Project Description

I take on the role of a security professional that monitors my employer's network for suspicious traffic.

In this lab I learn more about Suricata, an open source intrusion detection system, intrusion prevention system and network analysis tool.

Suricata monitors network interfaces and enforces created rules (also called signatures) on the packets that travel through it. It decides whether a packet should be dropped with an alert; rejected with an alert; generate an alert when passed; or be allowed to pass through as an exception and also generate an alert (see "Explore custom rules in Suricata" for more information).

In this lab I will learn about rule creation, alerts and logs in Suricata.

Provided materials

In this lab I am provided with a sample.pcap packet capture file that contains example network traffic data I will use to test Suricata.

I am also provided with a custom.rules file that allows me to add custom Suricata rules to run network traffic data from sample.pcap against.

Explore custom rules in Suricata

Fast.log

I will use fast.log (see "Run Suricata with a custom rule and examine fast.log" section for more information on fast.log) to check the number of alerts generated when I run Suricata to test a rule against sample.pcap network traffic.

Action

```
analyst@8f363633a8e1:~$ 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;)
```

The first part of the custom rule is the action, here it is alert, that states the action to take when specified conditions are met. The action can be drop, reject, alert, pass.

Drop generates an alert but drops traffic rather than allowing it to pass; drop is only used when Suricata is running in IPS mode.

Reject does not allow traffic to pass through the network interface. A TCP reset packet is sent as a reply, and Suricata drops the matching packet. The TCP reset packet tells computers to stop communicating with each other.

Alert sends out an alert when specified conditions are met.

The pass action allows traffic to pass through the network interface, and can be used to specify exclusions to other rules (for example singling out a specific IP address to allow traffic from from rules that otherwise block network traffic).

This is an example of a rule that blocks network http traffic on all computers on a home network except the machine with ip address 172.17.0.77 if the web browser used is not Mozilla 5.0.

```
drop http $HOME_NET any -> $EXTERNAL_NET any
(msg:"Blocking HTTP traffic on home network";
flow:established,to_server; sid: 12345; rev:1;)
pass http 172.17.0.77 any -> $EXTERNAL_NET any
(msg:"Exception for machine 172.17.0.77";
flow:established,to_server;content:!"Mozilla/5.0";htt
p_user_agent; sid: 12365; rev:1;)
```

Suricata loads rules in the order in which they are specified in a configuration file but processes rules in the order of: pass, drop, reject, alert. Rule order determines the final action taken for a packet, higher priority rules are given more importance when deciding the final action to take.

Header

The header section in the above screenshot is `http $HOME_NET any -> $EXTERNAL_NET any`.

The header states the network traffic protocol, source IP address and port, traffic direction and destination IP address and port.

Here http states that the rule only applies to http traffic.

The arrow states traffic is flowing from $HOME_NET to $EXTERNAL_NET.
$HOME_NET is a variable defined in /etc/suricata/suricata.yaml, it is used as a placeholder for the local or home network and is defined in this lab as 172.21.224.0/20. There are 32-20=12 host bits and therefore 2^12 -2 = 4096 – 2 = 4094 host addresses (reserving the first address for the network address and the last address for the broadcast address) on 4096/256 = 16 network subnets. The usable IP host range is therefore 172.21.224.1 – 172.21.239.254.

The any word means Suricata matches traffic from any port on the $HOME_NET and $EXTERNAL_NET networks.

Rule Options

The rule options are specified in the above screenshot in `(msg:"GET on wire"; flow:established,to_server; content:"GET"; http_method; sid:12345; rev:3;)`.

The msg option states the alert text that will be sent when the alert is triggered, here it is “GET on wire”.

The flow:established,to_server option states that packets from client to server should be identified. A server is the device that responds to the initial SYN packet with a SYN-ACK packet.

The content:"GET" option tells Suricata to look for the word GET in the http.method part of the packet.

The sid:12345 specifies the unique signature ID to identify the rule.

The rev:3 option states the signature's revision, it identifies the signature's version.

This rule sends an alert when Suricata sees the text GET as the HTTP method in a HTTP packet from the home network going to the external network.

Run Suricata with a custom rule and examine fast.log

The fast.log file is used for performing quick checks or quality assurance, it is a deprecated format that is not recommended for incident response or threat hunting.

The fast.log file is located in /var/log/suricata.

```
analyst@268a156e9ca9:~$ ls -l /var/log/suricata
total 0
```

The /var/log/suricata directory is initially empty.

```
analyst@268a156e9ca9:~$ sudo suricata -r sample.pcap -S custom.rules -k none
29/11/2023 -- 04:20:54 - <Notice> - This is Suricata version 6.0.1 RELEASE running
in USER mode
29/11/2023 -- 04:20:55 - <Notice> - all 2 packet processing threads, 4 management t
hreads initialized, engine started.
29/11/2023 -- 04:20:55 - <Notice> - Signal Received.  Stopping engine.
29/11/2023 -- 04:20:55 - <Notice> - Pcap-file module read 1 files, 200 packets, 542
38 bytes
```

I then run Suricata with the custom.rules and sample.pcap network capture data:
The -r sample.pcap option states the input file to simulate network traffic.
The -S custom.rules option tells Suricata to use rules defined in custom.rules.
The -k none opton tells Suricata to disable checksum checks.

Checksums tell if a packet has been modified in transit. I do not need Suricata to check the integrity of checksums because I am using a network traffic file.

The following files are generated in /var/log/suricata/log after running Suricata with the custom.rules and sample.pcap files.

```
analyst@268a156e9ca9:~$ cat /var/log/suricata/fast.log
11/23/2022-12:38:34.624866  [**] [1:12345:3] GET on wire [**] [Classification: (nul
l)] [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: (nul
l)] [Priority: 3] {TCP} 172.21.224.2:58494 -> 142.250.1.102:80
```

The lines in fast.lg are alerts generated by Suricata when it encounters a packet that meets the conditions of an alert rule. The [1:12345:3] identifies the rule number, signature ID and revision that triggered the alert. The source IP address, direction of traffic and destination IP address are also shown.

Examine eve.json

The eve.json log file is the main log for events in Suricata. It contains detailed information about triggered alerts for incident response or threat hunting. It is in JavaScript Object Notation (JSON) and is also stored in /var/log/suricata.

```
analyst@268a156e9ca9:~$ cat /var/log/suricata/eve.json
{"timestamp":"2022-11-23T12:38:34.624866+0000","flow_id":1638792356853909,"pcap_cnt
":70,"event_type":"alert","src_ip":"172.21.224.2","src_port":49652,"dest_ip":"142.2
50.1.139","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":"opensource.google.com","url":"/","http_user_agent":"curl/7.74
.0","http_content_type":"text/html","http_method":"GET","protocol":"HTTP/1.1","stat
us":301,"redirect":"https://opensource.google/","length":223},"app_proto":"http","f
low":{"pkts_toserver":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":1074023486362868,"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":"allowed","gid"
:1,"signature_id":12345,"rev":3,"signature":"GET on wire","category":"","severity":
3},"http":{"hostname":"opensource.google.com","url":"/","http_user_agent":"curl/7.7
4.0","http_content_type":"text/html","http_method":"GET","protocol":"HTTP/1.1","sta
tus":301,"redirect":"https://opensource.google/","length":223},"app_proto":"http","
flow":{"pkts_toserver":4,"pkts_toclient":3,"bytes_toserver":357,"bytes_toclient":79
7,"start":"2022-11-23T12:38:58.955636+0000"}}
```

The contents of eve.json are in JSON format. Another program called jq can be used to improve the formatting of the contents of this file.

```
"timestamp": "2022-11-23T12:38:34.624866+0000"
"flow_id": 1638792356853909,
"pcap_cnt": 70,
"event_type": "alert",
"src_ip": "172.21.224.2",
"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": "opensource.google.com",
  "url": "/",
  "http_user_agent": "curl/7.74.0",
  "http_content_type": "text/html",
  "http_method": "GET",
  "protocol": "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": 788,
  "start": "2022-11-23T12:38:34.620693+0000"
}
```

The lowercase f and b keys can be used to move forward and back in the output. The space button advances the content one line. Pressing Q exits the less command.

```
analyst@268a156e9ca9:~$ jq -c "[.timestamp,.flow_id,.alert.signature,.proto,.dest_i
p]" /var/log/suricata/eve.json
["2022-11-23T12:38:34.624866+0000",1638792356853909,"GET on wire","TCP","142.250.1.
139"]
["2022-11-23T12:38:58.958203+0000",1074023486362868,"GET on wire","TCP","142.250.1.
102"]
```

The jq command can be used to extract specific details from the eve.json file:
`jq -c "[.timestamp,.flow_id,.alert.signature,.proto,.dest_ip]" /var/log/suricata/eve.json` tells jq to extract only the timestamp, flow id, alert signature, protocol and destination IP.

The flowid is a unique 16 digit number that identifies the log entry. It can be used to extract the full log details of packets of interest. The network flow is a sequence of packets that share common characteristics such as IP address and protocol, etc.

```
analyst@268a156e9ca9:~$ jq "select(.flow_id==1638792356853909)" /var/log/suricata/e
ve.json
{
  "timestamp": "2022-11-23T12:38:34.624866+0000",
  "flow_id": 1638792356853909,
  "pcap_cnt": 70,
  "event_type": "alert",
  "src_ip": "172.21.224.2",
  "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": "opensource.google.com",
    "url": "/",
    "http_user_agent": "curl/7.74.0",
    "http_content_type": "text/html",
    "http_method": "GET",
    "protocol": "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": 788,
```

Summary

I used Suricata to analyze network traffic and trigger alerts. I gained practical experience in using Suricata to create custom rules and run against them, monitor traffic in packet capture files and examined the fast.log and eve.json logs.

References

Google Cybersecurity Certificate (2023) Examine alerts, logs, and rules with Suricata.