Suricata IDS/IPS configuration

Student: **Yagub Hajiyev**

Suricata installation and configuration

Suricata installation is explained clearly for both, **Linux** and **Windows**, platforms in its own web site, but as a short instraction I displayed the installation process for **Linux RedHat** family.

1-Install **EPEL** repo into your **Linux** with "**sudo dnf install epel-release dnf-plugins-core**" command.

```
[yagub@suricata ~]$ sudo dnf install epel-release dnf-plugins-core
Last metadata expiration check: 0:14:51 ago on Sun 14 Apr 2024 04:56:58 AM EDT.
Package dnf-plugins-core-4.0.21-23.0.1.el8.noarch is already installed.
Dependencies resolved.
================================================================================
 Package                 Architecture    Version                 Repository    Size
================================================================================
Installing:
 oracle-epel-release-el8 x86_64          1.0-5.el8               ol8_baseos_latest  15 k
Installing dependencies:
 yum-utils               noarch          4.0.21-23.0.1.el8       ol8_baseos_latest  75 k

Transaction Summary
================================================================================
Install  2 Packages

Total download size: 90 k
Installed size: 41 k
Is this ok [y/N]: y
```

2-After configuring **EPEL** repo, run "**sudo dnf copr enable @oisf/suricata-7.0**" command to enable **COPR** repo for **Suricata** version **7.0**.

```
[yagub@suricata ~]$ sudo dnf copr enable @oisf/suricata-7.0
Enabling a Copr repository. Please note that this repository is not part
of the main distribution, and quality may vary.

The Fedora Project does not exercise any power over the contents of
this repository beyond the rules outlined in the Copr FAQ at
<https://docs.pagure.org/copr.copr/user_documentation.html#what-i-can-build-in-copr>,
and packages are not held to any quality or security level.

Please do not file bug reports about these packages in Fedora
Bugzilla. In case of problems, contact the owner of this repository.

Do you really want to enable copr.fedorainfracloud.org/@oisf/suricata-7.0? [y/N]: y
```

3-Now we can download **Suricata** into our **Linux OS** with "**sudo dnf install suricata**" command.

```
[yagub@suricata ~]$ sudo dnf install suricata
Copr repo for suricata-7.0 owned by @oisf                1.4 kB/s |  14 kB     00:09
Oracle Linux 8 EPEL Packages for Development (x86_64)    3.4 MB/s |  64 MB     00:18
Oracle Linux 8 EPEL Modular Packages for Development (x86_64)  34 kB/s | 322 kB     00:09
Dependencies resolved.
================================================================================
 Package             Arch     Version          Repository                    Size
================================================================================
Installing:
 suricata            x86_64   1:7.0.4-1.el8    copr:copr.fedorainfracloud.org:group_oisf:suricata-7.0   3.7 M
Installing dependencies:
 dpdk                x86_64   21.11-3.el8      ol8_appstream                 3.8 M
 hiredis             x86_64   0.13.3-13.el8    ol8_developer_EPEL             38 k
 hyperscan           x86_64   5.3.0-5.el8      ol8_developer_EPEL            3.1 M
 libnetfilter_queue  x86_64   1.0.4-3.el8      ol8_baseos_latest              31 k

Transaction Summary
================================================================================
Install  5 Packages

Total download size: 11 M
Installed size: 37 M
Is this ok [y/N]: y
```

After the installation, we need to do some configurations for the tool. First, we have to learn the interface's name which we use in our OS. In this example, it is named **ens160**.

```
[yagub@suricata ~]$ ifconfig
ens160: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.130  netmask 255.255.255.0  broadcast 192.168.1.255
```

Suricata configuration file is called **suricata.yaml** (it is designed in **YAML** format) and this file is located under **/etc/suricata/suricata.yaml** path.

```
[yagub@suricata ~]$ sudo ls /etc/suricata/
classification.config  reference.config  suricata.yaml  threshold.config
[yagub@suricata ~]$
```

We need to make additional changes in this file to use **Suricata** properly. In this lab, we will use **IDS** and **IPS** in host based, which means we could control netwrok traffic only in our host, it will not affect any other host in the network. Now open the **suricata.yaml** file and change the **HOME_NET** variable's value to your host's **IP** address as shown below. The host's **IP** address is **192.168.1.130/32** in this lab.

NOTE: don't forget to type **CIDR** as **/32**, otherwise the tool will try to work in network mode.

```
  GNU nano 2.9.8                        /etc/suricata/suricata.yaml

%YAML 1.1
---

# Suricata configuration file. In addition to the comments describing all
# options in this file, full documentation can be found at:
# https://docs.suricata.io/en/latest/configuration/suricata-yaml.html

# This configuration file generated by Suricata 7.0.4.
suricata-version: "7.0"

##
## Step 1: Inform Suricata about your network
##

vars:
  # more specific is better for alert accuracy and performance
  address-groups:
    HOME_NET: "[192.168.0.0/16,10.0.0.0/8,172.16.0.0/12]"
    #HOME_NET: "[192.168.0.0/16]"
    #HOME_NET: "[10.0.0.0/8]"
    #HOME_NET: "[172.16.0.0/12]"
    #HOME_NET: "any"

    EXTERNAL_NET: "!$HOME_NET"
    #EXTERNAL_NET: "any"

    HTTP_SERVERS: "$HOME_NET"
                                [ Read 2173 lines ]
^G Get Help    ^O Write Out   ^W Where Is    ^K Cut Text    ^J Justify     ^C Cur Pos     M-U Undo
^X Exit        ^R Read File   ^\ Replace     ^U Uncut Text  ^T To Spell   ^_ Go To Line  M-E Redo
```

```
vars:
  # more specific is better for alert accuracy and performance
  address-groups:
    HOME_NET: "[192.168.1.130/32]"
    #HOME_NET: "[192.168.0.0/16]"
    #HOME_NET: "[10.0.0.0/8]"
    #HOME_NET: "[172.16.0.0/12]"
    #HOME_NET: "any"
```

After configuring the home network variable, find the **af-packet:** and change interface's name to yours.

```
# Linux high speed capture support
af-packet:
  - interface: eth0
```

```
# Linux high speed capture support
af-packet:
  - interface: ens160
```

We will use **Suricata**'s **GeoIP** filtering feature in our lab, so it also has to be configured in this file too. Find the **geoip-database** and comment it out by taking **#** symbol. **Suricata** uses **Maxmind**'s databases as default, and the database's path is defined under **/usr/local/share/GeoLite2** directory, but the **GeoLite2** folder isn't created as default. So we must create it and put our database file into it ourselves in following steps.

```
# GeoIP2 database file. Specify path and filename of GeoIP2 database
# if using rules with "geoip" rule option.
#geoip-database: /usr/local/share/GeoLite2/GeoLite2-Country.mmdb
```

```
# GeoIP2 database file. Specify path and filename of GeoIP2 database
# if using rules with "geoip" rule option.
geoip-database: /usr/local/share/GeoLite2/GeoLite2-Country.mmdb
```

In order to use **Maxmind GeoIP** database you need to subscribe that platfor or you can just download it from any other source on the **Internet**. Below, I shared one of these sources, download it into your host.
Database file: **https://git.io/GeoLite2-Country.mmdb**

```
[root@suricata ~]# wget https://git.io/GeoLite2-Country.mmdb
--2024-04-14 19:10:27--  https://git.io/GeoLite2-Country.mmdb
Resolving git.io (git.io)... 140.82.114.21
Connecting to git.io (git.io)|140.82.114.21|:443... connected.
HTTP request sent, awaiting response... 301 Moved Permanently
Location: https://github.com/P3TERX/GeoLite.mmdb/releases/latest/download/GeoLite2-Country.mmdb [following]
--2024-04-14 19:10:33--  https://github.com/P3TERX/GeoLite.mmdb/releases/latest/download/GeoLite2-Country.mmdb
Resolving github.com (github.com)... 140.82.121.3
Connecting to github.com (github.com)|140.82.121.3|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://github.com/P3TERX/GeoLite.mmdb/releases/download/2024.04.13/GeoLite2-Country.mmdb [following]
--2024-04-14 19:10:37--  https://github.com/P3TERX/GeoLite.mmdb/releases/download/2024.04.13/GeoLite2-Country.mmdb
Reusing existing connection to github.com:443.
HTTP request sent, awaiting response... 302 Found
Location: https://objects.githubusercontent.com/github-production-release-asset-2e65be/249855791/c619d627-1ebd-411f-a565-12f9a47df90f?X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Credential=AKIAVCODYLSA53PQK4ZA%2F20240414%2Fus-east-1%2Fs3%2Faws4_request&X-Amz-Date=20240414T151037Z&X-Amz-Expires=300&X-Amz-Signature=c949489c5093c896a5558f2de4f75efb242ab4bf322b0fd6d5074d8a94264ce0&X-Amz-SignedHeaders=host&actor_id=0&key_id=0&repo_id=249855791&response-content-disposition=attachment%3B%20filename%3DGeoLite2-Country.mmdb&response-content-type=application%2Foctet-stream [following]
--2024-04-14 19:10:37--  https://objects.githubusercontent.com/github-production-release-asset-2e65be/249855791/c619d627-1ebd-411f-a565-12f9a47df90f?X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Credential=AKIAVCODYLSA53PQK4ZA%2F20240414%2Fus-east-1%2Fs3%2Faws4_request&X-Amz-Date=20240414T151037Z&X-Amz-Expires=300&X-Amz-Signature=c949489c5093c896a5558f2de4f75efb242ab4bf322b0fd6d5074d8a94264ce0&X-Amz-SignedHeaders=host&actor_id=0&key_id=0&repo_id=249855791&response-content-disposition=attachment%3B%20filename%3DGeoLite2-Country.mmdb&response-content-type=application%2Foctet-stream
Resolving objects.githubusercontent.com (objects.githubusercontent.com)... 185.199.108.133, 185.199.109.133, 185.199.110.133, ...
Connecting to objects.githubusercontent.com (objects.githubusercontent.com)|185.199.108.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 6444337 (6.1M) [application/octet-stream]
Saving to: 'GeoLite2-Country.mmdb'

GeoLite2-Country.mmdb           100%[=======================================================>]   6.15M  5.94MB/s    in 1.0s

2024-04-14 19:10:43 (5.94 MB/s) - 'GeoLite2-Country.mmdb' saved [6444337/6444337]
```

Later create the **GeoLite2** directory under **/usr/local/share** path, move the database file into this directory and change the user and group owner of file to **suricata** as displayed below.

```
[root@suricata ~]# ll
total 6304
-rw-------. 1 root root    1034 Mar 31 16:30 anaconda-ks.cfg
-rw-r--r--. 1 root root 6444337 Apr 13 05:32 GeoLite2-Country.mmdb
-rw-r--r--. 1 root root    1261 Mar 31 16:49 initial-setup-ks.cfg
[root@suricata ~]# mkdir /usr/local/share/GeoLite2
[root@suricata ~]# mv GeoLite2-Country.mmdb /usr/local/share/GeoLite2/
[root@suricata ~]# chown -R suricata.suricata /usr/local/share/GeoLite2/
[root@suricata ~]# ll -d /usr/local/share/GeoLite2/
drwxr-xr-x. 2 suricata suricata 35 Apr 14 19:11 /usr/local/share/GeoLite2/
[root@suricata ~]#
```

Once we finish modifing configuration file, we can check if everything is okay with it. Run the "**sudo suricata -T -c /etc/suricata/suricata.yaml**" command to test configuration file. This test will display errors if there are misconfigurations od **Suricata**.

```
[yagub@suricata ~]$ sudo nano /etc/suricata/suricata.yaml
[yagub@suricata ~]$ sudo suricata -T -c /etc/suricata/suricata.yaml
i: suricata: This is Suricata version 7.0.4 RELEASE running in SYSTEM mode
W: detect: No rule files match the pattern /var/lib/suricata/rules/suricata.rules
[yagub@suricata ~]$
```

Suricata service is used by the user named **suricata**, and we need to define our interface name for this user in another configuration file. This file is located under **/etc/sysconfig** directory and named **suricata**. Open the

file and replace default interface name **eth0** with your interface name, which is **ens160** in this lab.

```
[yagub@suricata ~]$ sudo nano /etc/sysconfig/suricata
```

```
  GNU nano 2.9.8                                        /etc/sysconfig/suricata

# The following parameters are the most commonly needed to configure
# suricata. A full list can be seen by running /sbin/suricata --help
# -i <network interface device>
# --user <acct name>
# --group <group name>

# Add options to be passed to the daemon
OPTIONS="-i eth0 --user suricata "
```

```
  GNU nano 2.9.8                                        /etc/sysconfig/suricata

# The following parameters are the most commonly needed to configure
# suricata. A full list can be seen by running /sbin/suricata --help
# -i <network interface device>
# --user <acct name>
# --group <group name>

# Add options to be passed to the daemon
OPTIONS="-i ens160 --user suricata "
```

After modifing **suricata** file, run the **suricata-update** command. This command will test the **suricata.yaml** file, create the **rules** directory with **suricata.rules** file and pull some default **IDS/IPS** rules.

```
[yagub@suricata ~]$ sudo suricata-update
14/4/2024 -- 05:25:25 - <Info> -- Using data-directory /var/lib/suricata.
14/4/2024 -- 05:25:25 - <Info> -- Using Suricata configuration /etc/suricata/suricata.yaml
14/4/2024 -- 05:25:25 - <Info> -- Using /usr/share/suricata/rules for Suricata provided rules.
14/4/2024 -- 05:25:25 - <Info> -- Found Suricata version 7.0.4 at /sbin/suricata.
14/4/2024 -- 05:25:25 - <Info> -- Loading /etc/suricata/suricata.yaml
14/4/2024 -- 05:25:25 - <Info> -- Disabling rules for protocol pgsql
14/4/2024 -- 05:25:25 - <Info> -- Disabling rules for protocol modbus
14/4/2024 -- 05:25:25 - <Info> -- Disabling rules for protocol dnp3
14/4/2024 -- 05:25:25 - <Info> -- Disabling rules for protocol enip
14/4/2024 -- 05:25:25 - <Info> -- No sources configured, will use Emerging Threats Open
14/4/2024 -- 05:25:25 - <Info> -- Fetching https://rules.emergingthreats.net/open/suricata-7.0.4/emerging.rules.tar.gz.
 100% - 4243799/4243799
14/4/2024 -- 05:25:33 - <Info> -- Done.
14/4/2024 -- 05:25:33 - <Info> -- Loading distribution rule file /usr/share/suricata/rules/app-layer-events.rules
14/4/2024 -- 05:25:33 - <Info> -- Loading distribution rule file /usr/share/suricata/rules/decoder-events.rules
14/4/2024 -- 05:25:33 - <Info> -- Loading distribution rule file /usr/share/suricata/rules/dhcp-events.rules
14/4/2024 -- 05:25:33 - <Info> -- Loading distribution rule file /usr/share/suricata/rules/dnp3-events.rules
14/4/2024 -- 05:25:33 - <Info> -- Loading distribution rule file /usr/share/suricata/rules/dns-events.rules
14/4/2024 -- 05:25:33 - <Info> -- Loading distribution rule file /usr/share/suricata/rules/files.rules
14/4/2024 -- 05:25:33 - <Info> -- Loading distribution rule file /usr/share/suricata/rules/http-events.rules
14/4/2024 -- 05:25:33 - <Info> -- Loading distribution rule file /usr/share/suricata/rules/ipsec-events.rules
14/4/2024 -- 05:25:33 - <Info> -- Loading distribution rule file /usr/share/suricata/rules/kerberos-events.rules
14/4/2024 -- 05:25:33 - <Info> -- Loading distribution rule file /usr/share/suricata/rules/modbus-events.rules
14/4/2024 -- 05:25:33 - <Info> -- Loading distribution rule file /usr/share/suricata/rules/nfs-events.rules
14/4/2024 -- 05:25:33 - <Info> -- Loading distribution rule file /usr/share/suricata/rules/ntp-events.rules
14/4/2024 -- 05:25:33 - <Info> -- Loading distribution rule file /usr/share/suricata/rules/smb-events.rules
14/4/2024 -- 05:25:33 - <Info> -- Loading distribution rule file /usr/share/suricata/rules/smtp-events.rules
14/4/2024 -- 05:25:33 - <Info> -- Loading distribution rule file /usr/share/suricata/rules/stream-events.rules
14/4/2024 -- 05:25:33 - <Info> -- Loading distribution rule file /usr/share/suricata/rules/tls-events.rules
14/4/2024 -- 05:25:33 - <Info> -- Ignoring file 51f50afad3967db2f32b753c395f52bb/rules/emerging-deleted.rules
14/4/2024 -- 05:25:36 - <Info> -- Loaded 48646 rules.
14/4/2024 -- 05:25:36 - <Info> -- Disabled 14 rules.
14/4/2024 -- 05:25:36 - <Info> -- Enabled 0 rules.
14/4/2024 -- 05:25:36 - <Info> -- Modified 0 rules.
14/4/2024 -- 05:25:36 - <Info> -- Dropped 0 rules.
14/4/2024 -- 05:25:36 - <Info> -- Enabled 135 rules for flowbit dependencies.
14/4/2024 -- 05:25:36 - <Info> -- Creating directory /var/lib/suricata/rules.
14/4/2024 -- 05:25:36 - <Info> -- Backing up current rules.
14/4/2024 -- 05:25:36 - <Info> -- Writing rules to /var/lib/suricata/rules/suricata.rules: total: 48646; enabled: 37144; added: 48646; rem
oved 0; modified: 0
14/4/2024 -- 05:25:36 - <Info> -- Writing /var/lib/suricata/rules/classification.config
14/4/2024 -- 05:25:36 - <Info> -- Testing with suricata -T.
14/4/2024 -- 05:26:12 - <Info> -- Done.
[yagub@suricata ~]$
```

Suricata rules locate under **/var/lib/suricata/rules** path and the default rules file is **suricata.rules**.

```
[yagub@suricata ~]$ sudo ls /var/lib/suricata/rules
classification.config  suricata.rules
```

Although there is ready to use rules, we will create our own rules to understand the work principles of **IDS/IPS** service. Below I created the second rules file, **cs301.rules**, and typed first rule to test it by sending **ICMP** packet (**ping**). In **Suricata**, all rules files' names has to end with **".rules**" ending.

```
[yagub@suricata ~]$ sudo nano /var/lib/suricata/rules/cs301.rules
```

```
  GNU nano 2.9.8                                  /var/lib/suricata/rules/cs301.rules

alert icmp any any -> any any (msg:"Ping Detected"; sid:1;)
```

After adding our custom rules file, we need to define it in the **suricata.yaml** file. Open the file and find the "**rule-files:**" option, under this option replace default file's name with your file's name, which is **cs301.rules** in this lab. We can specify multiple file names one under another and **Suricata** will check all these files for matching signatures.

```
default-rule-path: /var/lib/suricata/rules

rule-files:
  - suricata.rules
```

```
default-rule-path: /var/lib/suricata/rules

rule-files:
  - cs301.rules
```

It is recommended to check the configuration after making modification on it every time.

```
[yagub@suricata ~]$ sudo nano /var/lib/suricata/rules/cs301.rules
[yagub@suricata ~]$ sudo nano /etc/suricata/suricata.yaml
[yagub@suricata ~]$ sudo suricata -T -c /etc/suricata/suricata.yaml
i: suricata: This is Suricata version 7.0.4 RELEASE running in SYSTEM mode
i: suricata: Configuration provided was successfully loaded. Exiting.
[yagub@suricata ~]$
```

Now we are ready to start **Suricata** as service, use "**sudo systemctl enable --now suricata.service**" command to enable it and start.
NOTE: always check for error logs in service's status.

```
[yagub@suricata ~]$ systemctl enable --now suricata.service
Created symlink /etc/systemd/system/multi-user.target.wants/suricata.service → /usr/lib/systemd/system/suricata.service.
[yagub@suricata ~]$ systemctl status suricata.service
● suricata.service - Suricata Intrusion Detection Service
   Loaded: loaded (/usr/lib/systemd/system/suricata.service; enabled; vendor preset: disable>
   Active: active (running) since Sun 2024-04-14 05:38:21 EDT; 11s ago
     Docs: man:suricata(1)
  Process: 35680 ExecStartPre=/bin/rm -f /var/run/suricata.pid (code=exited, status=0/SUCCES>
 Main PID: 35682 (Suricata-Main)
    Tasks: 10 (limit: 16720)
   Memory: 52.5M
   CGroup: /system.slice/suricata.service
           └─35682 /sbin/suricata -c /etc/suricata/suricata.yaml --pidfile /var/run/suricata>
```

In order to test our first rule, we need to send an **ICMP** packet with **ping** command.

```
[root@suricata ~]# ping -c 1 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=128 time=49.3 ms

--- 8.8.8.8 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 49.264/49.264/49.264/0.000 ms
[root@suricata ~]#
```

Suricata logs are stored under **/var/log/suricata** path and we will observe our operations from **fast.log** file. For the live monitoring run "**tail -f /var/log/suricata/fast.log/**" command.

NOTE: if you can't see any changes in the log files make sure the owner of files is **suricata** user.

```
[root@suricata ~]# tail -f /var/log/suricata/fast.log
04/14/2024-06:03:32.918099  [**] [1:1:0] Ping Detected [**] [Classification: (null)] [Priority: 3] {ICMP} 192.168.1.130:8 -> 8.8.8.8:0
04/14/2024-06:03:32.967304  [**] [1:1:0] Ping Detected [**] [Classification: (null)] [Priority: 3] {ICMP} 8.8.8.8:0 -> 192.168.1.130:0
```

IDS rules

You see our service is running properly and the rule could be matched with the signature of sending **ICMP** packet. I installed **DVWA** web application into my host, and want to catch **HTTP** responses which contain **404** error code.

```
[root@suricata ~]# vim /var/lib/suricata/rules/cs301.rules
[root@suricata ~]# systemctl restart suricata.service
[root@suricata ~]#
```

```
#detects HTTP 404 responses which goes from our web application
alert http $HOME_NET any -> $EXTERNAL_NET any (msg:"HTTP 404 Not Found Detected"; flow:established; content:"404"; http_stat_code; sid:2;)
```

Every time after editing rules file, restart the **suricata** service to reload new rules. In order to test this rule I tried to go unexisting path in my web application whcih should response with **not found** error.

```
┌──(yagub㉿kali)-[~]
└─$ curl http://192.168.1.130/admin
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html><head>
<title>404 Not Found</title>
</head><body>
<h1>Not Found</h1>
<p>The requested URL was not found on this server.</p>
</body></html>
```

Again we back to fast.log file and monitor the actions, below, you can see that our rule worked and detected the **404** error response goes from our server to client.

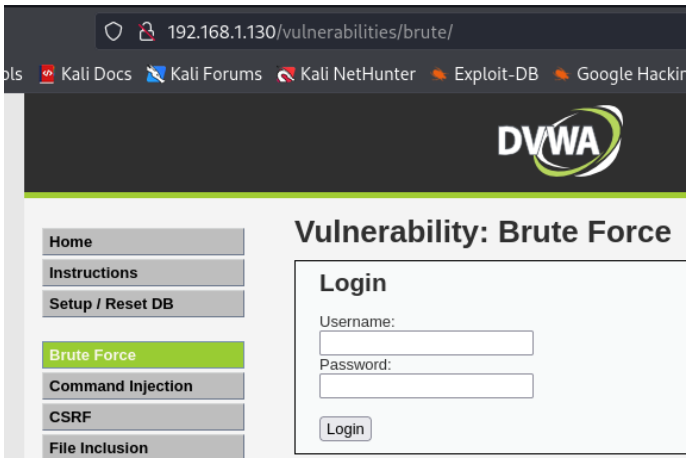
```
[root@suricata ~]# tail -f /var/log/suricata/fast.log

04/14/2024-06:18:54.383229  [**] [1:2:0] HTTP 404 Not Found Detected [**] [Classification: (null)] [Priority: 3] {TCP} 192.168.1.130:80 ->
192.168.1.133:56986
```

In the next rule I said to catch **HTTP** requests comes to specific path in my web application. This path is **/vulnerabilities/brute** in our lab.

```
#detects HTTP requests to the specific location in our web application
alert http $EXTERNAL_NET any -> $HOME_NET any (msg:"HTTP Request Detected To Login Page"; flow:established; content:"/vulnerabilities/
brute/"; http_uri; sid:3;)
```

In order to tests rule I tried to go **http://192.168.1.130/vulnerabilities/brute** address from my client machine.



We can the the alert log in below in the figure.

```
[root@suricata ~]# tail -f /var/log/suricata/fast.log

04/14/2024-07:08:47.606235  [**] [1:3:0] HTTP Request Detected To Login Page [**] [Classification: (null)] [Priority: 3] {TCP} 192.168
.1.133:51432 -> 192.168.1.130:80
```

One of the most essential feature of **IDS** is ability to catch the port scan action. In the rule, I said matching the requests which contains only **SYN** flag in it.

```
#detects port scans with SYN flag
alert tcp $EXTERNAL_NET any -> $HOME_NET any (msg:"Port Scan Detected (flag:SYN)"; flow:to_server,stateless; flags:S; threshold:type t
hreshold, track by_src, count 10, seconds 1; classtype:nmap-scan; sid:4;)
```

I ran **Nmap** port scan in my Kali.

```
┌──(yagub㉿kali)-[~]
└─$ sudo nmap 192.168.1.130 -sS
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-04-14 15:15 +04
Nmap scan report for 192.168.1.130
Host is up (0.0012s latency).
Not shown: 996 closed tcp ports (reset)
PORT     STATE SERVICE
22/tcp   open  ssh
80/tcp   open  http
111/tcp  open  rpcbind
3306/tcp open  mysql
MAC Address: 00:0C:29:43:1C:65 (VMware)

Nmap done: 1 IP address (1 host up) scanned in 12.32 seconds
```

And all requests captures by **IDS** and logged into log files of **Suricata**.

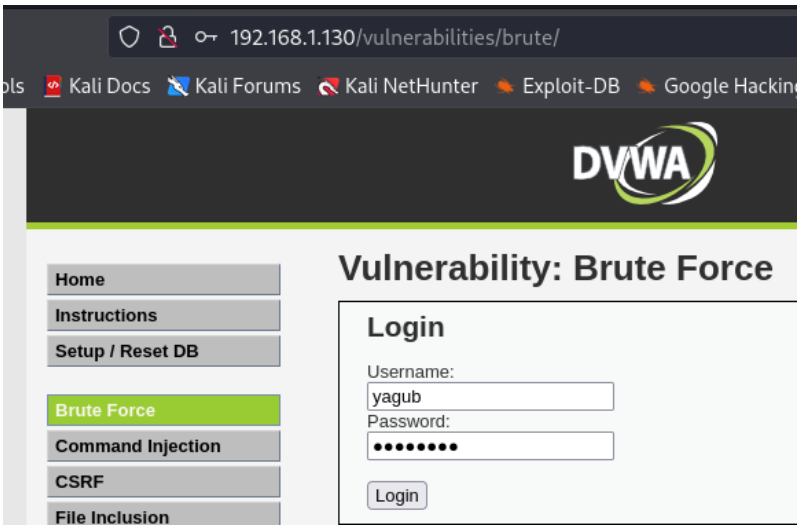
```
[root@suricata ~]# tail -f /var/log/suricata/fast.log

04/14/2024-07:16:01.951352  [**] [1:4:0] Port Scan Detected (flag:SYN) [**] [Classification: (null)] [Priority: 3] {TCP} 192.168.1.133
:47408 -> 192.168.1.130:445
04/14/2024-07:16:01.953403  [**] [1:4:0] Port Scan Detected (flag:SYN) [**] [Classification: (null)] [Priority: 3] {TCP} 192.168.1.133
:47408 -> 192.168.1.130:199
04/14/2024-07:16:01.952568  [**] [1:4:0] Port Scan Detected (flag:SYN) [**] [Classification: (null)] [Priority: 3] {TCP} 192.168.1.133
:47408 -> 192.168.1.130:443
```

This rule capture only incoming **HTTP POST** requests.

```
#detects HTTP POST requests which come to our web application
alert http $EXTERNAL_NET any -> $HOME_NET any (msg:"HTTP POST Request Detected"; flow:to_server,established; content:"POST"; http_meth
od; sid:5;)
```

Again I opened my web application from **Kali** and tried to login which generates a **HTTP POST** request.



This action displayed in the log file.

```
[root@suricata ~]# tail -f /var/log/suricata/fast.log

04/14/2024-15:19:32.673641  [**] [1:3:0] HTTP Request Detected To Login Page [**] [Classification: (null)] [Priority: 3] {TCP} 192.168
.1.133:41084 -> 192.168.1.130:80
04/14/2024-15:19:32.673641  [**] [1:5:0] HTTP POST Request Detected [**] [Classification: (null)] [Priority: 3] {TCP} 192.168.1.133:41
084 -> 192.168.1.130:80
```

If you remember we enabled **GeoIP** feature of **Suricata**, and in this rule, I defined to capture the all network traffic which goes and comes from **US** based **IP** addresses.

```
#detects any network traffic with US
alert ip any any -> any any (msg:"Network Traffic With  US"; geoip:US; sid:6;)
```

In order to test the rule, I send the HTTP GET request to amazon.com address.

```
[root@suricata ~]# curl amazon.com
<html>
<head><title>301 Moved Permanently</title></head>
<body>
<center><h1>301 Moved Permanently</h1></center>
<hr><center>Server</center>
</body>
</html>
[root@suricata ~]#
```

This request is displayed in the **fast.log** file too.

```
[root@suricata ~]# tail -f /var/log/suricata/fast.log

04/14/2024-19:19:58.033341  [**] [1:6:0] Network Traffic With  US [**] [Classification: (null)] [Priority: 3] {TCP} 192.168.1.130:35
042 -> 205.251.242.103:80
04/14/2024-19:19:58.199526  [**] [1:6:0] Network Traffic With  US [**] [Classification: (null)] [Priority: 3] {TCP} 205.251.242.103:
80 -> 192.168.1.130:35042
04/14/2024-19:19:58.200303  [**] [1:6:0] Network Traffic With  US [**] [Classification: (null)] [Priority: 3] {TCP} 192.168.1.130:35
```

IPS rules

Suricata works in **IDS** mode as default, but we will enable **IPS** mode of it in this capture. Open **/etc/sysconfig/suricata** file and comment in the **OPTIONS** line to disable **IDS** mode, add new line as displayed in the figure.

```
# Add options to be passed to the daemon
OPTIONS="-i ens160 --user suricata "
```

```
# Add options to be passed to the daemon
#OPTIONS="-i ens160 --user suricata "
OPTIONS="-q 0 -vvv --user suricata"
```

In order to use **Suricata** in **IPS** mode, we need to stop **suricata.service** and enable **Iptables** tool as shown in the figure by running two commands, "**iptables -I INPUT -j NFQUEUE**" and "**iptables -I OUTPUT -j NFQUEUE**".

```
[root@suricata ~]# systemctl stop suricata.service
[root@suricata ~]# vim /etc/sysconfig/suricata
[root@suricata ~]#  iptables -I INPUT -j NFQUEUE
[root@suricata ~]#  iptables -I OUTPUT -j NFQUEUE
[root@suricata ~]# systemctl start suricata.service
[root@suricata ~]#
```

After starting **suricata.service** our **IPS** will be active and running. As you remember we used **ICMP** packet to test our rule, and this time we will change the action from **alert** to **drop**. This rule will block **ICMP** packets goes from any port of our host to **8.8.8.8** address's any port.

```
drop icmp $HOME_NET any -> 8.8.8.8 any (msg:"ICMP Packet Blocked"; sid: 1;)
```

When we try to send ping to defined address it will be timed out.

```
[root@suricata ~]# ping -c 1 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.

--- 8.8.8.8 ping statistics ---
1 packets transmitted, 0 received, 100% packet loss, time 0ms

[root@suricata ~]#
```

```
[root@suricata ~]# tail -f /var/log/suricata/fast.log

04/14/2024-16:36:55.181603  [Drop] [**] [1:1:0] ICMP Packet Blocked [**] [Classification: (null)] [Priority: 3] {ICMP} 192.168.1.130:8
 -> 8.8.8.8:0
```

This rule blocks all **HTTP** response contains **404** error code, if the response's intensity is 100 per second.

```
#drops HTTP requests with 404 status code
drop http $HOME_NET any -> $EXTERNAL_NET any (msg:"Possile URI Brute-Force Attack Blocked"; flow:established; content:"404"; http_stat
_code; threshold: type limit, track by_src, count 100, seconds 1; sid:11;)
```

In order to tests it, I simulated a directory fuzzing attack from my Kali. You can see there is no result for attack

```
┌──(yagub㉿kali)-[~]
└─$ sudo gobuster dir -u http://192.168.1.130/ -w /usr/share/wordlists/rockyou.txt
===============================================================
Gobuster v3.6
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     http://192.168.1.130/
[+] Method:                  GET
[+] Threads:                 10
[+] Wordlist:                /usr/share/wordlists/rockyou.txt
[+] Negative Status codes:   404
[+] User Agent:              gobuster/3.6
[+] Timeout:                 10s
===============================================================
Starting gobuster in directory enumeration mode
===============================================================

Error: error on running gobuster: Get "http://192.168.1.130/c33030b6-7e77-4cc8-a0d3-d5743d8a5dbf": context deadline exceeded (C
lient.Timeout exceeded while awaiting headers)
```

```
[root@suricata ~]# tail -f /var/log/suricata/fast.log

04/14/2024-16:47:45.402455  [Drop] [**] [1:11:0] Possile URI Brute-Force Attack Blocked [**] [Classification: (null)] [Priority: 3] {T
CP} 192.168.1.130:80 -> 192.168.1.133:58916
```

In **IDS** mode, we detected port scan action, but this time we will block it by replacing **alert** to **drop** in the rule.

```
#drops port scan activities whose intensity is 100 requests per second
drop tcp $EXTERNAL_NET any -> $HOME_NET any (msg:"Port Scan Activity Blocked"; flags:S; flow: to_server, stateless; threshold: type
limit, track by_src, count 100, seconds 1; sid:12;)
```

```
┌──(yagub㉿kali)-[~]
└─$ sudo nmap 192.168.1.130 -sS
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-04-14 16:51 +04
Nmap scan report for 192.168.1.130
Host is up (0.00068s latency).
All 1000 scanned ports on 192.168.1.130 are in ignored states.
Not shown: 1000 filtered tcp ports (no-response)
MAC Address: 00:0C:29:43:1C:65 (VMware)

Nmap done: 1 IP address (1 host up) scanned in 33.44 seconds
```

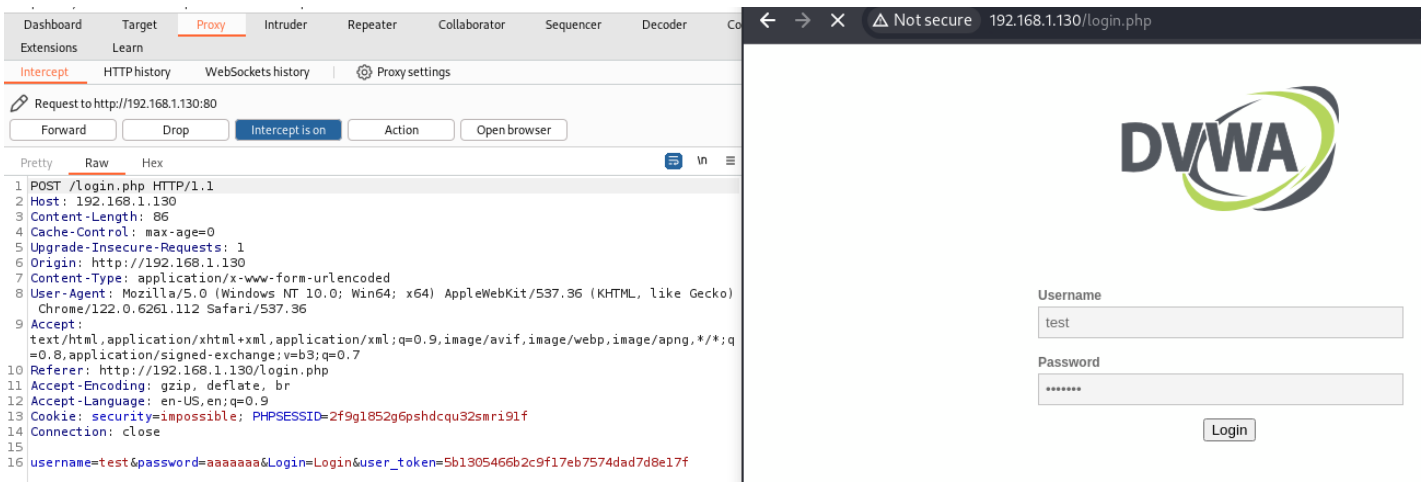
```
[root@suricata ~]# tail -f /var/log/suricata/fast.log

04/14/2024-16:51:12.442317  [Drop] [**] [1:12:0] Port Scan Activity Blocked [**] [Classification: (null)] [Priority: 3] {TCP} 192.16
8.1.133:34567 -> 192.168.1.130:21
04/14/2024-16:51:12.442322  [Drop] [**] [1:12:0] Port Scan Activity Blocked [**] [Classification: (null)] [Priority: 3] {TCP} 192.16
8.1.133:34567 -> 192.168.1.130:8888
04/14/2024-16:51:12.442307  [Drop] [**] [1:12:0] Port Scan Activity Blocked [**] [Classification: (null)] [Priority: 3] {TCP} 192.16
8.1.133:34567 -> 192.168.1.130:8080
04/14/2024-16:51:12.442315  [Drop] [**] [1:12:0] Port Scan Activity Blocked [**] [Classification: (null)] [Priority: 3] {TCP} 192.16
8.1.133:34567 -> 192.168.1.130:23
```

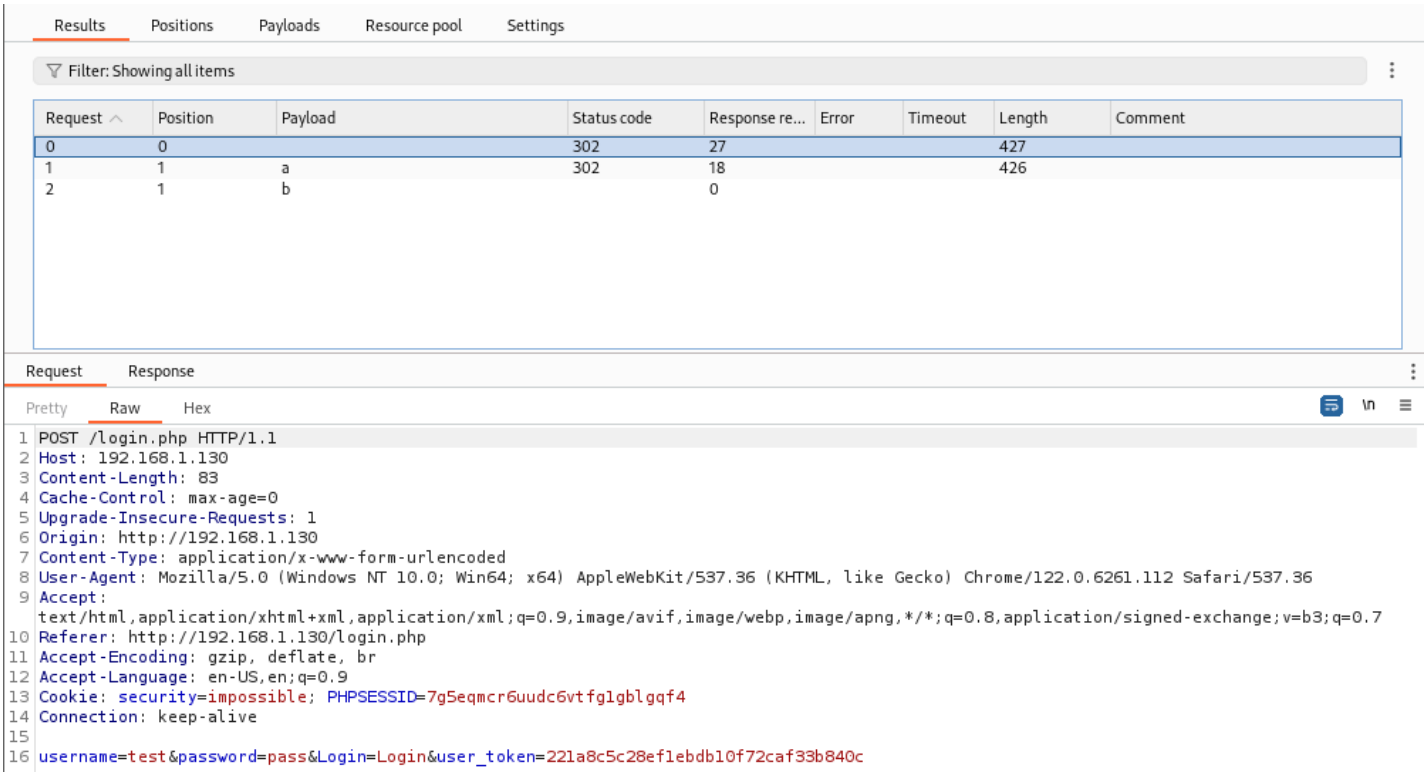
In the rule below, I defined block **HTTP POST** requests if its intensity is over 3 requests per second.

```
#drops HTTP POST requests when intensity is over 3 requests per seconds
drop http $EXTERNAL_NET any -> $HOME_NET any (msg:"Possible Login Brute-Force Attack Blocked"; flow:to_server,established; content:"
POST"; http_method; threshold: type threshold, track by_src,  count 3, seconds 1; sid:13;)
```

For this purpose I simulated a brute-force attack with Burpsuite tool on my web application.



Below in the figures, you can see that our brute-force attack became unsuccessful and stopped in third try.




```
[root@suricata ~]# tail -f /var/log/suricata/fast.log

04/14/2024-18:54:42.165975  [Drop] [**] [1:13:0] Possible Login Brute-Force Attack Blocked [**] [Classification: (null)] [Priority:
3] {TCP} 192.168.1.133:39348 -> 192.168.1.130:80
```

In the last rule, I blocked all network traffic with US based IP addresses.

```
#drops any network traffic with US
drop ip any any -> any any (msg:"Network Traffic With  US Blocked"; geoip:US; sid:14;)
```

As you see, when I try to go **amazon.com** address again the connection was unsuccessful because of **IPS**.

```
[root@suricata ~]# curl amazon.com
curl: (7) Failed to connect to amazon.com port 80: Connection timed out
[root@suricata ~]#
```

```
[root@suricata ~]# tail -f /var/log/suricata/fast.log

04/14/2024-19:26:28.484943  [Drop] [**] [1:14:0] Network Traffic With  US Blocked [**] [Classification: (null)] [Priority: 3] {TCP} 192.168.1.130
:58172 -> 52.94.236.248:80
04/14/2024-19:26:29.486033  [Drop] [**] [1:14:0] Network Traffic With  US Blocked [**] [Classification: (null)] [Priority: 3] {TCP} 192.168.1.130
:58172 -> 52.94.236.248:80
04/14/2024-19:26:31.533579  [Drop] [**] [1:14:0] Network Traffic With  US Blocked [**] [Classification: (null)] [Priority: 3] {TCP} 192.168.1.130
```