Intrusion Detection System (IDS) – Suricata

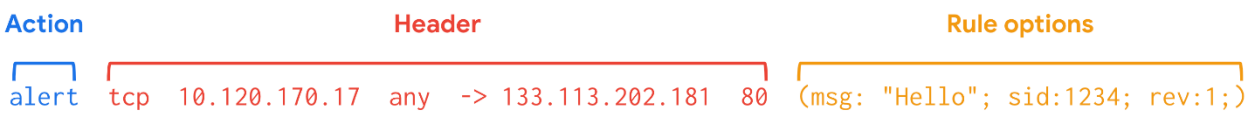
Suricata is an open-source intrusion detection system, intrusion prevention system, and network analysis tool. An intrusion detection system (IDS) is an application that monitors system activity and alerts on possible intrusions. IDS technologies help organizations monitor the activity that happens on their systems and networks to identify indications of malicious activity.

There are three main ways Suricata can be used:

- **Intrusion detection system** (**IDS**): As a network-based IDS, Suricata can monitor network traffic and alert on suspicious activities and intrusions. Suricata can also be set up as a host-based IDS to monitor the system and network activities of a single host like a computer.
- **Intrusion prevention system** (**IPS**): Suricata can also function as an intrusion prevention system (IPS) to detect and block malicious activity and traffic. Running Suricata in IPS mode requires additional configuration such as enabling IPS mode.
- **Network security monitoring** (**NSM**): In this mode, Suricata helps keep networks safe by producing and saving relevant network logs. Suricata can analyze live network traffic, existing packet capture files, and create and save full or conditional packet captures. This can be useful for forensics, incident response, and for testing signatures. For example, you can trigger an alert and capture the live network traffic to generate traffic logs, which you can then analyze to refine detection signatures.

Suricata uses signatures analysis, which is a detection method used to find events of interest. Signatures consist of three components:

- Action: The first component of a signature. It describes the action to take if network or system activity matches the signature. Examples include: alert, pass, drop, or reject.
- Header: The header includes network traffic information like source and destination IP addresses, source and destination ports, protocol, and traffic direction.
- Rule options: The rule options provide you with different options to customize signatures.



Examine alerts, logs and rules with Suricata

In this scenario, you're a security analyst who must monitor traffic on your employer's network. You'll be required to configure Suricata and use it to trigger alerts.

First, you'll explore custom rules in Suricata. Second, you'll run Suricata with a custom rule in order to trigger it, and examine the output logs in the fast.log file. Finally, you'll examine the additional output that Suricata generates in the standard eve.json log file.

For the purposes of the tests you'll run in this lab activity, you've been supplied with a sample.pcap file and a custom.rules file. These reside in your home folder.

Let's define the files:

1. The sample.pcap file is a packet capture file that contains an example of network traffic data, which you'll use to test the Suricata rules. This will allow you to simulate and repeat the exercise of monitoring network traffic.

2. The custom.rules file contains a custom rule when the lab activity starts. You'll add rules to this file and run them against the network traffic data in the sample.pcap file.

3. The fast.log file will contain the alerts that Suricata generates. The fast.log file is empty when the lab starts. Each time you test a rule, or set of rules, against the sample network traffic data, Suricata adds a new alert line to the fast.log file when all the conditions in any of the rules are met. The fast.log file can be located in the /var/log/suricata directory after Suricata runs. The fast.log file is considered to be a depreciated format and is not recommended for incident response or threat hunting tasks but can be used to perform quick checks or tasks related to quality assurance.

4. The eve.json file is the main, standard, and default log for events generated by Suricata. It contains detailed information about alerts triggered, as well as other network telemetry events, in JSON format. The eve.json file is generated when Suricate runs, and can also be located in the /var/log/suricata directory.

When you create a new rule, you'll need to test the rule to confirm whether or not it worked as expected. You can use the fast.log file to quickly compare the number of alerts generated each time you run Suricata to test a signature against the sample.pcap file.

Expectation

- Create custom rules and run them in Suricata
- Monitor traffic captured in a packet capture file
- Examine the fast.log and eve.json output

1. Examine a custom rule in Suricata

Use the cat command to display the rule in the custom.rules file:

```
analyst@b8b3bfd46e1b:~$ cat custom.rules
alert http $HOME_NET any -> $EXTERNAL_NET any (msg:"GET on wire"; flow:established,to_server; content:
"GET"; http_method; sid:12345; rev:3;)
```

- Action Alert : Instructs to alert on selected network traffic. The IDS will inspect the traffic packets and send out an alert in case it matches.
- Header http: The rule that applies is only to HTTP traffic. The arrow indicates the direction of the traffic from $HOME_NET and going to the destination IP address $EXTERNAL_NET. In this scenario, $HOME_NET is a suricata variable defined in /etc/suricata/suricata/yaml as a rule definitions. $HOME_NET is definatd as the 172.21.224/0/20 subnet.
- Rule options Rule: Customize signatures with additional parameters.
    - The msg: The alert will print out the text GET on wire.
    - The flow:established, to_server: Determines that packet from the client to the server should be matched (The handshakes: SYN-ACK packet).
    - The content: "GET" tells Suricata to look for the word GET in the http.method of the packet.
    - The sid:12345: Unique numerical value that identify the rule.
    - The rev:3 indicates the signature's version which is used to identify the signature's version.

2. Trigger a custom rule in Suricata

- List the files in the /var/log/suricata folder:

  ls -l /var/log/suricata

```
analyst@b8b3bfd46e1b:~$ ls -l /var/log/suricata
total 0
```

It is expected to see none listed at this point in the task.

- Run suricata using the custom.rules and sample.pcap files:

  sudo suricata -r sample.pcap -S custom.rules -k none

- The -r sample.pcap option specifies an input file to mimic network traffic. In this case, the sample.pcap file.
- The -S custom.rules option instructs Suricata to use the rules defined in the custom.rules file.
- The -k none option instructs Suricata to disable all checksum checks.

```
analyst@b8b3bfd46e1b:~$ sudo suricata -r sample.pcap -S custom.rules -k none
11/12/2024 -- 01:03:13 - <Notice> - This is Suricata version 4.1.2 RELEASE
11/12/2024 -- 01:03:14 - <Notice> - all 2 packet processing threads, 4 management threads initialized,
 engine started.
11/12/2024 -- 01:03:14 - <Notice> - Signal Received.  Stopping engine.
11/12/2024 -- 01:03:16 - <Notice> - Pcap-file module read 1 files, 200 packets, 54238 bytes
```

- List the files in the /var/log/suricata folder again:

```
analyst@b8b3bfd46e1b:~$ ls -l /var/log/suricata
total 16
-rw-r--r-- 1 root root 1432 Dec 11 01:03 eve.json
-rw-r--r-- 1 root root  292 Dec 11 01:03 fast.log
-rw-r--r-- 1 root root 2687 Dec 11 01:03 stats.log
-rw-r--r-- 1 root root  357 Dec 11 01:03 suricata.log
```

- Use the cat command to display the fast.log file generated by Suricata:

```
analyst@b8b3bfd46e1b:~$ cat /var/log/suricata/fast.log
11/23/2022-12:38:34.624866  [**] [1:12345:3] GET on wire [**] [Classification: (null)] [Priority: 3] {TCP} 172.21.224.2:
49652 -> 142.250.1.139:80
11/23/2022-12:38:58.958203  [**] [1:12345:3] GET on wire [**] [Classification: (null)] [Priority: 3] {TCP} 172.21.224.2:
58494 -> 142.250.1.102:80
```

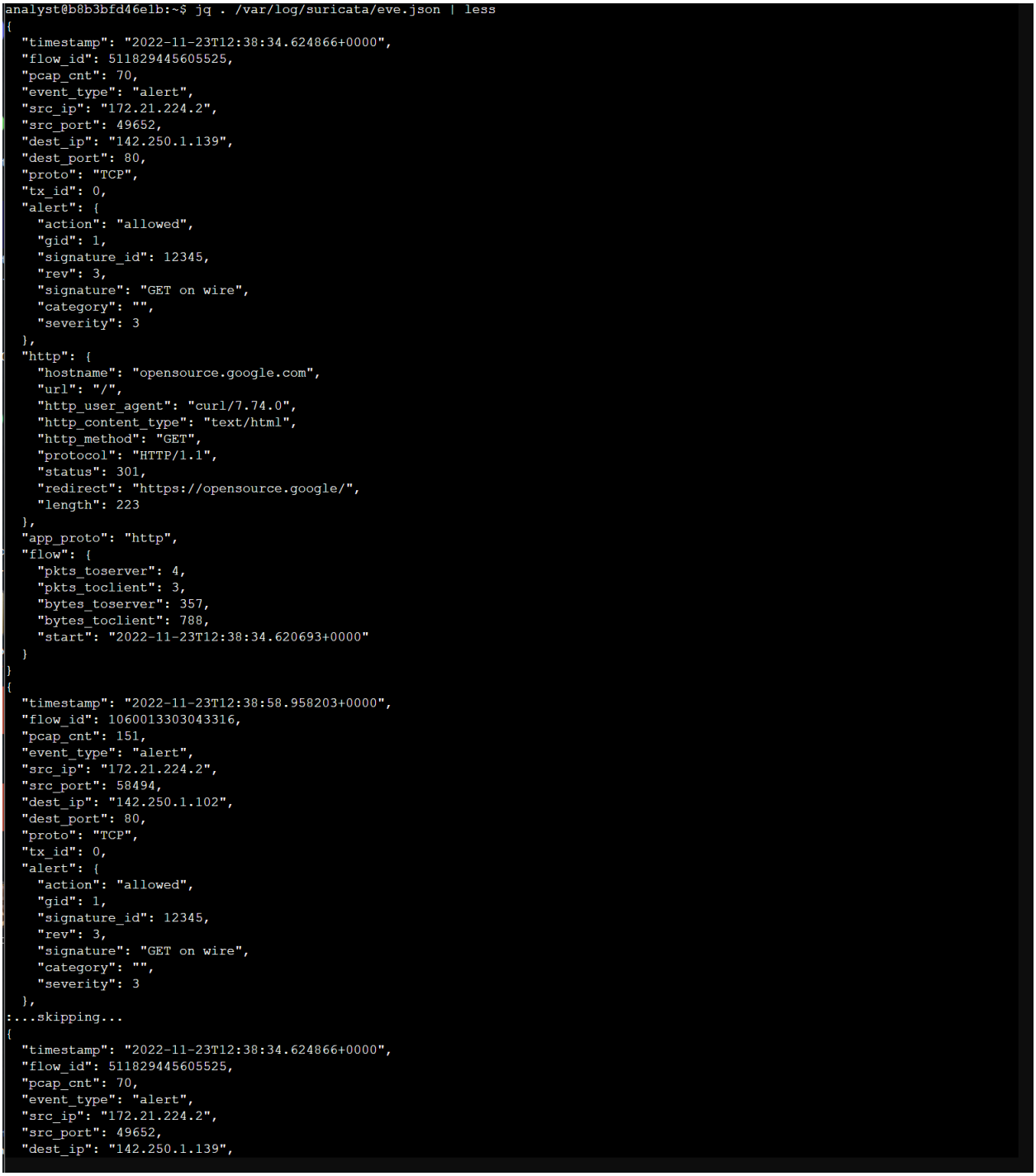
Each line or entry in the fast.log file corresponds to an alert generated by Suricata when it processes a packet that meets the conditions of an alert generating rule. Each alert line includes the message that identifies the rule that triggered the alert, as well as the source, destination, and direction of the traffic.

3. Examine eve.json output

- Use the cat command to display the entries in the eve.json file:


```
analyst@b8b3bfd46e1b:~$ cat /var/log/suricata/eve.json
{"timestamp":"2022-11-23T12:38:34.624866+0000","flow_id":511829445605525,"pcap_cnt":70,"event_type":"alert","src_ip":"17
2.21.224.2","src_port":49652,"dest_ip":"142.250.1.139","dest_port":80,"proto":"TCP","tx_id":0,"alert":{"action":"allowed
","gid":1,"signature_id":12345,"rev":3,"signature":"GET on wire","category":"","severity":3},"http":{"hostname":"opensou
rce.google.com","url":"\/","http_user_agent":"curl\/7.74.0","http_content_type":"text\/html","http_method":"GET","protoc
ol":"HTTP\/1.1","status":301,"redirect":"https:\/\/opensource.google\/","length":223},"app_proto":"http","flow":{"pkts_t
oserver":4,"pkts_toclient":3,"bytes_toserver":357,"bytes_toclient":788,"start":"2022-11-23T12:38:34.620693+0000"}}
{"timestamp":"2022-11-23T12:38:58.958203+0000","flow_id":1060013303043316,"pcap_cnt":151,"event_type":"alert","src_ip":"
172.21.224.2","src_port":58494,"dest_ip":"142.250.1.102","dest_port":80,"proto":"TCP","tx_id":0,"alert":{"action":"allow
ed","gid":1,"signature_id":12345,"rev":3,"signature":"GET on wire","category":"","severity":3},"http":{"hostname":"opens
ource.google.com","url":"\/","http_user_agent":"curl\/7.74.0","http_content_type":"text\/html","http_method":"GET","prot
ocol":"HTTP\/1.1","status":301,"redirect":"https:\/\/opensource.google\/","length":223},"app_proto":"http","flow":{"pkts
_toserver":4,"pkts_toclient":3,"bytes_toserver":357,"bytes_toclient":797,"start":"2022-11-23T12:38:58.955636+0000"}}
```


- Let's display the entries in an improved format: jq . /var/log/suricata/eve/json | less.



- Use the jq command to extract specific event data from the eve.json file:

jq -c "[.timestamp,.flow_id,.alert.signature,.proto,.dest_ip]" /var/log/suricata/eve.json