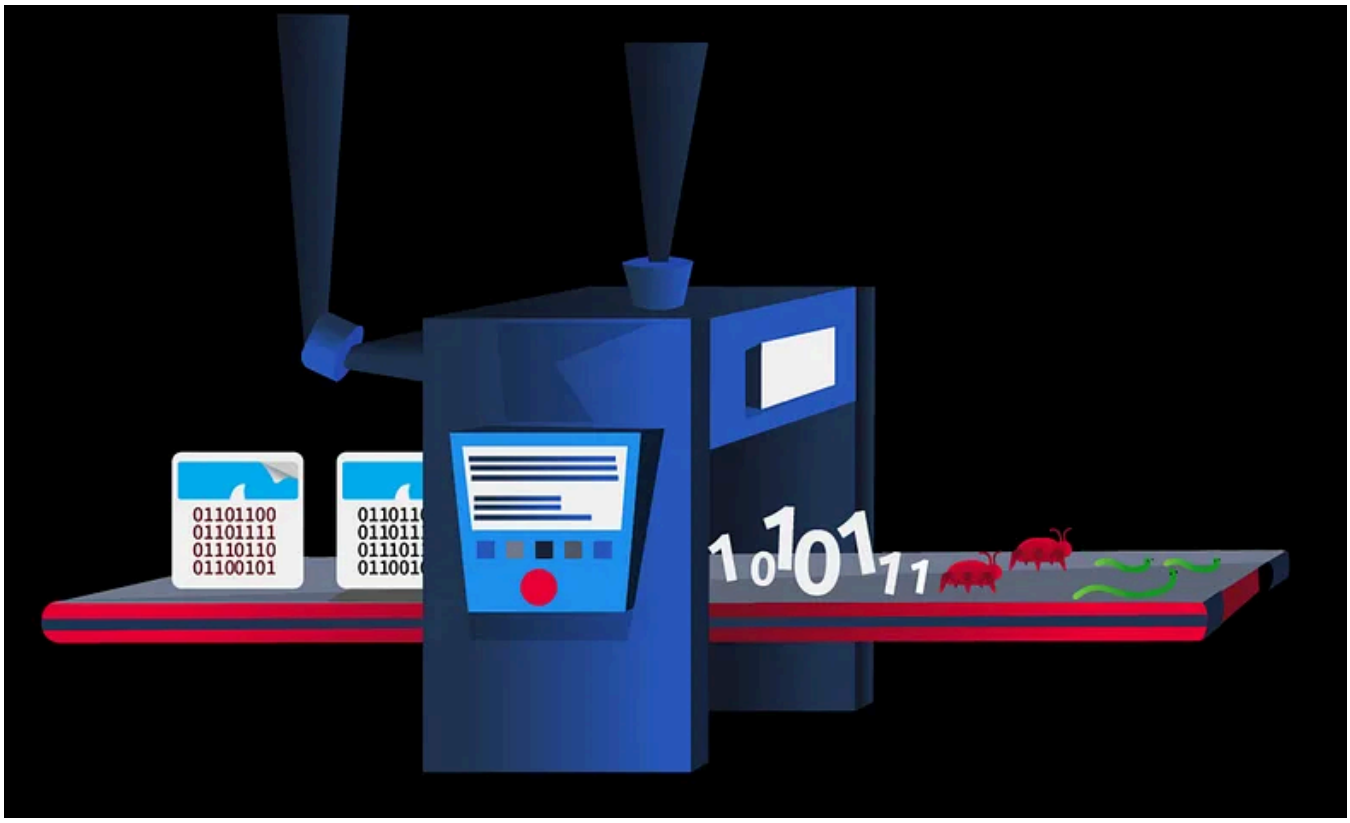
```
sudo ./traffic-generator.sh
```

What is the number of the detected HTTP GET methods?

Answer: 2

```
HTTP Inspect - encodings (Note: stream-reassembled packets included):
    POST methods:                         0
    GET methods:                          2
    HTTP Request Headers extracted:       2
    HTTP Request Cookies extracted:       0
    Post parameters extracted:            0
    HTTP response Headers extracted:      3
    HTTP Response Cookies extracted:      0
    Unicode:                              0
    Double unicode:                       0
    Non-ASCII representable:              0
    Directory traversals:                 0
    Extra slashes ("//"):                 1
    Self-referencing paths ("./"):        0
    HTTP Response Gzip packets extracted: 1
    Gzip Compressed Data Processed:       1272.00
    Gzip Decompressed Data Processed:     3608.00
    Total packets processed:              2142
```

Task 8: Operation Mode 4: PCAP Investigation



Capabilities of Snort are not limited to sniffing, logging and detecting/preventing the threats. PCAP read/investigate mode helps us work with pcap files. Once we have a pcap file and process it with Snort, we will receive default traffic statistics with alerts depending on our rule set.

PCAP mode parameters:

- **- r / — pcap-single= :**Read a single pcap
- **— pcap-list="" :**Read pcaps provided in command (space separated).
- **— pcap-show :**Show pcap name on console during processing.

Investigating single pcap file with a configuration file.

```
sudo snort -c /etc/snort/snort.conf -q -r icmp-test.pcap -A console -n 10
```

Investigating multiple PCAPs with parameter " — pcap-list"

```
sudo snort -c /etc/snort/snort.conf -q --pcap-list="icmp-test.pcap http2.pcap"
```

Investigating multiple PCAPs with parameter " — pcap-show"

Snort will identify the traffic, distinguish each pcap file and prompts the alerts according to our ruleset.


```
sudo snort -c /etc/snort/snort.conf -q --pcap-list="icmp-test.pcap http2.pcap"
```

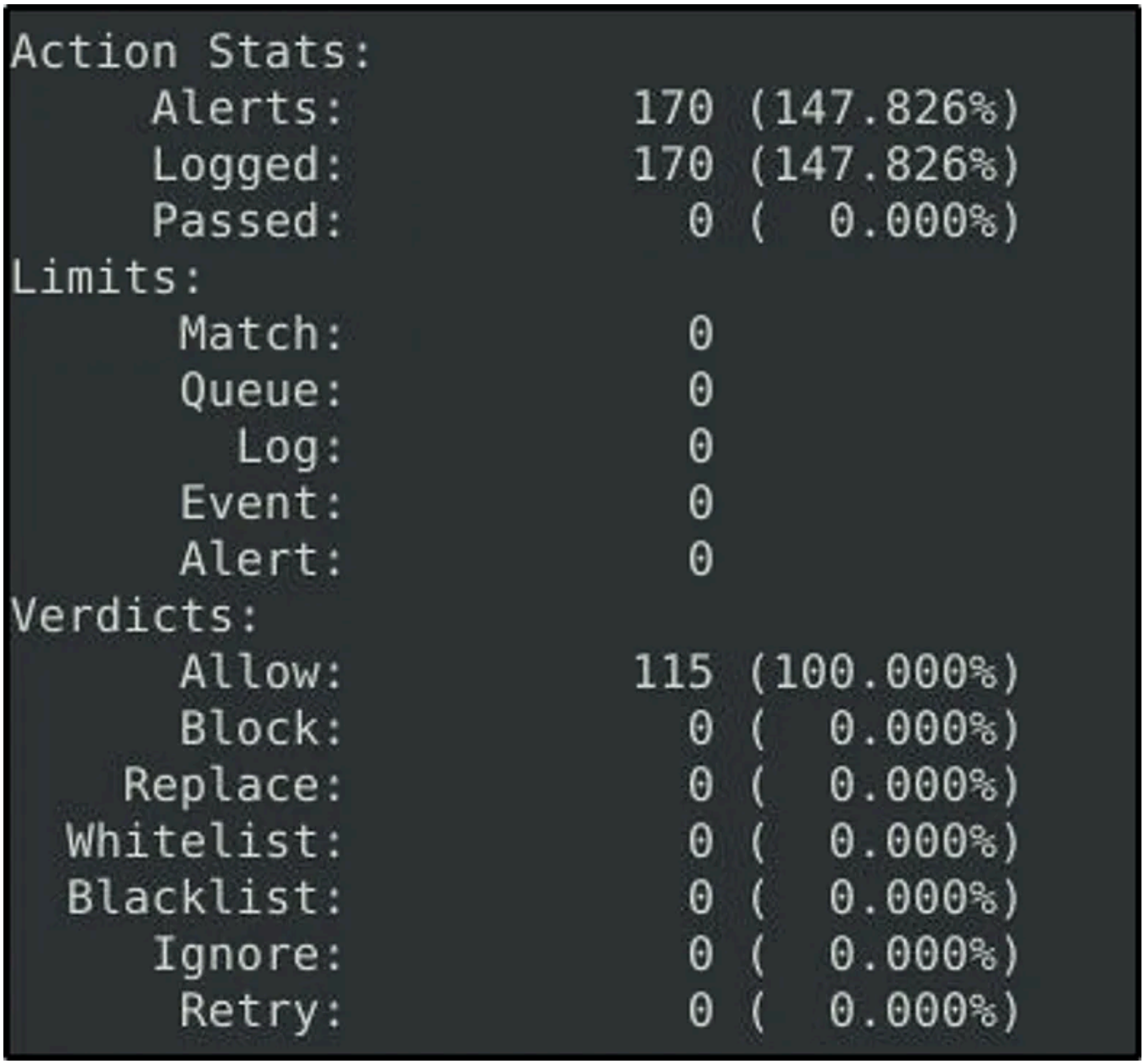

Answer the questions below

Investigate the mx-1.pcap file with the default configuration file.

What is the number of the generated alerts?

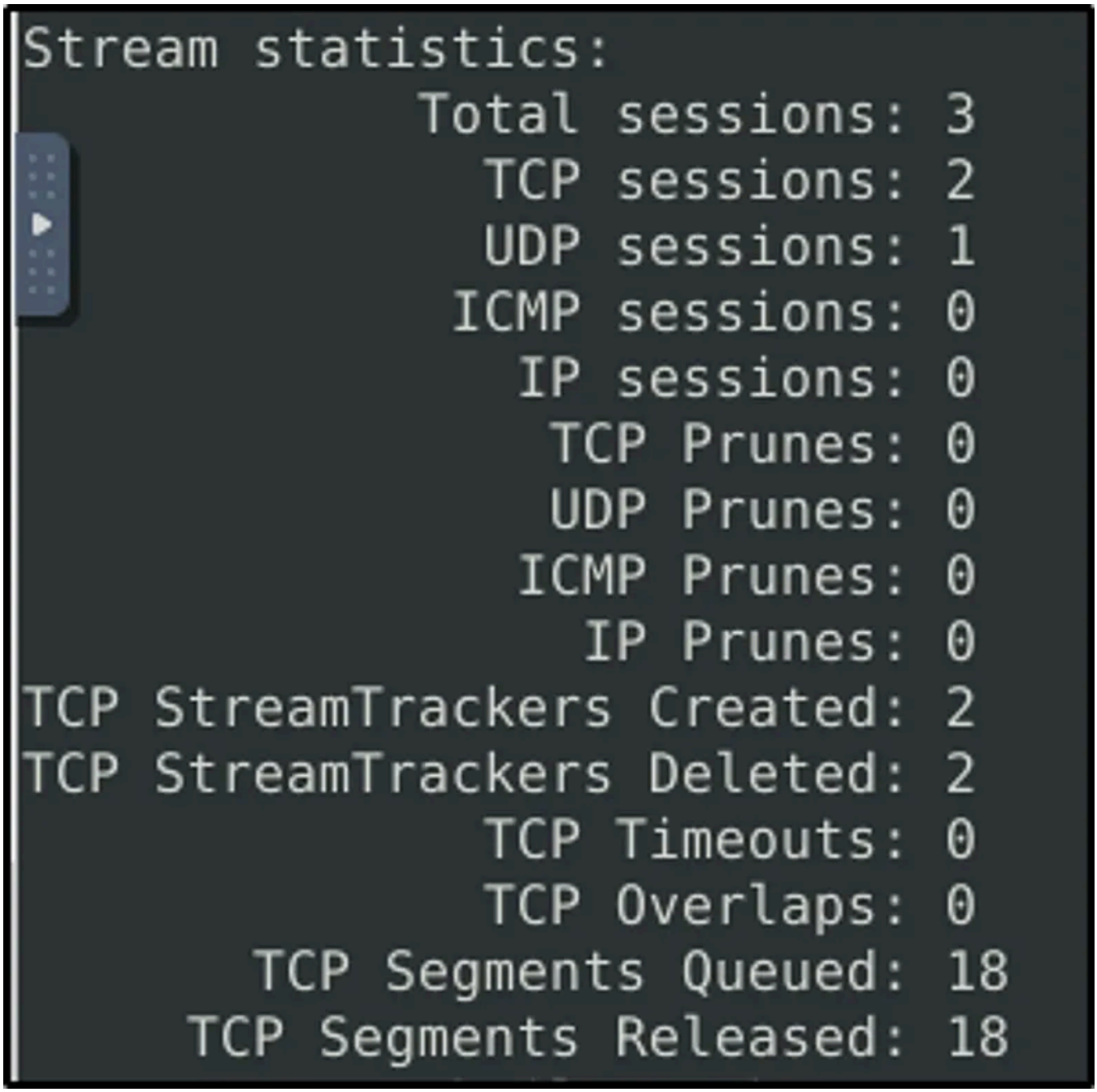
Answer: 170

```
sudo snort -c /etc/snort/snort.conf -A full -l . -r mx-1.pcap
```



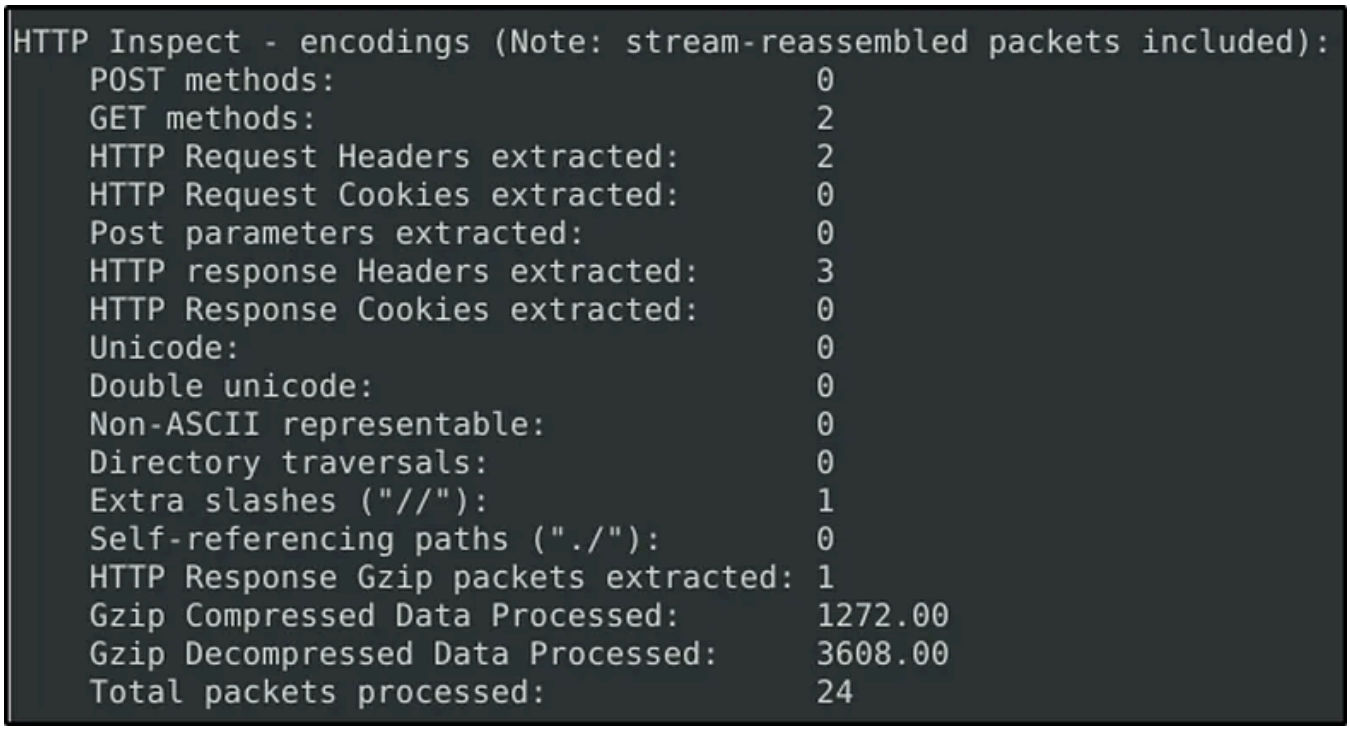
Keep reading the output. How many TCP Segments are Queued?

Answer: 18



Keep reading the output.How many “HTTP response headers” were extracted?

Answer: 3

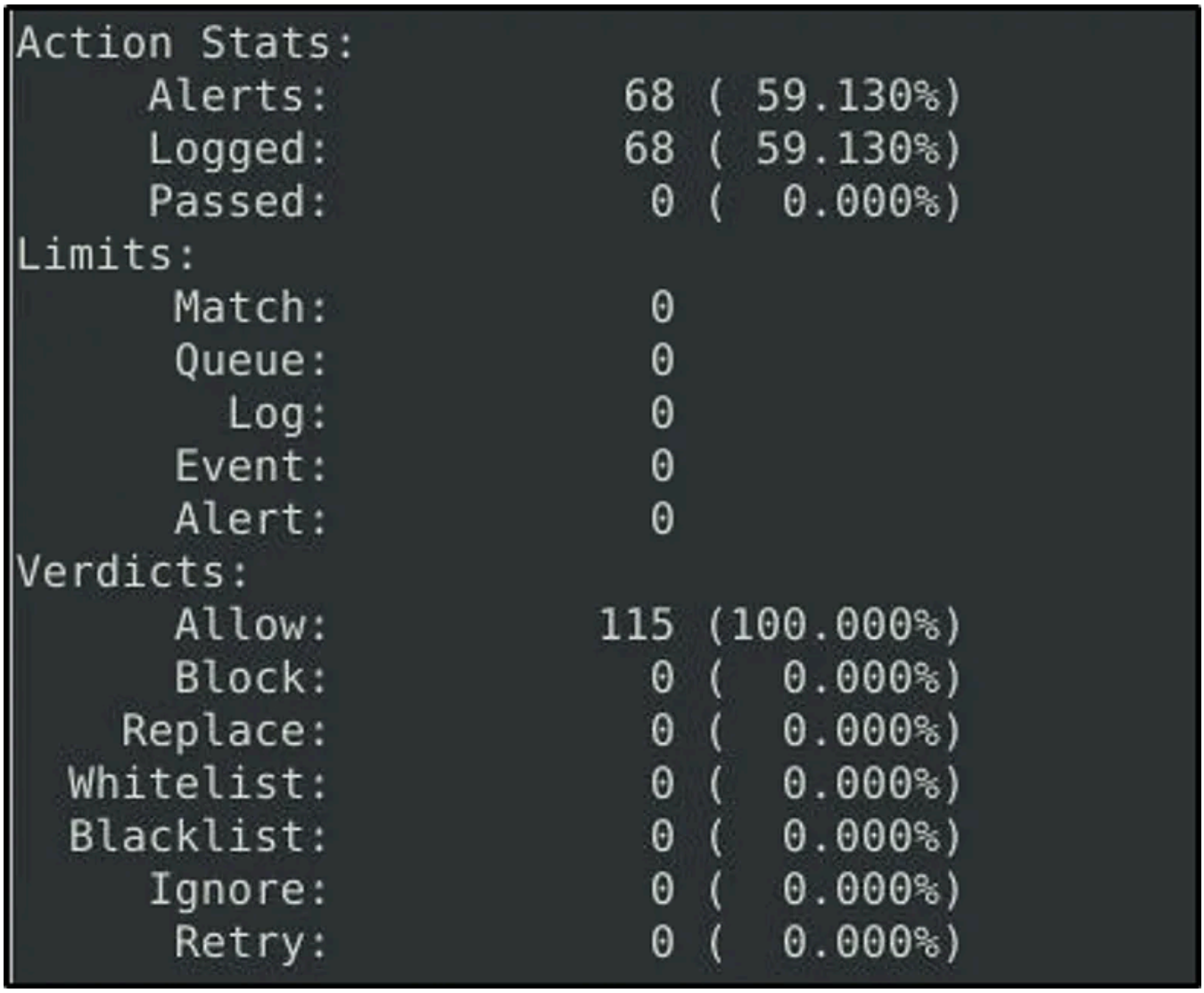


Investigate the mx-1.pcap file with the second configuration file.

```
sudo snort -c /etc/snort/snortv2.conf -A full -l . -r mx-1.pcap
```

What is the number of the generated alerts?

Answer: 68

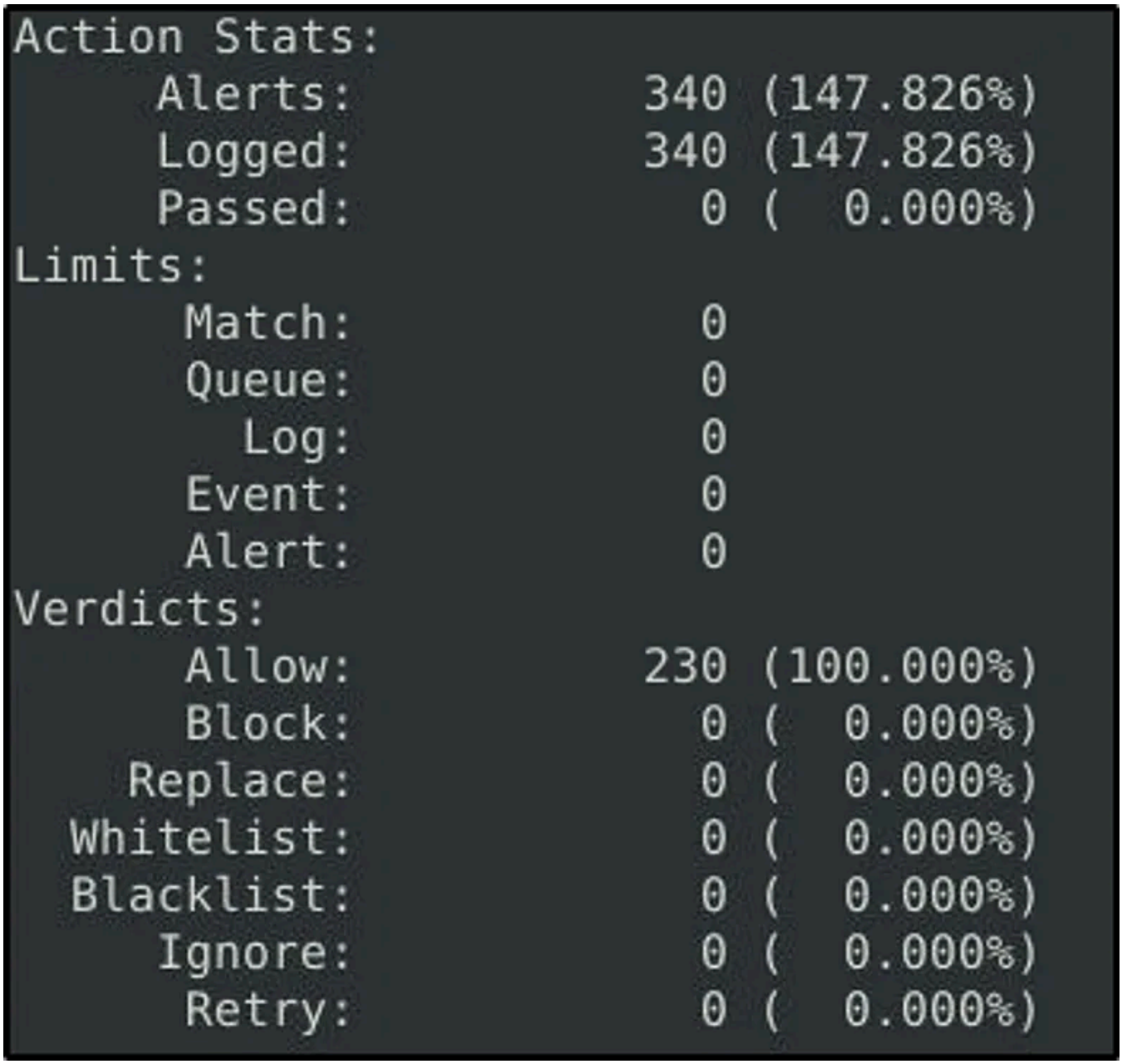


Investigate the mx-2.pcap file with the default configuration file.

```
sudo snort -c /etc/snort/snort.conf -A full -l . -r mx-2.pcap
```

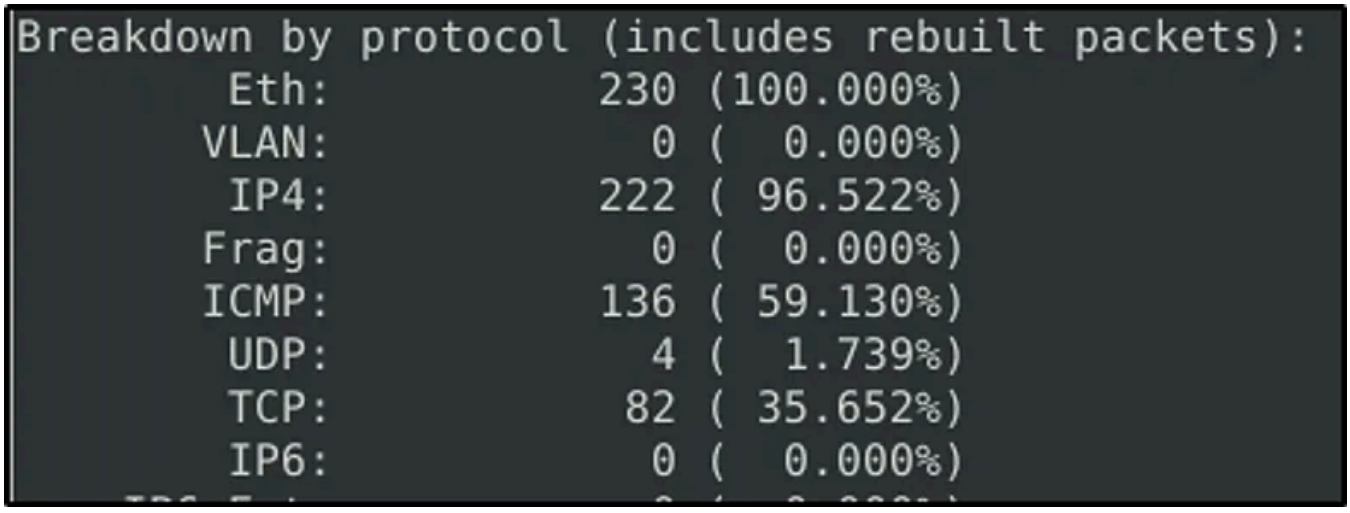
What is the number of the generated alerts?

Answer: 340



Keep reading the output. What is the number of the detected TCP packets?

Answer: 82

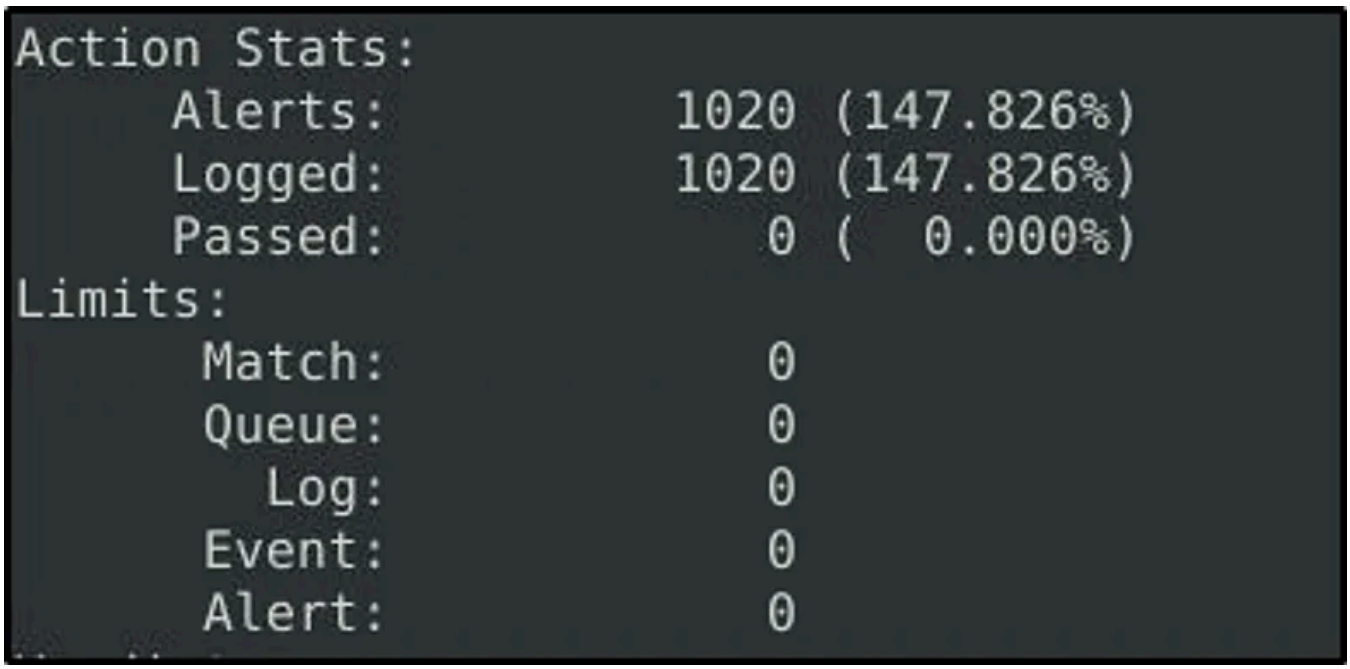


Investigate the mx-2.pcap and mx-3.pcap files with the default configuration file.

```
sudo snort -c /etc/snort/snort.conf -A full -l . --pcap-list="mx-2.pcap mx-3.pc
```

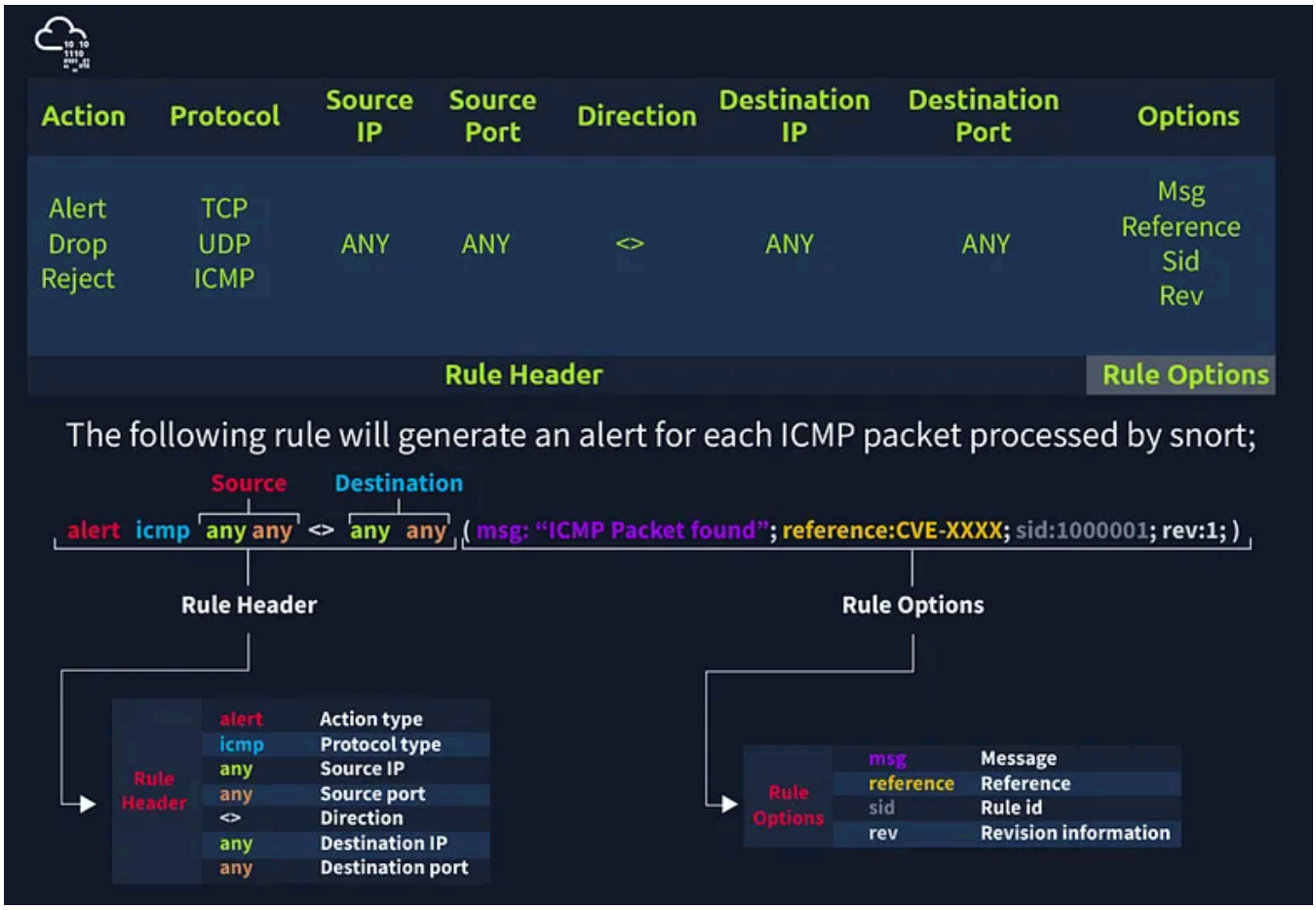
What is the number of the generated alerts?

Answer: 1020



Task 9: Snort Rule Structure

Understanding the Snort rule format is essential for any blue and purple teams. The primary structure of the snort rule is shown below;



Each rule should have a type of **action**, **protocol**, **source** and **destination IP**, **source** and **destination port** and an **option.** Snort is in passive mode by default. So most of the time, we will use Snort as an IDS. We will need to start "inline mode" to turn on IPS mode. But before we start playing with inline mode, let' s familiarize ourselves with Snort features and rules.

The **Rule Header** is **required** and a rule will not work without it. Meanwhile, the **Rule Options** are "**optional**". But we will see later that some attacks are detected by using the rule options.

Action: The most common actions are listed below.

- **alert:** Generate an alert and log the packet.
- **log:** Log the packet.
- **drop:** Block and log the packet.
- **reject:** Block the packet, log it and terminate the packet session.

Protocol : Snort2 supports only four protocols filters in the rules (IP, TCP, UDP and ICMP). However, we can detect the application (like ftp, or ssh) flows using port

numbers (21, 22, 23, etc)and options by investigating TCP traffic.

IP and Port Numbers

- IP Filtering: ***alert icmp 192.168.1.56 any <> any any (msg: "ICMP Packet Found"; sid: 100001; rev:1;)*** This rule will create alerts for each ICMP packet originating from the 192.168.1.56 IP address.
- Filter an IP range: ***alert icmp 192.168.1.0/24 any <> any any (msg: "ICMP Packet Found"; sid: 100001; rev:1;)*** This rule will create alerts for each ICMP packet originating from the 192.168.1.0/24 subnet.
- Filter multiple IP ranges: ***alert icmp [192.168.1.0/24, 10.1.1.0/24] any <> any any (msg: "ICMP Packet Found"; sid: 100001; rev:1;)*** This rule will create alerts for each ICMP packet originating from the 192.168.1.0/24 and 10.1.1.0/24 subnets.
- Exclude IP addresses/ranges: "negation operator" is used for excluding specific addresses and ports. Negation operator is indicated with "!" ***alert icmp !192.168.1.0/24 any <> any any (msg: "ICMP Packet Found"; sid: 100001; rev:1;)*** This rule will create alerts for each ICMP packet not originating from the 192.168.1.0/24 subnet.
- Port Filtering: ***alert tcp !192.168.1.0/24 21 <> any any (msg: "ICMP Packet Found"; sid: 100001; rev:1;)*** This rule will create alerts for each TCP packet originating from port 21.
- Exclude a specific port: ***alert tcp !192.168.1.0/24 !21 <> any any (msg: "ICMP Packet Found"; sid: 100001; rev:1;)*** This rule will create alerts for each TCP packet not originating from port 21.
- Filter a port range (Type 1): ***alert tcp !192.168.1.0/24 1:1024 <> any any (msg: "ICMP Packet Found"; sid: 100001; rev:1;)*** This rule will create alerts for each TCP packet originating from ports between 1–1024.
- Filter a port range (Type 2): ***alert icmp any :1024 <> any any (msg: "ICMP Packet Found"; sid: 100001; rev:1;)*** This rule will create alerts for each TCP packet originating from ports less than or equal to 1024.
- Filter a port range (Type 3): ***alert icmp any 1024: <> any any (msg: "ICMP Packet Found"; sid: 100001; rev:1;)*** This rule will create alerts for each TCP packet originating from a source port higher than or equal to 1024.

- Filter a port range (Type 4): ***alert icmp any 80,1024: <> any any (msg: "ICMP Packet Found"; sid: 100001; rev:1;)*** This rule will create alerts for each TCP packet originating from a source port 80 and higher than or equal to 1024.

Direction

The direction operator indicates the traffic flow to be filtered by Snort. The left side of the rule shows the source, and the right side shows the destination.

- > Source to destination flow.
- <> Bidirectional flow

Note that there is **no "<-"** operator in Snort.

Three main rule options in Snort;

- General Rule Options — Fundamental rule options for Snort.
- Payload Rule Options — Rule options that help to investigate the payload data. These options are helpful to detect specific payload patterns.
- Non-Payload Rule Options — Rule options that focus on non-payload data. These options will help create specific patterns and identify network issues.

General Rule Options

- **Msg :** Basic prompt and quick identifier of the rule. Once the rule is triggered, the message filed will appear in the console or log. Usually, the message part is a one-liner that summarises the event.
- **Sid :** Snort rule IDs (SID) come with a pre-defined scope, and each rule must have a SID in a proper format. There are three different scopes for SIDs shown below.
- <100: Reserved rules
- 100–999,999: Rules came with the build.
- >=1,000,000: Rules created by user. Briefly, the rules we will create should have sid greater than 100.000.000. Another important point is; SIDs should not overlap, and each id must be unique.

- **Reference :** Each rule can have additional information or reference to explain the purpose of the rule or threat pattern. That could be a Common Vulnerabilities and Exposures (CVE) id or external information. Having references for the rules will always help analysts during the alert and incident investigation.

- **Rev :** Refers to the revision number for the rule as rules can be modified and updated for performance and efficiency issues. It only indicates the number of times a rule has been modified. Rules have no history backup, so analyst should keep the history themselves. Each rule should have a unique rev number. *alert icmp any any <> any any (msg: "ICMP Packet Found"; sid: 100001; reference:cve,CVE-XXXX; rev:1;)*

Payload Detection Rule Options

Content — Payload data. It matches specific payload data by ASCII, HEX or both. It is possible to use this option multiple times in a single rule. However, the more you create specific pattern match features, the more it takes time to investigate a packet. Following rules will create an alert for each HTTP packet containing the keyword "GET". This rule option is case sensitive!

- ASCII mode — alert tcp any any <> any 80 (msg: "GET Request Found"; content:"GET"; sid: 100001; rev:1;)

- HEX mode — alert tcp any any <> any 80 (msg: "GET Request Found"; content:"|47 45 54|"; sid: 100001; rev:1;)

Nocase — Disabling case sensitivity. Used for enhancing the content searches.

- alert tcp any any <> any 80 (msg: "GET Request Found"; content:"GET"; nocase; sid: 100001; rev:1;)

Fast_pattern — Prioritise content search to speed up the payload search operation. By default, Snort uses the biggest content and evaluates it against the rules. "fast_pattern" option helps you select the initial packet match with the specific value for further investigation. This option always works case insensitive and can be used once per rule. Note that this option is required when using multiple "content" options. The following rule has two content options, and the fast_pattern option tells to snort to use the first content option (in this case, "GET") for the initial packet match.

- alert tcp any any <> any 80 (msg: “GET Request Found”; content:“GET”; fast_pattern; content:“www”; sid:100001; rev:1;)

Non-Payload Detection Rule Options

ID — Filtering the IP id field.

- alert tcp any any <> any any (msg: “ID TEST”; id:123456; sid: 100001; rev:1;)

Flags — Filtering the TCP flags.

- F — FIN
- S — SYN
- R — RST
- P — PSH
- A — ACK
- U — URG
- alert tcp any any <> any any (msg: “FLAG TEST”; flags:S; sid: 100001; rev:1;)

Dsize — Filtering the packet payload size.

- dsize:min<>max;
- dsize:>100
- dsize:<100
- alert ip any any <> any any (msg: “SEQ TEST”; dsize:100<>300; sid: 100001; rev:1;)

Sameip — Filtering the source and destination IP addresses for duplication.

- alert ip any any <> any any (msg: “SAME-IP TEST”; sameip; sid: 100001; rev:1;)

Remember, once you create a rule, it is a local rule and should be in your “local.rules” file. This file is located under “**/etc/snort/rules/local.rules**”. A quick reminder on how to edit your local rules is shown below.

```
sudo gedit /etc/snort/rules/local.rules
```

In this task, the default Snort rules have been deactivated and the location of rule to be applied is in the current working directory.

Use the attached VM and navigate to the Task-Exercises/Exercise-Files/TASK-9 folder to answer the questions! Note that you can use the following command to create the logs in the current directory: -l .

Use **“task9.pcap”.**

Write a rule to filter **IP ID “35369”** and run it against the given pcap file. What is the request name of the detected packet?

```
sudo snort -c local.rules -A full -l . -r task9.pcap
```

Answer: TIMESTAMP REQUEST

Before we run the command, we need to edit the rule to filter IP ID “35369”. Refer to the section above for Non-Payload Detection Rule Options. We will create only one rule.

```
sudo nano local.rules
```

- *alert tcp any any <> any any (msg:”ID Test”;id:35369;sid:10000000001; rev:1;)*

```
# ----------------
# LOCAL RULES
# ----------------
# This file intentionally does not come with signatures.  Put your local
# additions here.
alert ip any any <> any any (msg:"ID Test";id:35369;sid:10000000001; rev:1;)
```

Let's now run Snort. Observe that it read only the rule we have applied.

```
ubuntu@ip-1[redacted]:~/Desktop/Task-Exercises/Exercise-Files/TASK-9$ sudo snort -c local.rules -A full -l . -r task9.pcap
Running in IDS mode

        --== Initializing Snort ==--
Initializing Output Plugins!
Initializing Preprocessors!
Initializing Plug-ins!
Parsing Rules file "local.rules"
Tagged Packet Limit: 256
Log directory = .

+++++++++++++++++++++++++++++++++++++++++++++++++
Initializing rule chains...
1 Snort rules read
    1 detection rules
    0 decoder rules
    0 preprocessor rules
1 Option Chains linked into 1 Chain Headers
0 Dynamic rules
```

We read the alert file and we see the request name.

```
ubuntu@ip[redacted]:~/Desktop/Task-Exercises/Exercise-Files/TASK-9$ cat alert
[**] [1:1410065409:1] ID Test [**]
[Priority: 0]
03/03-20:00:32.042975 192.168.121.2 -> 192.168.120.1
ICMP TTL:255 TOS:0x0 ID:35369 IpLen:20 DgmLen:40
Type:13  Code:0  ID: 7  Seq: 6  TIMESTAMP REQUEST
```

Clear the previous log and alarm files and deactivate/comment out the old rule

Create a rule to filter **packets with Syn flag** and run it against the given pcap file. What is the number of detected packets?

Answer: 1

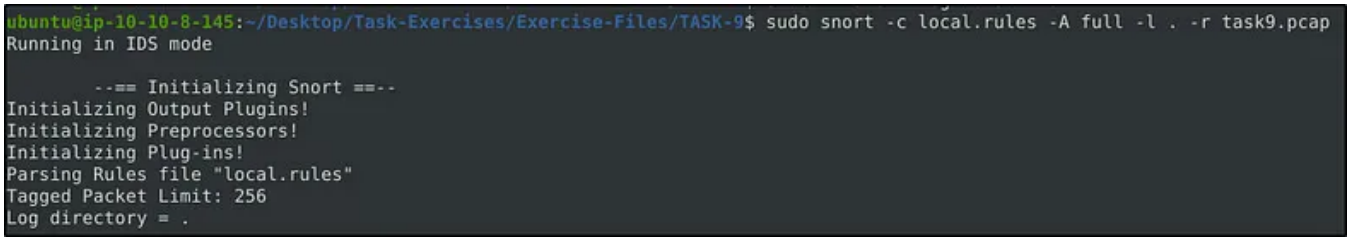
Again, refer to the Non-Payload Detection Rule Options. We will include the Option "flags" with a value of "S" to detect SYN flags.

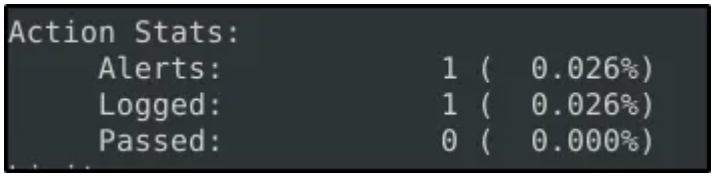
- alert tcp any any <> any any (msg:"FLAG TEST";flags:S;sid:10000000002; rev:1)

```
# ---------------
# LOCAL RULES
# ---------------
# This file intentionally does not come with signatures.  Put your local
# additions here.
#alert ip any any <> any any (msg:"ID Test";id:35369;sid:10000000001; rev:1;)
alert tcp any any <> any any (msg:"FLAG TEST";flags:S;sid:10000000002; rev:1)
```

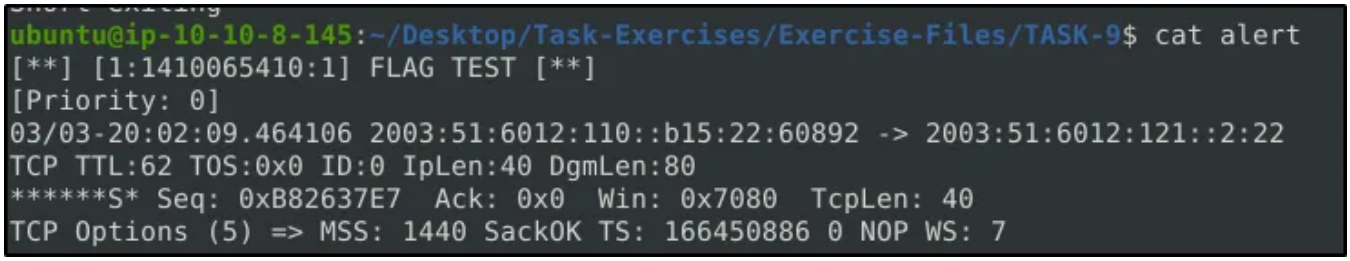
Let's run Snort.

```
sudo snort -c local.rules -A full -l . -r task9.pcap
```





The alert file would confirm that only one packet was detected.



Clear the previous log and alarm files and deactivate/comment out the old rule.

Write a rule to filter **packets with Push-Ack flags** and run it against the given pcap file. What is the number of detected packets?

Answer: 216

We just need to change the value of the option “flags” to “PA” to detect Push-Ack flags.

- alert tcp any any <> any any (msg:”Push-Ack FLAG TEST”;flags:PA;sid:10000000003; rev:1)

```
# ---------------
# LOCAL RULES
# ---------------
# This file intentionally does not come with signatures.  Put your local
# additions here.
#alert ip any any <> any any (msg:"ID Test";id:35369;sid:10000000001; rev:1;)
#alert tcp any any <> any any (msg:"FLAG TEST";flags:S;sid:10000000002; rev:1)
alert tcp any any <> any any (msg:"FLAG TEST";flags:PA;sid:10000000003; rev:1)
```

Run Snort. Modified a bit with “-q” so it won’t display results in the screen.

```
sudo snort -c local.rules -A full -l -q . -r task9.pcap
```

Again, there are ways on how to determine the detected flags. One, is by reading from the log file created.

```
sudo snort -r snort.log.1689840434
```

```
===================================================
Packet I/O Totals:
   Received:          216
   Analyzed:          216 (100.000%)
    Dropped:            0 (  0.000%)
   Filtered:            0 (  0.000%)
Outstanding:            0 (  0.000%)
   Injected:            0
===================================================
```

Or from the alert file that was created. We will concatenate the file, then grep some of the keywords we used in the option, and then count the results by line.

```
cat alert | grep "Push-Ack" | wc -l
```

```
ubuntu@ip-[redacted]:~/Desktop/Task-Exercises/Exercise-Files/TASK-9$ cat alert | grep "Push-Ack" | wc -l
216
```

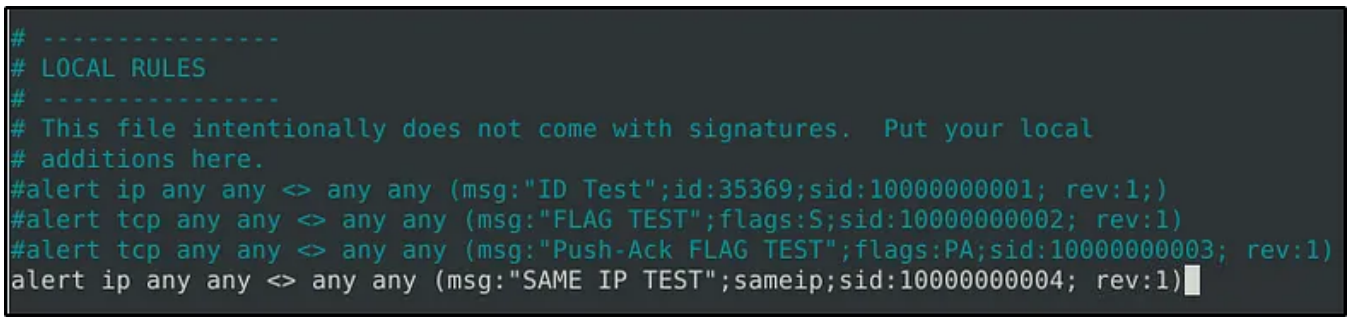
Clear the previous log and alarm files and deactivate/comment out the old rule.

Create a rule to filter **packets with the same source and destination IP** and run it against the given pcap file. What is the number of detected packets?

Answer: 10

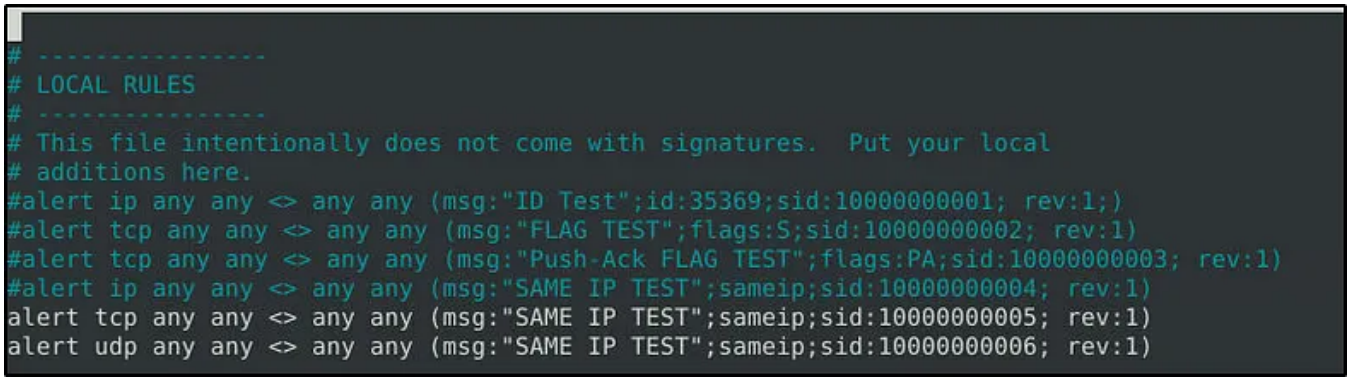
Refer on the Non-Payload Rule Options on SameIP. We will be using the option “sameip”.

```
alert ip any any <> any any (msg:"SAME IP TEST";sameip;sid:10000000004; rev:1)
```

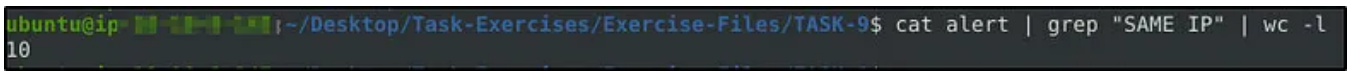


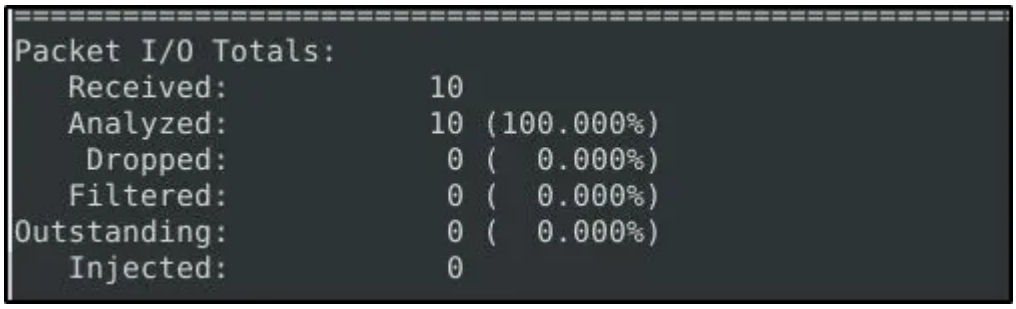
Run the command as above to start Snort detecting. Then look for the result.

Initially I got 13, but he hint says we need to filter TCP and UDP.



Run again Snort then read the alert or log file.





Case Example — An analyst modified an existing rule successfully. Which rule option must the analyst change after the implementation?

Answer: rev

As the rules are modified for performance and efficiency issues, “rev” number will change too.

Task 10: Snort2 Operation Logic: Points to Remember

Points to Remember

Main Components of Snort

- **Packet Decoder** — Packet collector component of Snort. It collects and prepares the packets for pre-processing.
- **Pre-processors** — A component that arranges and modifies the packets for the detection engine.
- **Detection Engine** — The primary component that process, dissect and analyse the packets by applying the rules.
- Logging and Alerting — Log and alert generation component.
- Outputs and Plugins — Output integration modules (i.e. alerts to syslog/mysql) and additional plugin (rule management detection plugins) support is done with this component.

There are three types of rules available for snort

- Community Rules — Free ruleset under the GPLv2. Publicly accessible, no need for registration.

- Registered Rules — Free ruleset (requires registration). This ruleset contains subscriber rules with 30 days delay.
- Subscriber Rules (Paid) — Paid ruleset (requires subscription). This ruleset is the main ruleset and is updated twice a week (Tuesdays and Thursdays).

You can download and read more on the rules here.

Note: Once you install Snort2, it automatically creates the required directories and files. However, if you want to use the community or the paid rules, you need to indicate each rule in the snort.conf file.

Since it is a long, all-in-one configuration file, editing it without causing misconfiguration is troublesome for some users. **That is why Snort has several rule updating modules and integration tools.** To sum up, never replace your configured Snort configuration files; you must edit your configuration files manually or update your rules with additional tools and modules to not face any fail/crash or lack of feature.

- snort.conf: *Main configuration file.*
- local.rules: *User-generated rules file.*

Let's start with overviewing the main configuration file (snort.conf) `sudo gedit /etc/snort/snort.conf`

Navigate to the "Step #1: Set the network variables." section.

This section manages the scope of the detection and rule paths.

TAG NAME	INFO	EXAMPLE
HOME_NET	That is where we are protecting.	'any' OR '192.168.1.1/24'
EXTERNAL_NET	This field is the external network, so we need to keep it as 'any' or '!$HOME_NET'.	'any' OR '!$HOME_NET'
RULE_PATH	Hardcoded rule path.	/etc/snort/rules
SO_RULE_PATH	*These rules come with registered and subscriber rules.*	$RULE_PATH/so_rules
PREPROC_RULE_PATH	*These rules come with registered and subscriber rules.*	$RULE_PATH/plugin_rules

Navigate to the “Step #2: Configure the decoder.” section.

In this section, you manage the IPS mode of snort. The single-node installation model IPS model works best with “afpacket” mode. You can enable this mode and run Snort in IPS.

TAG NAME	INFO	EXAMPLE
#config daq:	IPS mode selection.	afpacket
#config daq_mode:	Activating the inline mode	inline
#config logdir:	Hardcoded default log path.	/var/logs/snort

Data Acquisition Modules (DAQ) are specific libraries used for packet I/O, bringing flexibility to process packets. It is possible to select DAQ type and mode for different purposes.

There are six DAQ modules available in Snort;

- **Pcap:** Default mode, known as Sniffer mode.
- **Afpacket:** Inline mode, known as IPS mode.
- **Ipq:** Inline mode on Linux by using Netfilter. It replaces the snort_inline patch.
- **Nfq:** Inline mode on Linux.
- **Ipfw:** Inline on OpenBSD and FreeBSD by using divert sockets, with the pf and ipfw firewalls.

- **Dump:** Testing mode of inline and normalisation.

The most popular modes are the default (pcap) and inline/IPS (Afpacket).

Navigate to the “Step #6: Configure output plugins” section.

This section manages the outputs of the IDS/IPS actions, such as logging and alerting format details. The default action prompts everything in the console application, so configuring this part will help you use the Snort more efficiently.

Navigate to the “Step #7: Customise your ruleset” section.

TAG NAME	INFO	EXAMPLE
# site specific rules	Hardcoded local and user-generated rules path.	include $RULE_PATH/local.rules
#include $RULE_PATH/	Hardcoded default/downloaded rules path.	include $RULE_PATH/rulename

Note that “#” is commenting operator. You should uncomment a line to activate it.

Task 11: Conclusion

In this room, we covered Snort, what it is, how it operates, and how to create and use the rules to investigate threats.

- Understanding and practising the fundamentals is crucial before creating advanced rules and using additional options.
- Do not create complex rules at once; try to add options step by step to notice possible syntax errors or any other problem easily.
- Do not reinvent the wheel; use it or modify/enhance it if there is a smooth rule.
- Take a backup of the configuration files before making any change.
- Never delete a rule that works properly. Comment it if you don’t need it.
- Test newly created rules before migrating them to production.

Now, we invite you to complete the snort challenge room: Snort Challenge — Live Attacks

A great way to quickly recall snort rules and commands is to download and refer to the TryHackMe snort cheatsheet.

Please refer to THM's room https://tryhackme.com/room/snort.

Thanks for reading :-)

Happy learning!

Cybersecurity Tryhackme Hacking Learning Ctf

Follow

Written by igor_sec

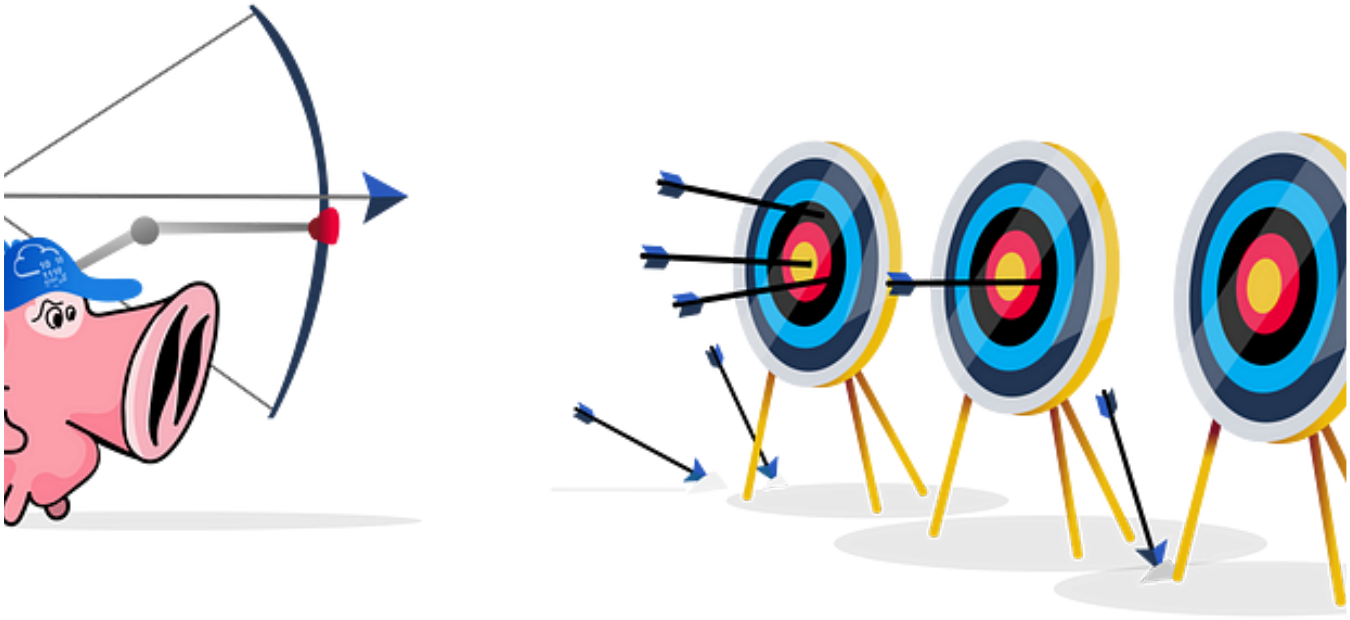
370 Followers · 11 Following

No responses yet

What are your thoughts?

Respond

More from igor_sec



igor_sec

Snort Challenge—The Basics : TryHackMe

Task 1: Introduction

Jul 20, 2023 75 2

igor_sec

TryHackMe | Zeek

Introduction to hands-on network monitoring and threat detection with Zeek (formerly Bro).

Jul 12, 2023 58 1

igor_sec

CyberDefenders | Boss Of The SOC v1

Jul 5, 2023 12

igor_sec

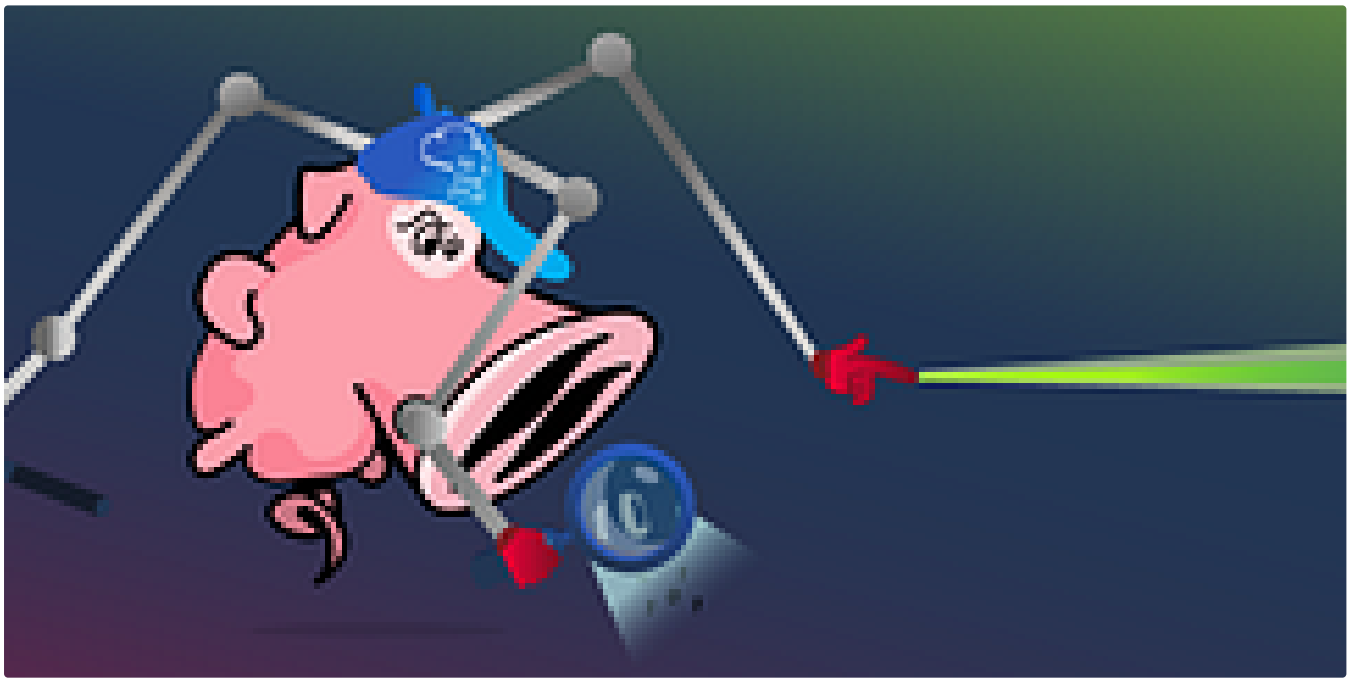
TryHackMe | Boogeyman 1

The room provided a phishing email, endpoint logs, and network traffic to analyze. By studying email headers, parsing JSON logs with JQ...

Nov 20, 2023 13

See all from igor_sec

Recommended from Medium



In T3CH by Axoloth

TryHackMe | Snort Challenge—The Basics | WriteUp

Put your snort skills into practice and write snort rules to analyse live capture network traffic

Nov 9, 2024 100

Trnty

TryHackMe | Introduction To Honeypots Walkthrough

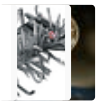
A guided room covering the deployment of honeypots and analysis of botnet activities

Sep 7, 2024 10

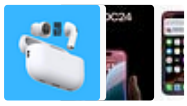
Lists

Self-Improvement 101
20 stories · 3191 saves

How to Find a Mentor
11 stories · 785 saves



Good Product Thinking
13 stories · 794 saves



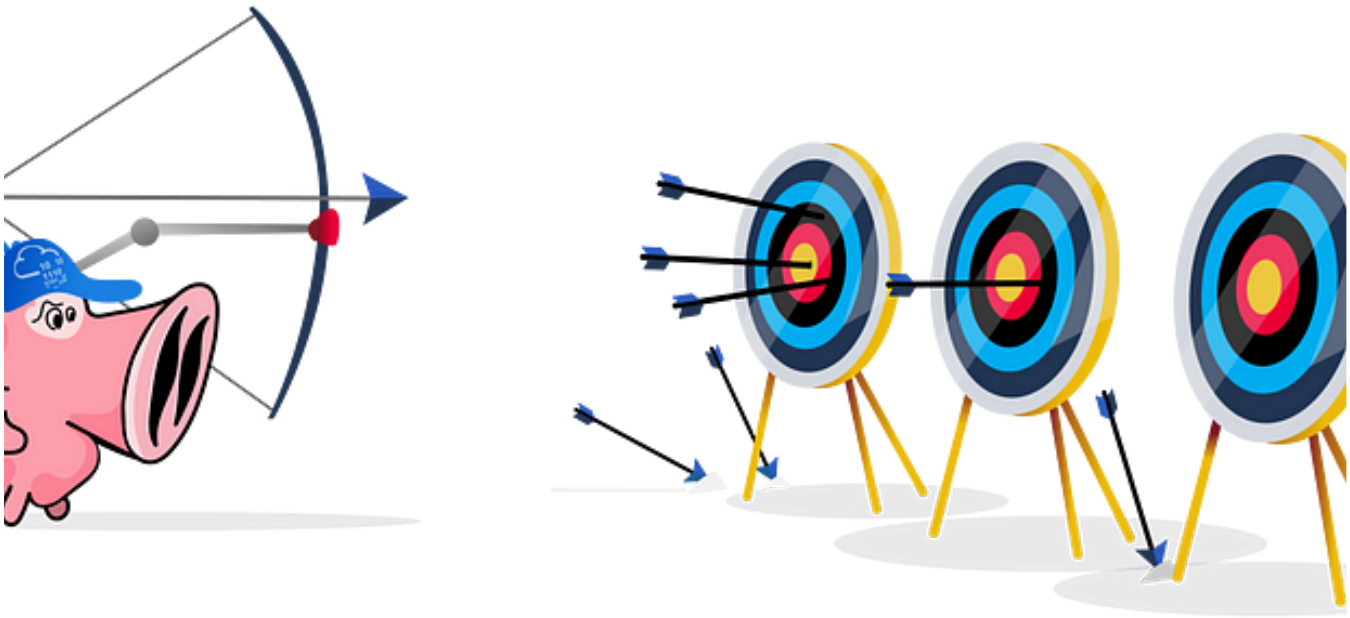
Tech & Tools
22 stories · 380 saves

In T3CH by Axoloth

TryHackMe | FlareVM: Arsenal of Tools| WriteUp

Learn the arsenal of investigative tools in FlareVM

Nov 28, 2024 50



Manivel

Snort Challenge—The Basics : TryHackMe—Medium

Snort Challenge—The Basics by TryHackMe. Writeup and Answers the question below

Dec 30, 2024 3

Fritzadriano

Retracted—TryHackMe WriteUp

IInvestigate the case of the missing ransomware. After learning about Wazuh previously, today's task is a bit different.

Sep 4, 2024 50

In System Weakness by Joseph Alan

TryHackMe Critical Write-Up: Using Volatility For Windows Memory Forensics

This challenge focuses on memory forensics, which involves understanding its concepts, accessing and setting up the environment using tools...

Jul 18, 2024 46 1