Wazuh – Suricata IDS

Network Intrusion Detection

Lab Created By: MUHAMMAD MOIZ UD DIN RAFAY

Follow Me: linkedin.com/in/moizuddinrafay

Integrating Wazuh and Suricata for Enhanced Security Monitoring

Wazuh and **Suricata** are powerful open-source security tools that, when integrated, provide a robust and comprehensive security monitoring solution. This integration enhances the visibility and detection capabilities of an organization's security infrastructure by combining network and host-based monitoring.

Overview of the Integration

1. **Data Collection and Analysis**:

Suricata acts as a network-based intrusion detection and prevention system (IDS/IPS) that monitors network traffic for suspicious activity. It inspects incoming and outgoing packets, identifies potential threats using signature-based detection and anomaly detection techniques, and generates alerts for detected threats.

Wazuh serves as a centralized security management platform that collects logs from various sources, including Suricata. It processes these logs, correlates them with other security events, and provides comprehensive analysis and reporting.

2. **Unified Threat Detection**:

Suricata's network traffic alerts are sent to Wazuh for aggregation and correlation. Wazuh can then combine these network-based alerts with logs and events from endpoints and other sources, providing a holistic view of the security posture.

This unified approach enables security teams to detect complex threats that span both network and host environments, improving the overall detection accuracy and reducing false positives.

3. **Incident Response and Forensics**:

By leveraging Suricata's detailed network traffic analysis and Wazuh's extensive log management capabilities, organizations can conduct thorough investigations into security incidents.

Wazuh provides tools for incident response, such as real-time alerting, automated actions, and detailed reports, which are enriched with data from Suricata. This facilitates a faster and more effective response to threats.

4. **Scalability and Flexibility**:

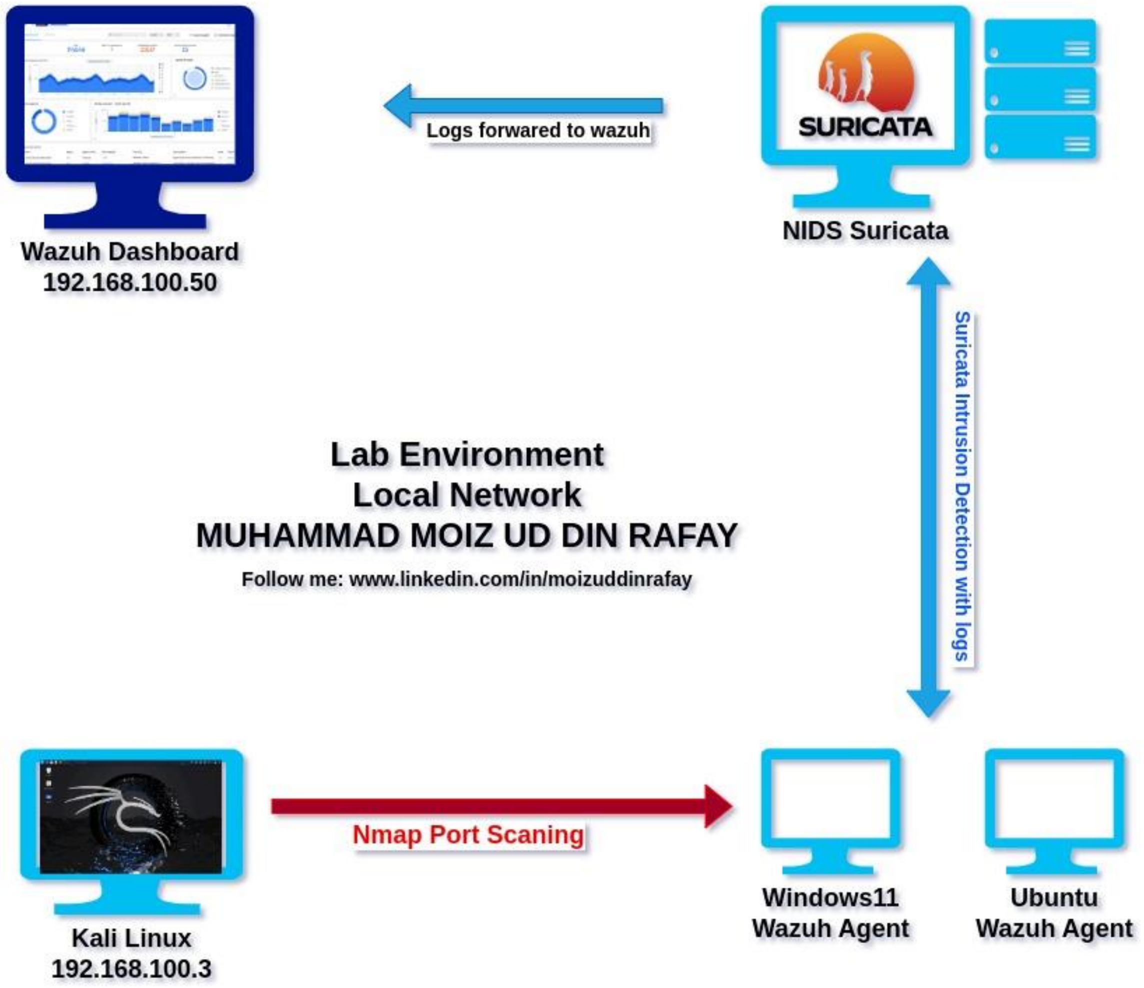
Both Wazuh and Suricata are designed to scale horizontally, making them suitable for deployment in environments of any size. Their integration allows organizations to expand their security monitoring capabilities without significant reconfiguration.

The flexibility of both tools ensures they can be tailored to specific organizational needs, allowing for customized rules, alerting mechanisms, and reporting formats.

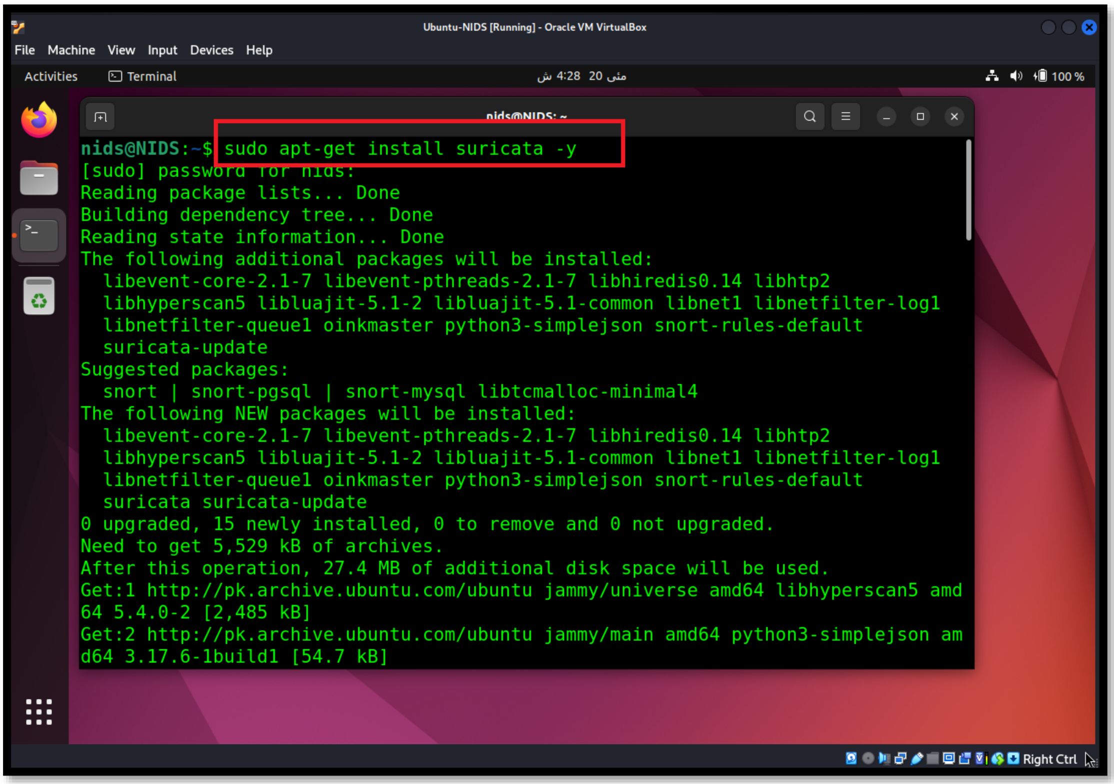
5. **Enhanced Visibility and Correlation**:

The integration allows for enhanced visibility into both network and endpoint activities. Wazuh's dashboard can display Suricata alerts alongside other security events, providing a single pane of glass for security monitoring.

Correlating Suricata's network-based alerts with Wazuh's host-based data can uncover sophisticated attack patterns and help identify the root cause of security incidents.

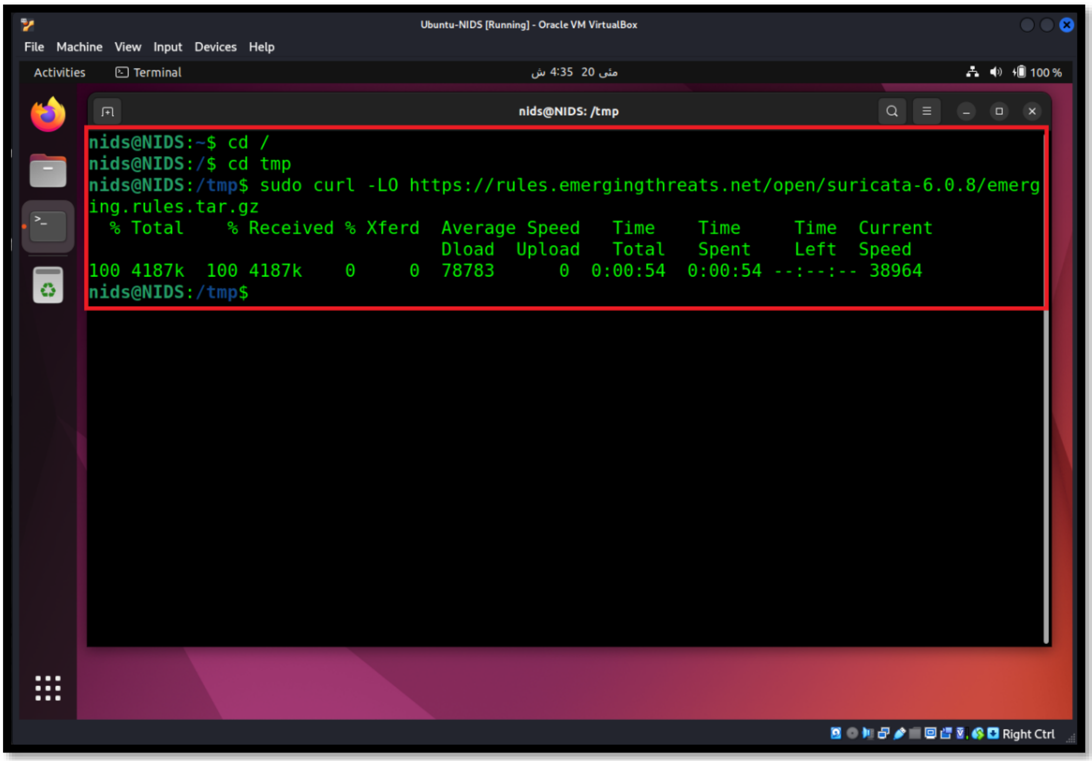

Logs forwarded to wazuh
SURICATA
NIDS Suricata
Wazuh Dashboard
192.168.100.50
Suricata Intrusion Detection with logs
Lab Environment
Local Network
MUHAMMAD MOIZ UD DIN RAFAY
Follow me: www.linkedin.com/in/moizuddinrafay
Nmap Port Scaning
Kali Linux
192.168.100.3
Windows11
Wazuh Agent
Ubuntu
Wazuh Agent

I installed new “Ubuntu-NIDS” virtual machine in my network and installing Suricata IDS on new machine.
Command: sudo apt install suricata -y


Ubuntu-NIDS [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities
Terminal
ش 4:28 20 مئی
100 %
nids@NIDS: ~
Right Ctrl

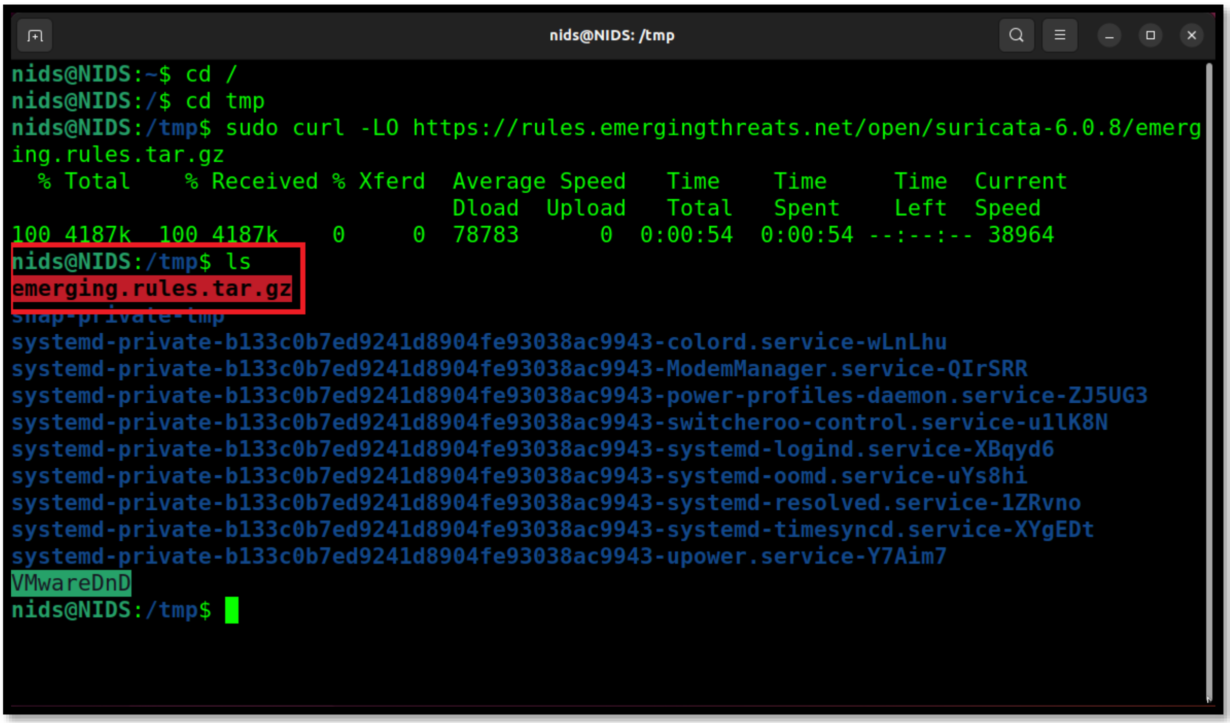
```
nids@NIDS:~$ sudo apt-get install suricata -y
[sudo] passwora for nids:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  libevent-core-2.1-7 libevent-pthreads-2.1-7 libhiredis0.14 libhtp2
  libhyperscan5 libluajit-5.1-2 libluajit-5.1-common libnet1 libnetfilter-log1
  libnetfilter-queue1 oinkmaster python3-simplejson snort-rules-default
  suricata-update
Suggested packages:
  snort | snort-pgsql | snort-mysql libtcmalloc-minimal4
The following NEW packages will be installed:
  libevent-core-2.1-7 libevent-pthreads-2.1-7 libhiredis0.14 libhtp2
  libhyperscan5 libluajit-5.1-2 libluajit-5.1-common libnet1 libnetfilter-log1
  libnetfilter-queue1 oinkmaster python3-simplejson snort-rules-default
  suricata suricata-update
0 upgraded, 15 newly installed, 0 to remove and 0 not upgraded.
Need to get 5,529 kB of archives.
After this operation, 27.4 MB of additional disk space will be used.
Get:1 http://pk.archive.ubuntu.com/ubuntu jammy/universe amd64 libhyperscan5 amd
64 5.4.0-2 [2,485 kB]
Get:2 http://pk.archive.ubuntu.com/ubuntu jammy/main amd64 python3-simplejson am
d64 3.17.6-1build1 [54.7 kB]
```

After installing we have to download suricata rules, follow as shown.


Ubuntu-NIDS [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities
Terminal
ش 4:35 20 مئی
100 %
nids@NIDS: /tmp
Right Ctrl

```
nids@NIDS:~$ cd /
nids@NIDS:/$ cd tmp
nids@NIDS:/tmp$ sudo curl -LO https://rules.emergingthreats.net/open/suricata-6.0.8/emerg
ing.rules.tar.gz
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100 4187k  100 4187k    0     0  78783      0  0:00:54  0:00:54 --:--:-- 38964
nids@NIDS:/tmp$
```

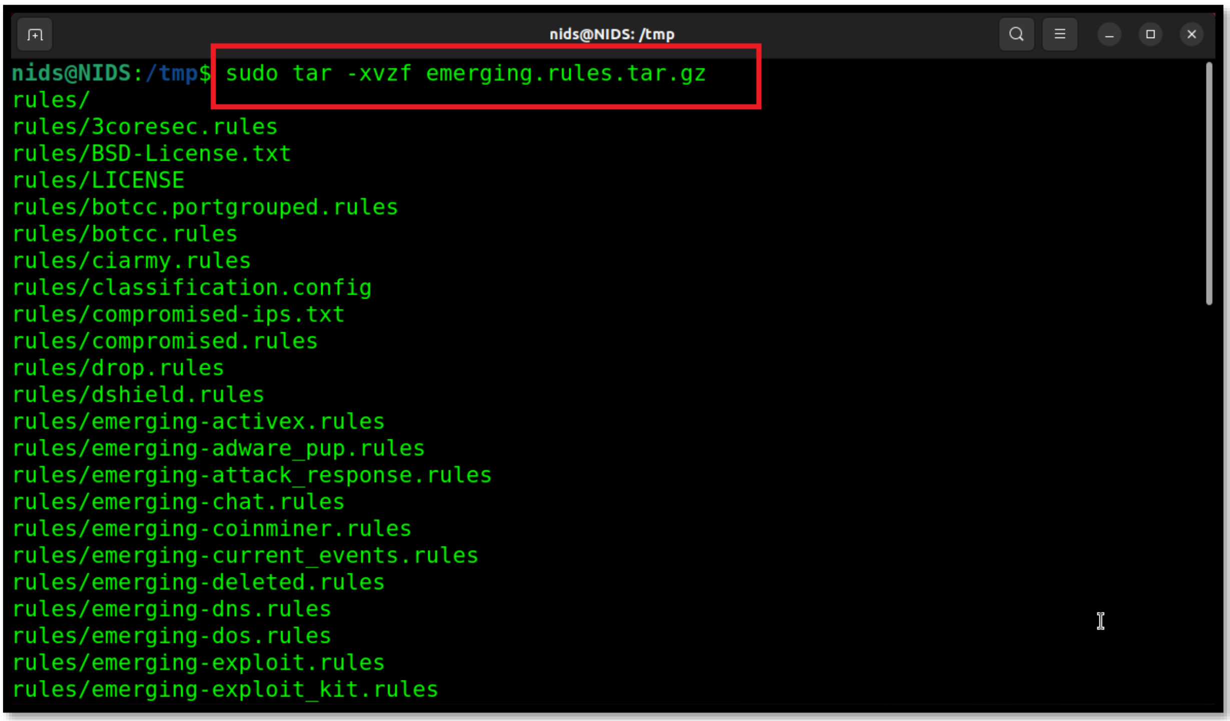
Here is the downloaded file of suricata rules “emerging.rules.tar.gz”



```
nids@NIDS:~$ cd /
nids@NIDS:/$ cd tmp
nids@NIDS:/tmp$ sudo curl -LO https://rules.emergingthreats.net/open/suricata-6.0.8/emerg
ing.rules.tar.gz
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100 4187k  100 4187k    0     0  78783      0  0:00:54  0:00:54 --:--:-- 38964
nids@NIDS:/tmp$ ls
emerging.rules.tar.gz
snap-private-tmp
systemd-private-b133c0b7ed9241d8904fe93038ac9943-colord.service-wLnLhu
systemd-private-b133c0b7ed9241d8904fe93038ac9943-ModemManager.service-QIrSRR
systemd-private-b133c0b7ed9241d8904fe93038ac9943-power-profiles-daemon.service-ZJ5UG3
systemd-private-b133c0b7ed9241d8904fe93038ac9943-switcheroo-control.service-u1lK8N
systemd-private-b133c0b7ed9241d8904fe93038ac9943-systemd-logind.service-XBqyd6
systemd-private-b133c0b7ed9241d8904fe93038ac9943-systemd-oomd.service-uYs8hi
systemd-private-b133c0b7ed9241d8904fe93038ac9943-systemd-resolved.service-1ZRvno
systemd-private-b133c0b7ed9241d8904fe93038ac9943-systemd-timesyncd.service-XYgEDt
systemd-private-b133c0b7ed9241d8904fe93038ac9943-upower.service-Y7Aim7
VMwareDnD
nids@NIDS:/tmp$
```

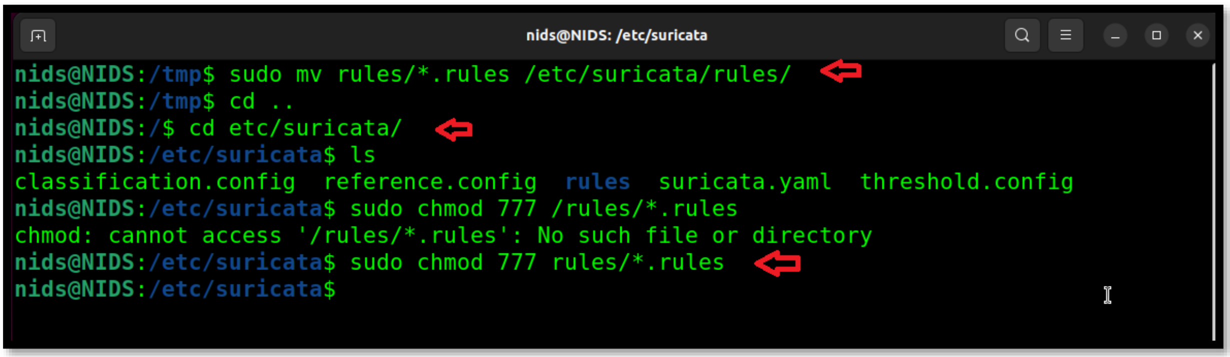
Now we have to extract tar.gz file

Command: sudo tar -xvzf emerging.rules.tar.gz



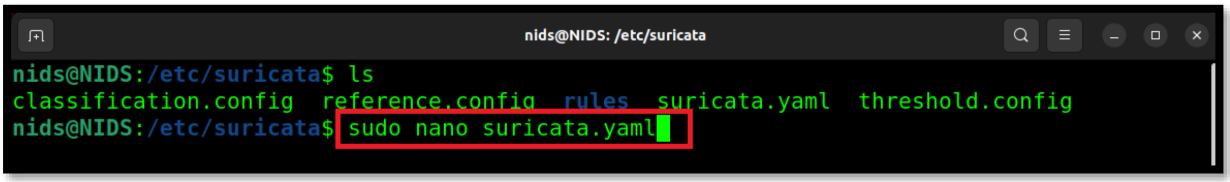
```
nids@NIDS:/tmp$ sudo tar -xvzf emerging.rules.tar.gz
rules/
rules/3coresec.rules
rules/BSD-License.txt
rules/LICENSE
rules/botcc.portgrouped.rules
rules/botcc.rules
rules/ciarmy.rules
rules/classification.config
rules/compromised-ips.txt
rules/compromised.rules
rules/drop.rules
rules/dshield.rules
rules/emerging-activex.rules
rules/emerging-adware_pup.rules
rules/emerging-attack_response.rules
rules/emerging-chat.rules
rules/emerging-coinminer.rules
rules/emerging-current_events.rules
rules/emerging-deleted.rules
rules/emerging-dns.rules
rules/emerging-dos.rules
rules/emerging-exploit.rules
rules/emerging-exploit_kit.rules
```

Now we have to move rules into suricata rules directory and give the permission to all rules. Follow as shown in figure.



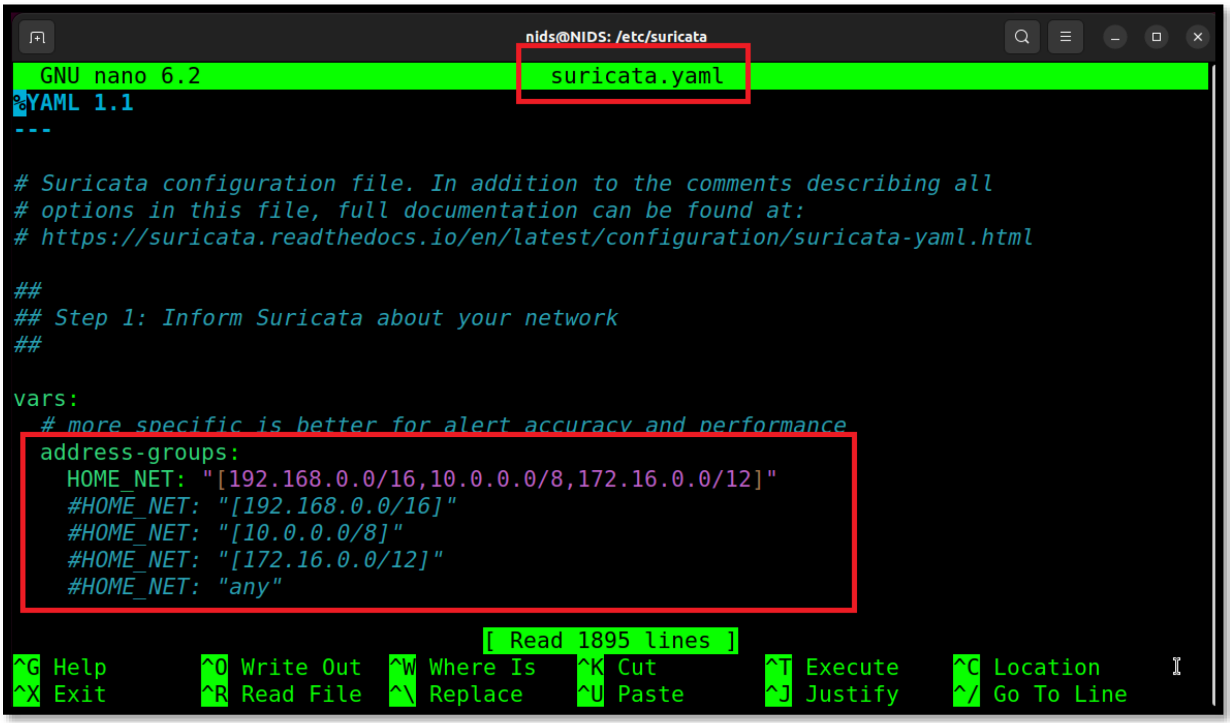
```
nids@NIDS:/tmp$ sudo mv rules/*.rules /etc/suricata/rules/
nids@NIDS:/tmp$ cd ..
nids@NIDS:/$ cd etc/suricata/
nids@NIDS:/etc/suricata$ ls
classification.config  reference.config  rules  suricata.yaml  threshold.config
nids@NIDS:/etc/suricata$ sudo chmod 777 /rules/*.rules
chmod: cannot access '/rules/*.rules': No such file or directory
nids@NIDS:/etc/suricata$ sudo chmod 777 rules/*.rules
nids@NIDS:/etc/suricata$
```

Now we have to configure suricata according to our network setting.



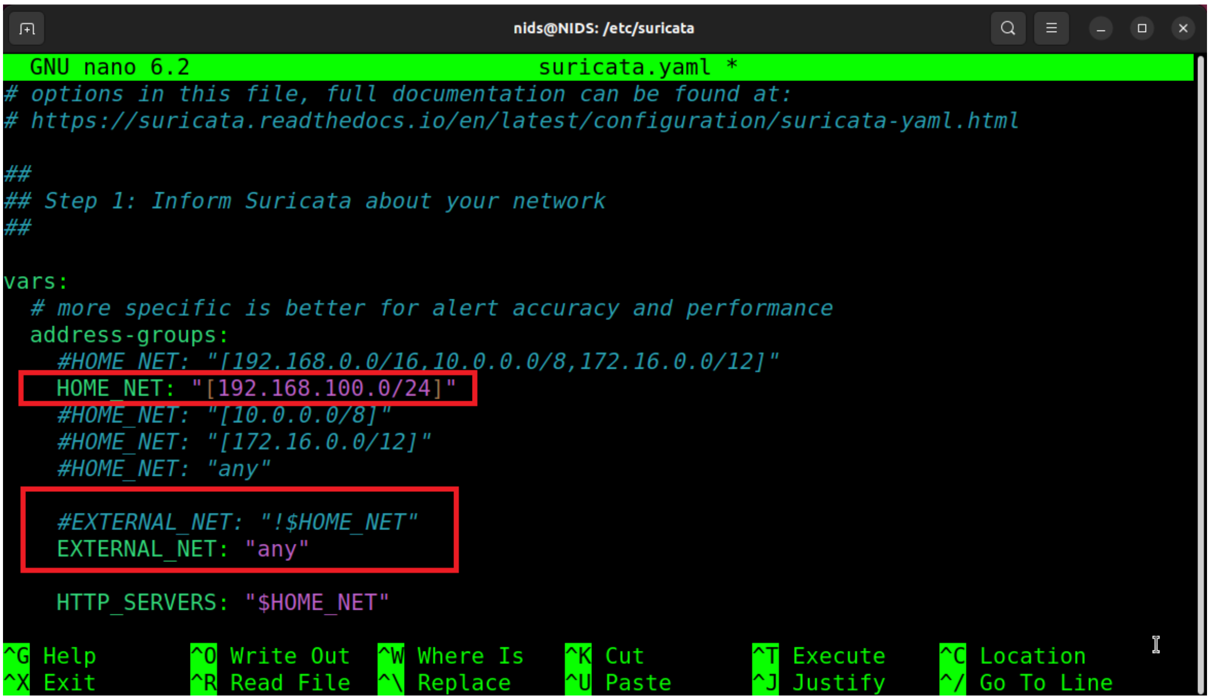
```
nids@NIDS:/etc/suricata$ ls
classification.config  reference.config  rules  suricata.yaml  threshold.config
nids@NIDS:/etc/suricata$ sudo nano suricata.yaml
```

Here is “suricata.yaml” configuration file. We have to Edit “HOME_NET” and “EXTERNAL_NET” follow same as shown in figure.



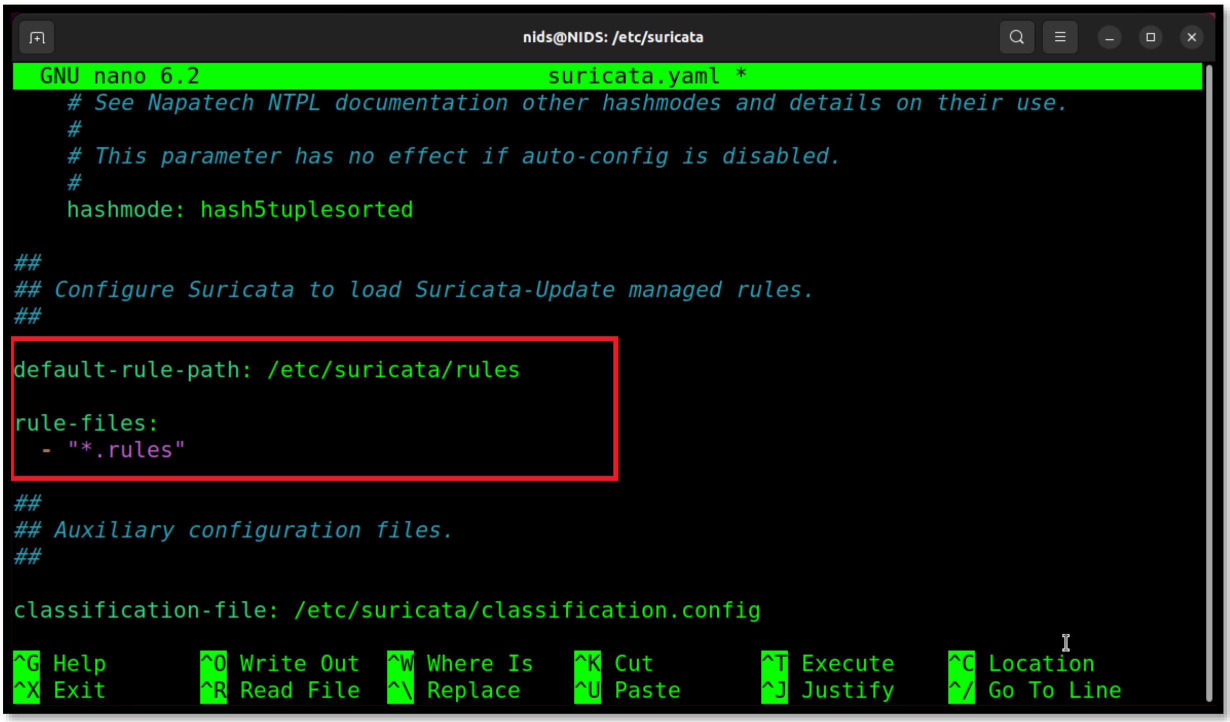
```
  GNU nano 6.2                    suricata.yaml
%YAML 1.1
---

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

##
## 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"
                              [ Read 1895 lines ]
^G Help          ^O Write Out     ^W Where Is      ^K Cut           ^T Execute       ^C Location
^X Exit          ^R Read File     ^\ Replace       ^U Paste         ^J Justify       ^/ Go To Line
```

Make sure you are following according to your own network.



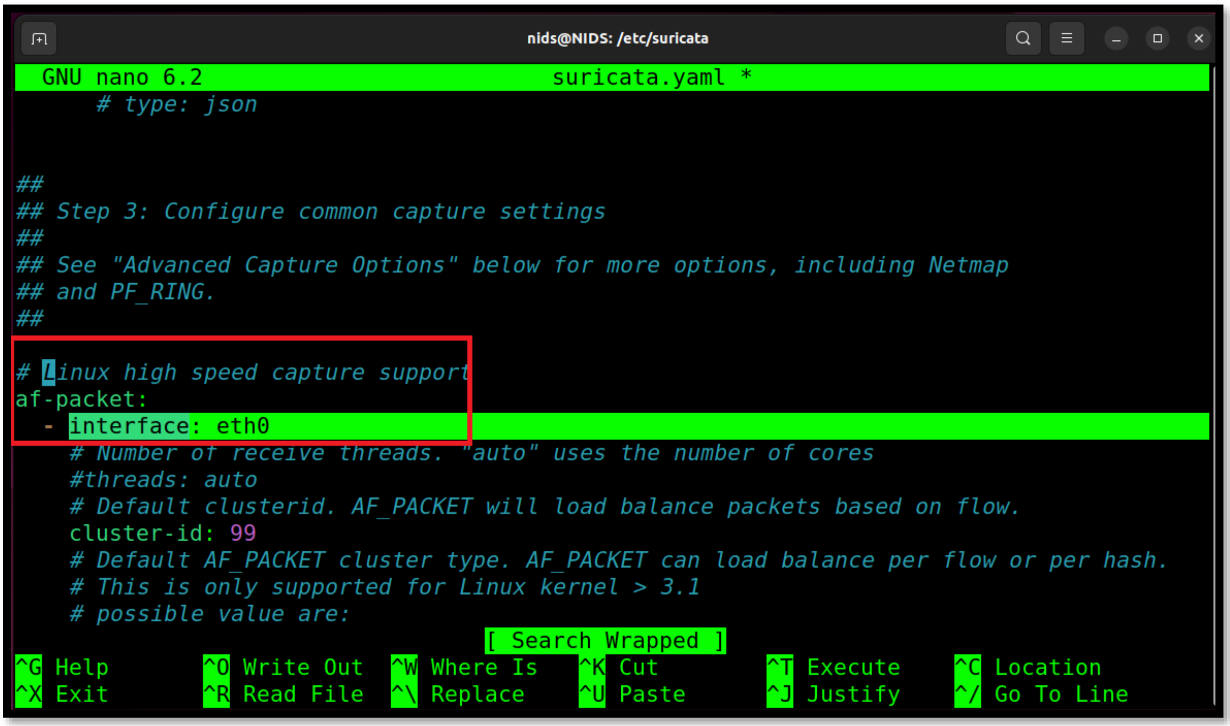
```
  GNU nano 6.2                                      suricata.yaml *
# options in this file, full documentation can be found at:
# https://suricata.readthedocs.io/en/latest/configuration/suricata-yaml.html

##
## 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: "[192.168.100.0/24]"
    #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"

^G Help          ^O Write Out     ^W Where Is      ^K Cut           ^T Execute       ^C Location
^X Exit          ^R Read File     ^\ Replace       ^U Paste         ^J Justify       ^/ Go To Line
```

Now scroll down and edit “rule-files:” section. Follow same as shown in figure.



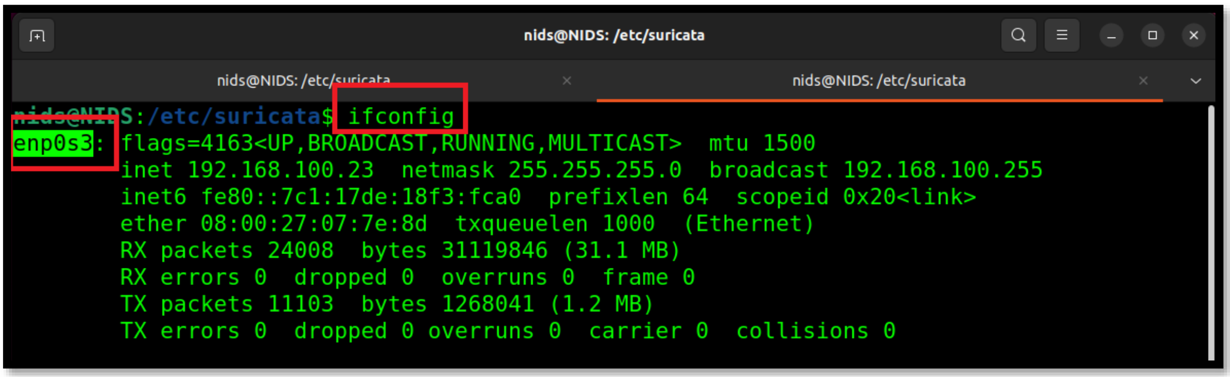
```
  GNU nano 6.2                                      suricata.yaml *
    # See Napatech NTPL documentation other hashmodes and details on their use.
    #
    # This parameter has no effect if auto-config is disabled.
    #
    hashmode: hash5tuplesorted

##
## Configure Suricata to load Suricata-Update managed rules.
##

default-rule-path: /etc/suricata/rules

rule-files:
  - "*.rules"

##
## Auxiliary configuration files.
##

classification-file: /etc/suricata/classification.config

^G Help          ^O Write Out     ^W Where Is      ^K Cut           ^T Execute       ^C Location
^X Exit          ^R Read File     ^\ Replace       ^U Paste         ^J Justify       ^/ Go To Line
```

Now edit the interface configuration

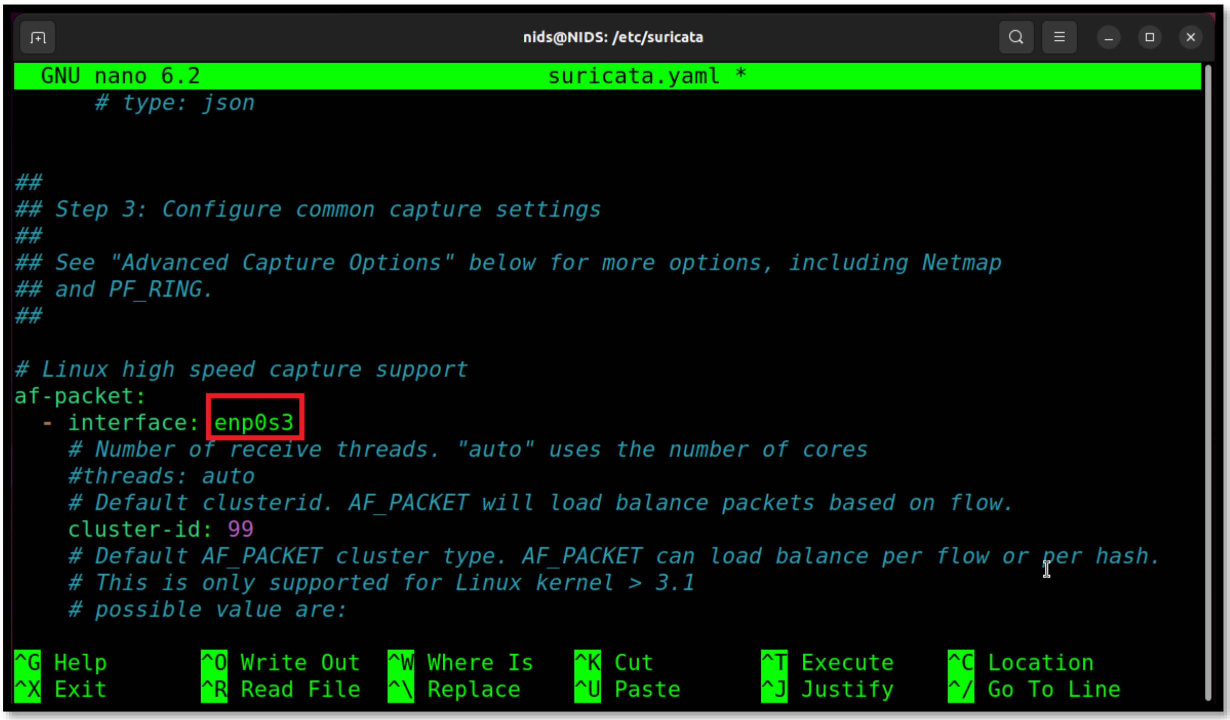


```
  GNU nano 6.2                          suricata.yaml *
      # type: json


##
## Step 3: Configure common capture settings
##
## See "Advanced Capture Options" below for more options, including Netmap
## and PF_RING.
##

# Linux high speed capture support
af-packet:
  - interface: eth0
    # Number of receive threads. "auto" uses the number of cores
    #threads: auto
    # Default clusterid. AF_PACKET will load balance packets based on flow.
    cluster-id: 99
    # Default AF_PACKET cluster type. AF_PACKET can load balance per flow or per hash.
    # This is only supported for Linux kernel > 3.1
    # possible value are:
                                   [ Search Wrapped ]
^G Help         ^O Write Out    ^W Where Is     ^K Cut          ^T Execute      ^C Location
^X Exit         ^R Read File    ^\ Replace      ^U Paste        ^J Justify      ^/ Go To Line
```

Find network interface in my case it’s “enp0s3”
Command: ifconfig



```
nids@NIDS:/etc/suricata$ ifconfig
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.100.23  netmask 255.255.255.0  broadcast 192.168.100.255
        inet6 fe80::7c1:17de:18f3:fca0  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:07:7e:8d  txqueuelen 1000  (Ethernet)
        RX packets 24008  bytes 31119846 (31.1 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 11103  bytes 1268041 (1.2 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

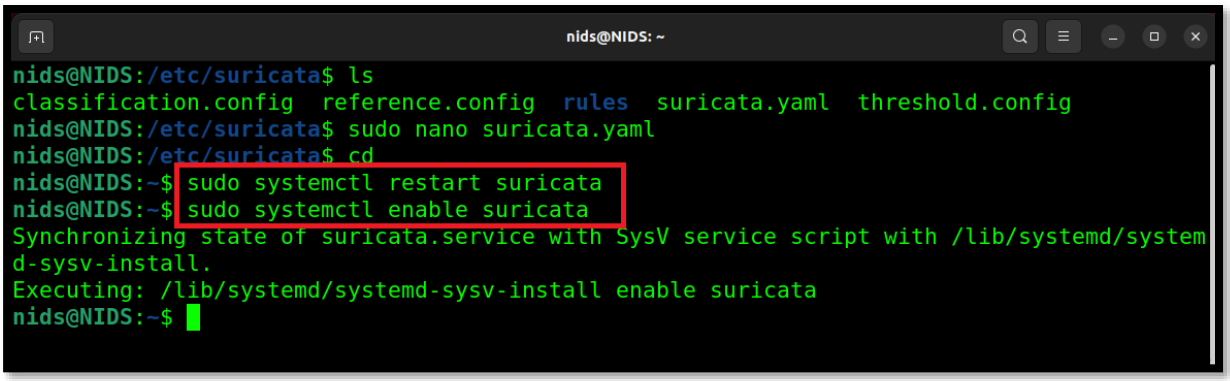


```
  GNU nano 6.2                        suricata.yaml *
      # type: json


##
## Step 3: Configure common capture settings
##
## See "Advanced Capture Options" below for more options, including Netmap
## and PF_RING.
##

# Linux high speed capture support
af-packet:
  - interface: enp0s3
    # Number of receive threads. "auto" uses the number of cores
    #threads: auto
    # Default clusterid. AF_PACKET will load balance packets based on flow.
    cluster-id: 99
    # Default AF_PACKET cluster type. AF_PACKET can load balance per flow or per hash.
    # This is only supported for Linux kernel > 3.1
    # possible value are:

^G Help        ^O Write Out   ^W Where Is    ^K Cut         ^T Execute     ^C Location
^X Exit        ^R Read File   ^\ Replace     ^U Paste       ^J Justify     ^/ Go To Line
```

Now save the configuration and restart suricata.

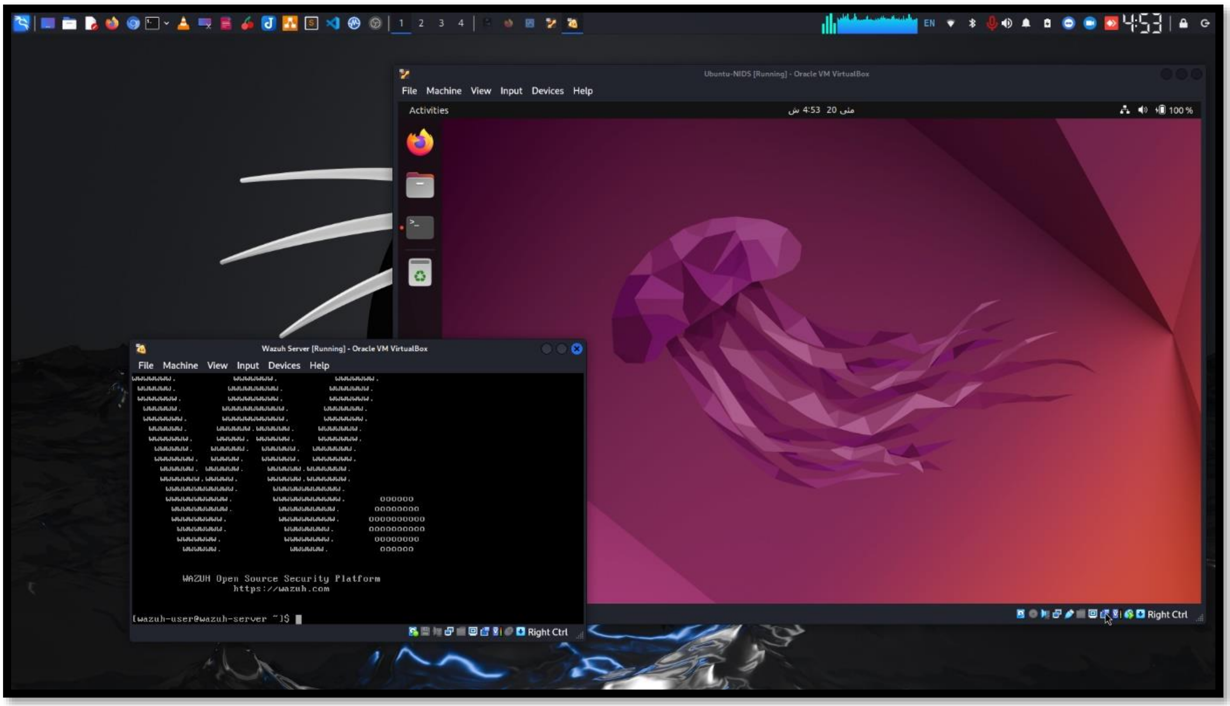
Command: sudo systemctl restart suricata

Command: sudo systemctl enable suricata (For enable at boot time)



```
nids@NIDS:/etc/suricata$ ls
classification.config  reference.config  rules  suricata.yaml  threshold.config
nids@NIDS:/etc/suricata$ sudo nano suricata.yaml
nids@NIDS:/etc/suricata$ cd
nids@NIDS:~$ sudo systemctl restart suricata
nids@NIDS:~$ sudo systemctl enable suricata
Synchronizing state of suricata.service with SysV service script with /lib/systemd/system
d-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable suricata
nids@NIDS:~$ 
```

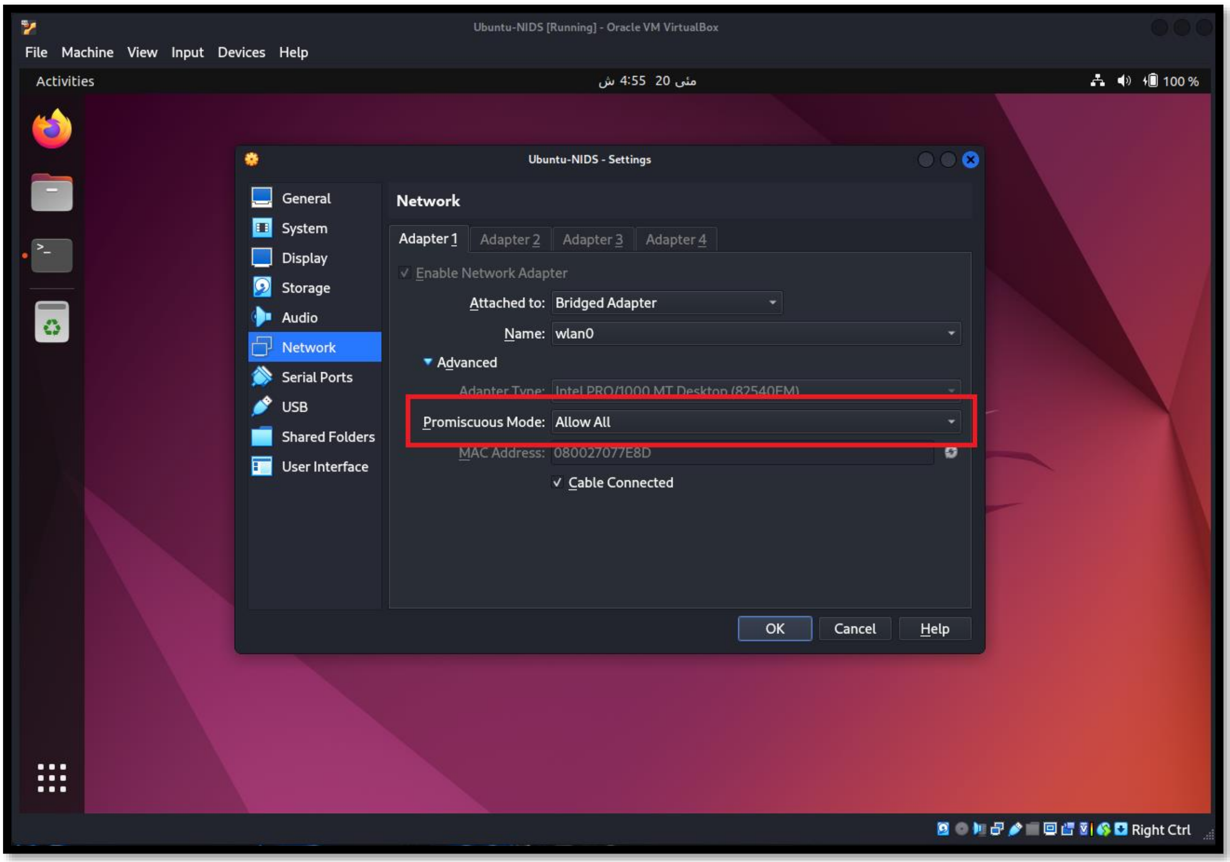
Now open Wazuh and configure agent.



Wazuh Server [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Ubuntu-NIDS [Running] - Oracle VM VirtualBox
Activities
Right Ctrl

```
      WAZUH Open Source Security Platform
               https://wazuh.com

[wazuh-user@wazuh-server ~]$
```

Before moving on wazuh we have to enable “Promiscuous Mode” Allow All.



Network

☑ Enable Network Adapter

Attached to: Bridged Adapter

Name: wlan0

Advanced

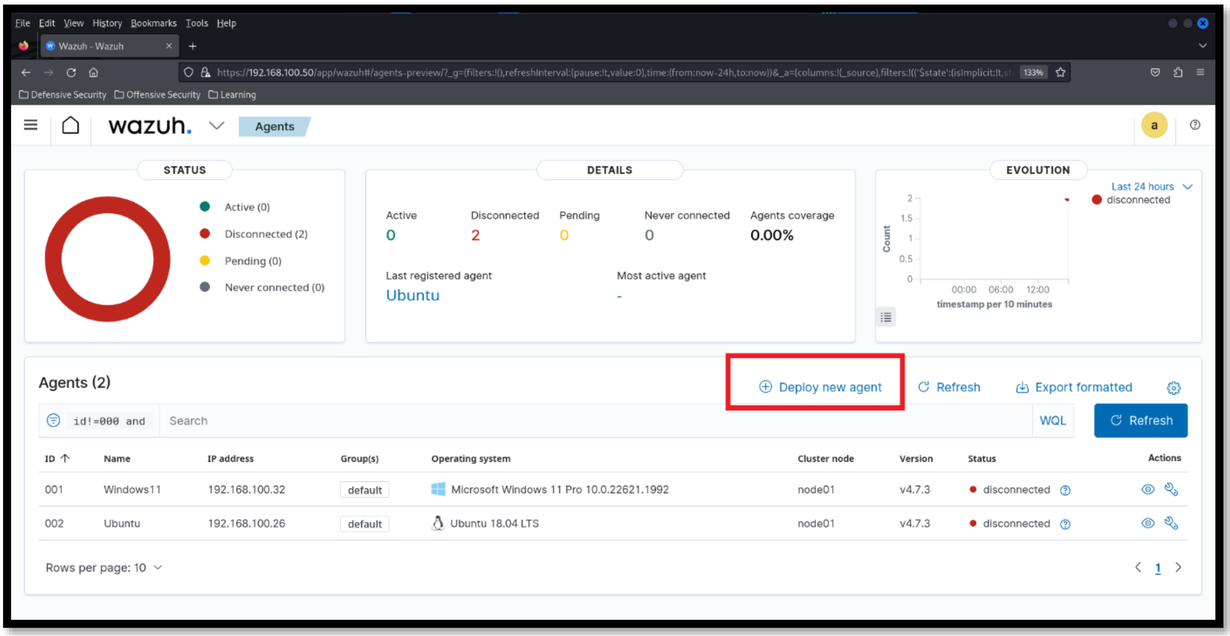
Adapter Type: Intel PRO/1000 MT Desktop (82540EM)

Promiscuous Mode: Allow All

MAC Address: 080027077E8D

☑ Cable Connected

Go to Wazuh Dashboard and click on “Deploy new agent”


File Edit View History Bookmarks Tools Help
Wazuh - Wazuh
https://192.168.100.50/app/wazuh#/agents-preview/?_g=(filters:!(),refreshInterval:(pause:!t,value:0),time:(from:now-24h,to:now))&_a=(columns:!(_source),filters:!(('$state':(isImplicit:!t,st
Defensive Security
Offensive Security
Learning
wazuh.
Agents
STATUS
Active (0)
Disconnected (2)
Pending (0)
Never connected (0)
DETAILS
EVOLUTION
Last 24 hours
disconnected
Count
timestamp per 10 minutes

Active	Disconnected	Pending	Never connected	Agents coverage
0	2	0	0	0.00%

Last registered agent: Ubuntu

Most active agent: -

Agents (2)

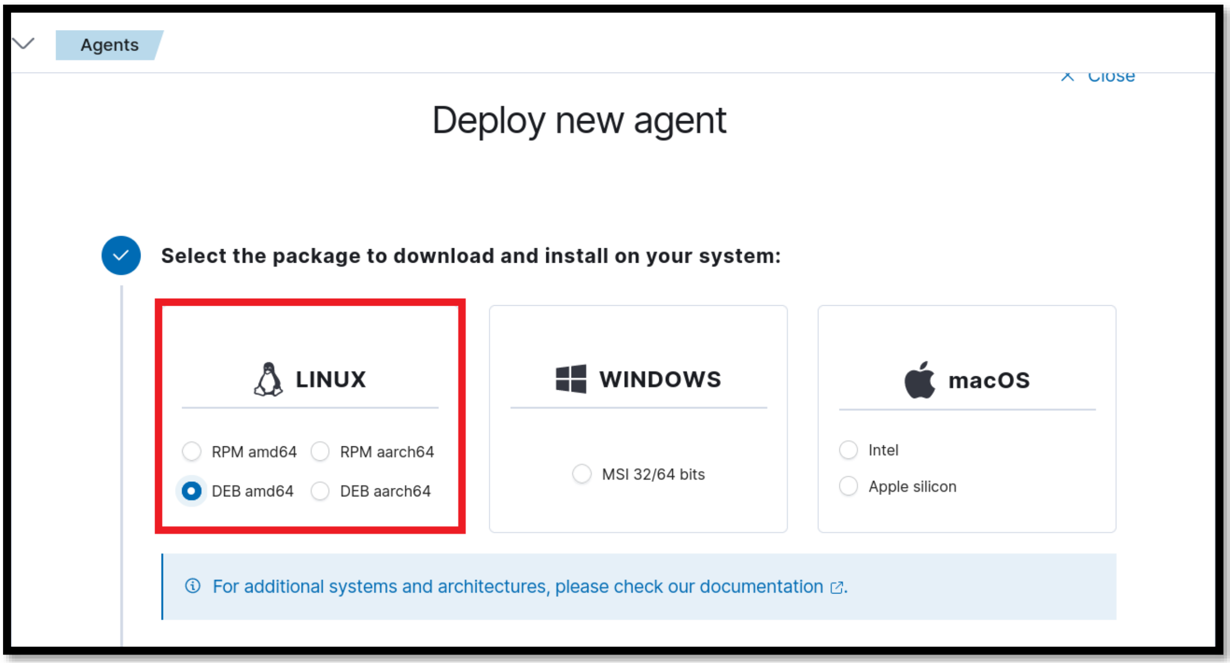
Deploy new agent Refresh Export formatted

id!=000 and Search WQL Refresh

ID	Name	IP address	Group(s)	Operating system	Cluster node	Version	Status	Actions
001	Windows11	192.168.100.32	default	Microsoft Windows 11 Pro 10.0.22621.1992	node01	v4.7.3	● disconnected	
002	Ubuntu	192.168.100.26	default	Ubuntu 18.04 LTS	node01	v4.7.3	● disconnected	

Rows per page: 10

Now chose Linux (DEB amd64) if you are using ubuntu os.


Agents
Close
For additional systems and architectures, please check our documentation.

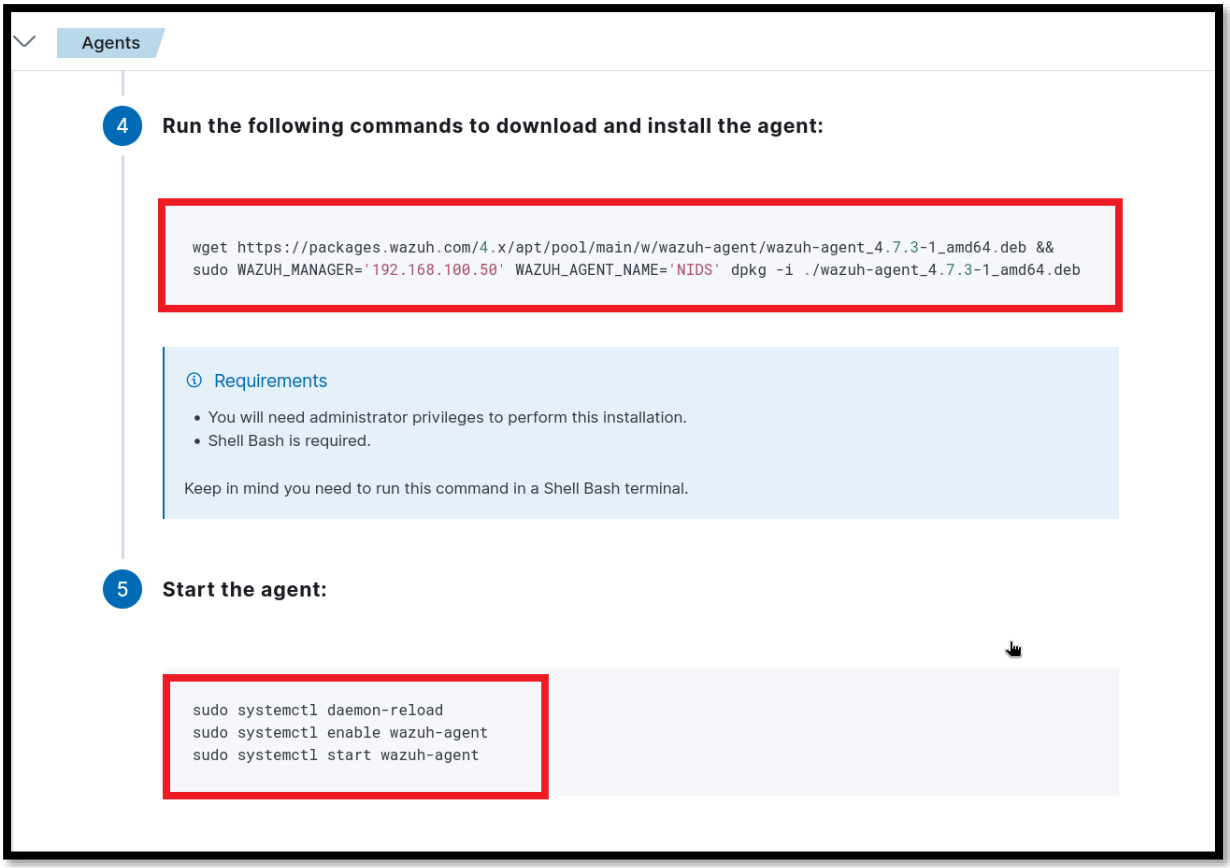
Deploy new agent

Select the package to download and install on your system:

LINUX
○ RPM amd64 ○ RPM aarch64
● DEB amd64 ○ DEB aarch64

WINDOWS
○ MSI 32/64 bits

macOS
○ Intel
○ Apple silicon



4 **Run the following commands to download and install the agent:**

```
wget https://packages.wazuh.com/4.x/apt/pool/main/w/wazuh-agent/wazuh-agent_4.7.3-1_amd64.deb &&
sudo WAZUH_MANAGER='192.168.100.50' WAZUH_AGENT_NAME='NIDS' dpkg -i ./wazuh-agent_4.7.3-1_amd64.deb
```

Requirements

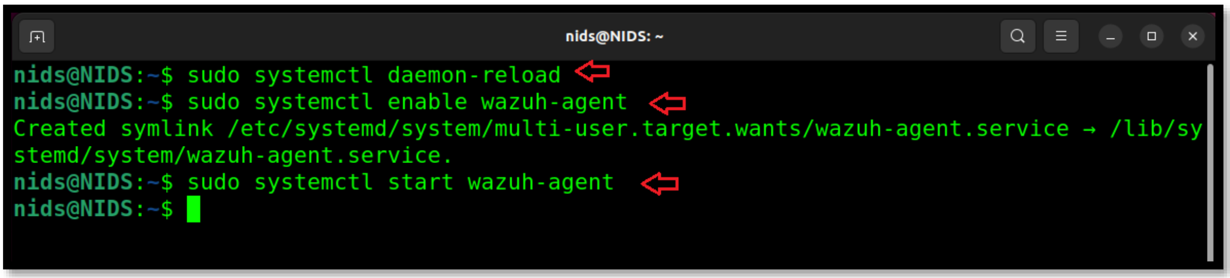
- You will need administrator privileges to perform this installation.
- Shell Bash is required.

Keep in mind you need to run this command in a Shell Bash terminal.

5 **Start the agent:**

```
sudo systemctl daemon-reload
sudo systemctl enable wazuh-agent
sudo systemctl start wazuh-agent
```

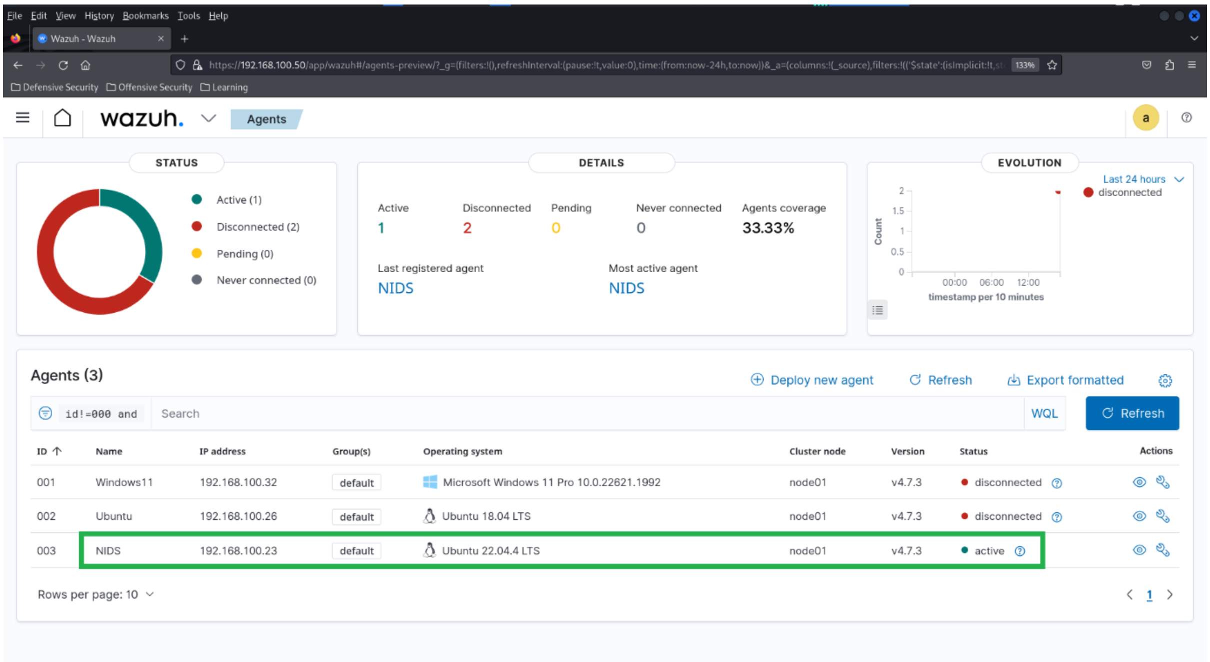
Follow same as shown in figure.



```
nids@NIDS:~$ sudo systemctl daemon-reload
nids@NIDS:~$ sudo systemctl enable wazuh-agent
Created symlink /etc/systemd/system/multi-user.target.wants/wazuh-agent.service → /lib/sy
stemd/system/wazuh-agent.service.
nids@NIDS:~$ sudo systemctl start wazuh-agent
nids@NIDS:~$
```

Note: In first lab of Wazuh series I already covered how to configure agents.

You can see new agent is active and deployed.



File Edit View History Bookmarks Tools Help
Wazuh - Wazuh
https://192.168.100.50/app/wazuh#/agents-preview/?_g=(filters:!(),refreshInterval:(pause:!t,value:0),time:(from:now-24h,to:now))&_a=(columns:!(_source),filters:!(('$state':(isImplicit:!t,st
133%
Defensive Security
Offensive Security
Learning
wazuh.
Agents
a

STATUS

- ● Active (1)
- ● Disconnected (2)
- ● Pending (0)
- ● Never connected (0)

DETAILS

Active	Disconnected	Pending	Never connected	Agents coverage
1	2	0	0	33.33%

Last registered agent: NIDS

Most active agent: NIDS

EVOLUTION
Last 24 hours
● disconnected
Count
2
1.5
1
0.5
0
00:00
06:00
12:00
timestamp per 10 minutes

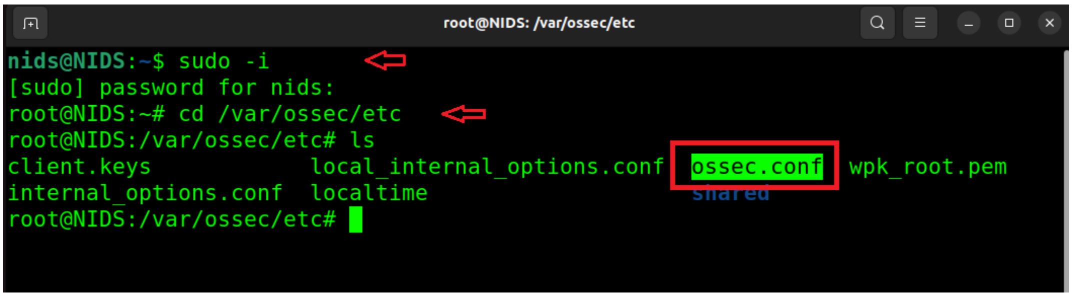
Agents (3)

Deploy new agent
Refresh
Export formatted
id!=000 and
Search
WQL
Refresh

ID ↑	Name	IP address	Group(s)	Operating system	Cluster node	Version	Status	Actions
001	Windows11	192.168.100.32	default	Microsoft Windows 11 Pro 10.0.22621.1992	node01	v4.7.3	● disconnected	
002	Ubuntu	192.168.100.26	default	Ubuntu 18.04 LTS	node01	v4.7.3	● disconnected	
003	NIDS	192.168.100.23	default	Ubuntu 22.04.4 LTS	node01	v4.7.3	● active	

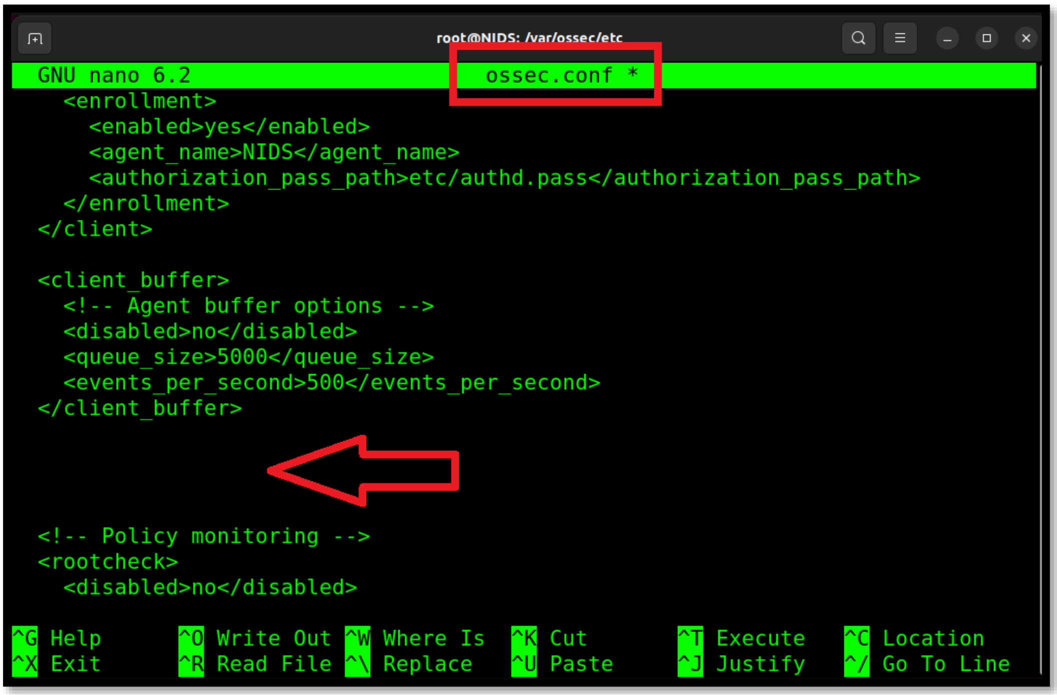
Rows per page: 10

Now we have to configure suricata logs into Wazuh-agent “ossec.conf” file.



root@NIDS: /var/ossec/etc

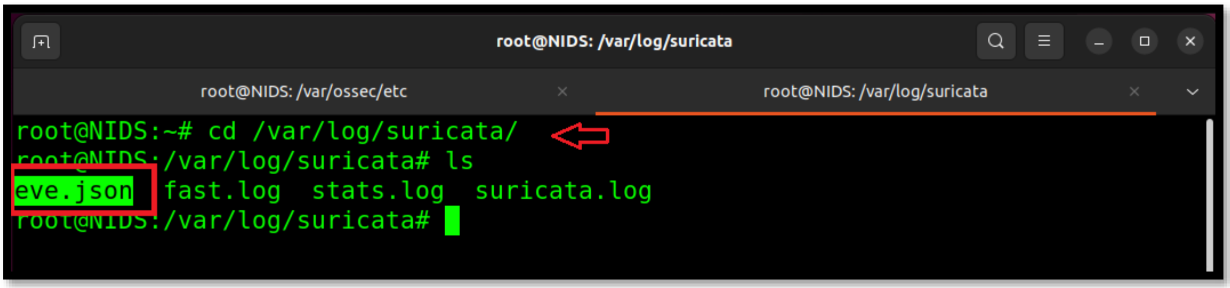
```
nids@NIDS:~$ sudo -i
[sudo] password for nids:
root@NIDS:~# cd /var/ossec/etc
root@NIDS:/var/ossec/etc# ls
client.keys            local_internal_options.conf  ossec.conf  wpk_root.pem
internal_options.conf  localtime                    shared
root@NIDS:/var/ossec/etc# 
```



root@NIDS: /var/ossec/etc

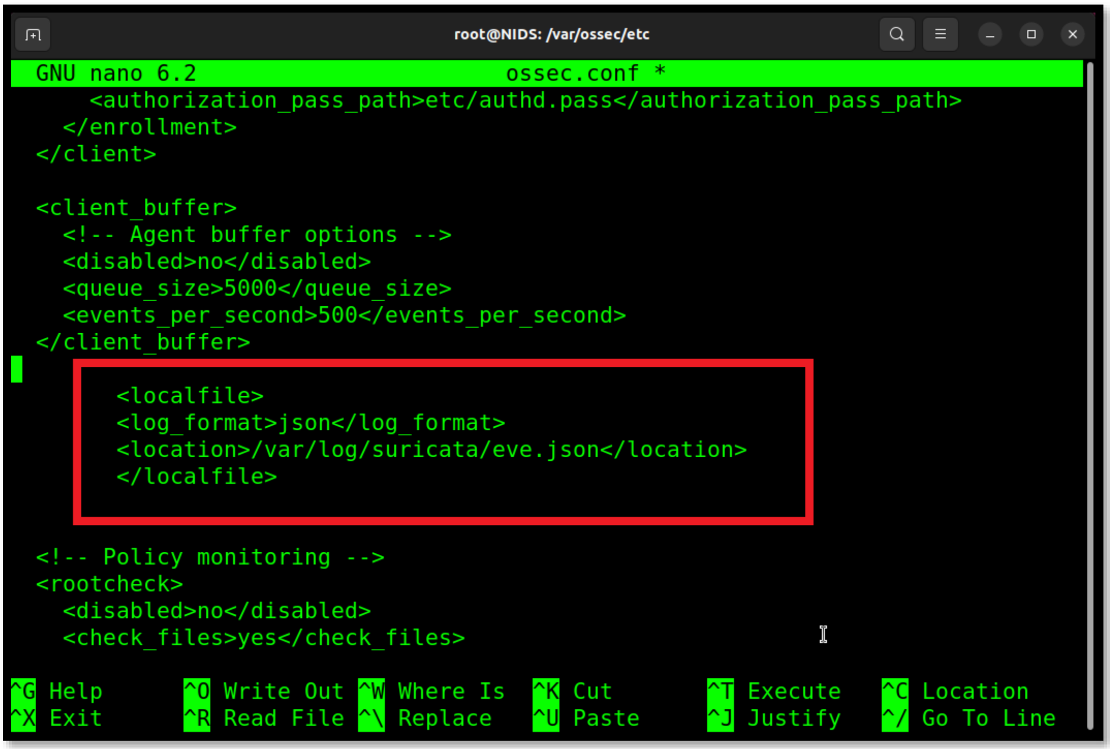
```
  GNU nano 6.2                  ossec.conf *
    <enrollment>
      <enabled>yes</enabled>
      <agent_name>NIDS</agent_name>
      <authorization_pass_path>etc/authd.pass</authorization_pass_path>
    </enrollment>
  </client>

  <client_buffer>
    <!-- Agent buffer options -->
    <disabled>no</disabled>
    <queue_size>5000</queue_size>
    <events_per_second>500</events_per_second>
  </client_buffer>



  <!-- Policy monitoring -->
  <rootcheck>
    <disabled>no</disabled>

^G Help      ^O Write Out ^W Where Is  ^K Cut       ^T Execute   ^C Location
^X Exit      ^R Read File ^\ Replace   ^U Paste     ^J Justify   ^/ Go To Line
```

Here is the location of suricata logs “eve.json”


root@NIDS: /var/log/suricata
root@NIDS: /var/ossec/etc
root@NIDS: /var/log/suricata

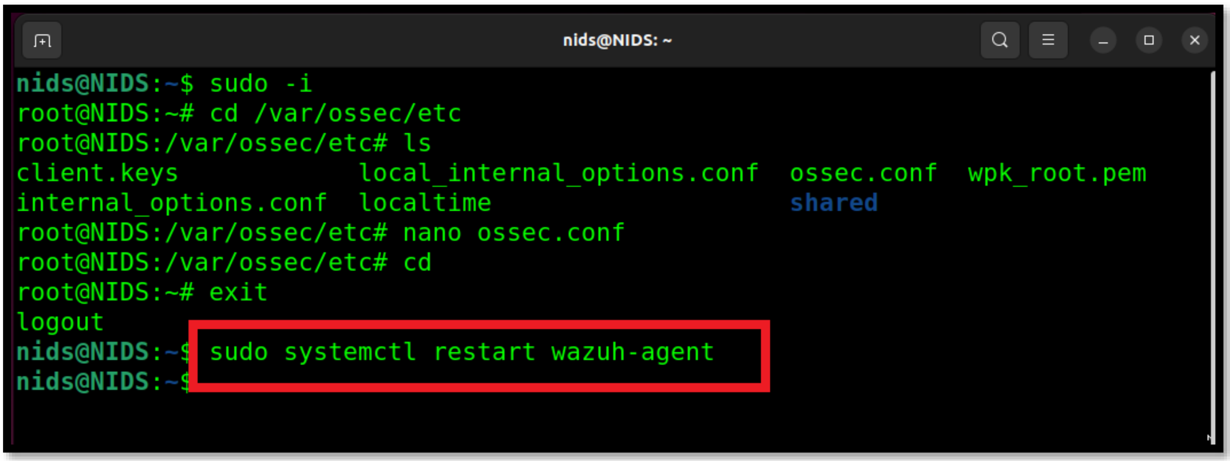
```
root@NIDS:~# cd /var/log/suricata/
root@NIDS:/var/log/suricata# ls
eve.json  fast.log  stats.log  suricata.log
root@NIDS:/var/log/suricata#
```

Now configure suricata logs into “ossec.conf” file.s



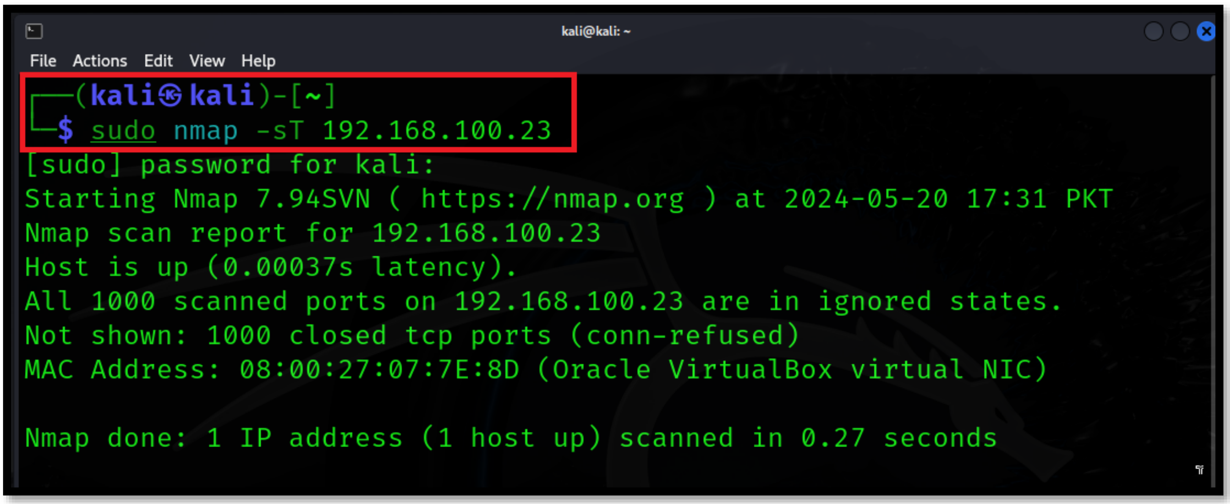
```
      <authorization_pass_path>etc/authd.pass</authorization_pass_path>
    </enrollment>
  </client>

  <client_buffer>
    <!-- Agent buffer options -->
    <disabled>no</disabled>
    <queue_size>5000</queue_size>
    <events_per_second>500</events_per_second>
  </client_buffer>

        <localfile>
        <log_format>json</log_format>
        <location>/var/log/suricata/eve.json</location>
        </localfile>

  <!-- Policy monitoring -->
  <rootcheck>
    <disabled>no</disabled>
    <check_files>yes</check_files>
```

After saving we have to restart wazuh-agent.



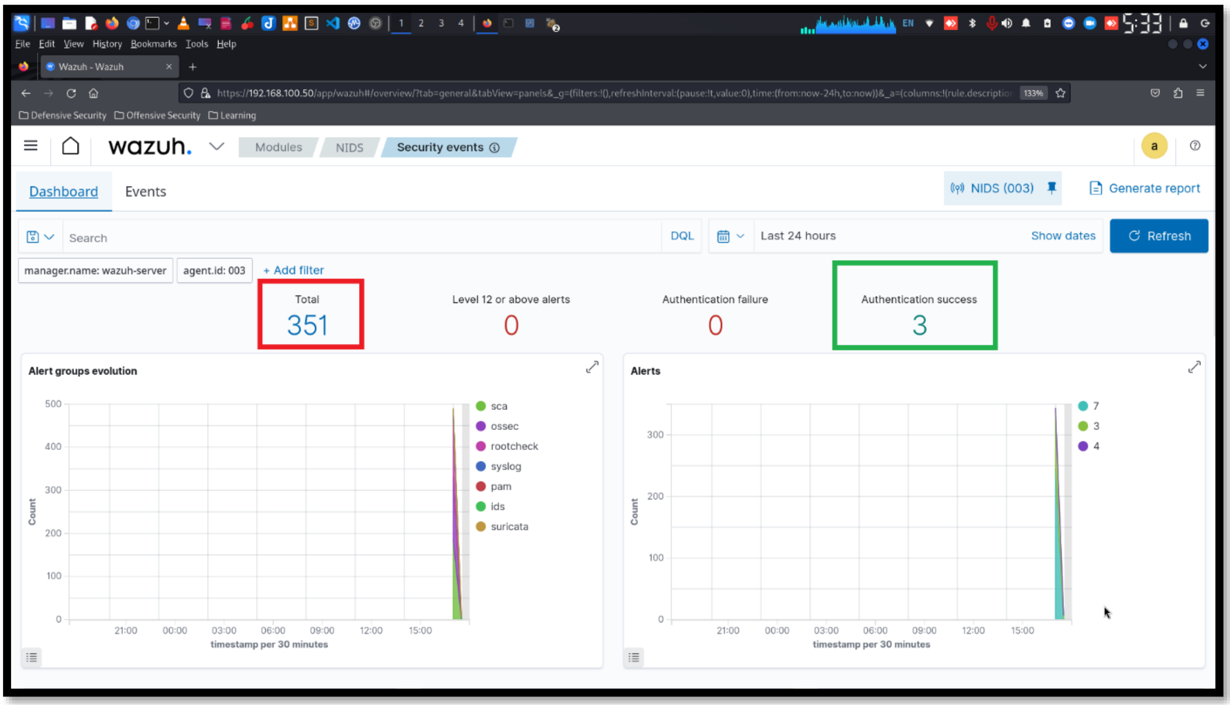
```
nids@NIDS:~$ sudo -i
root@NIDS:~# cd /var/ossec/etc
root@NIDS:/var/ossec/etc# ls
client.keys             local_internal_options.conf  ossec.conf  wpk_root.pem
internal_options.conf   localtime                    shared
root@NIDS:/var/ossec/etc# nano ossec.conf
root@NIDS:/var/ossec/etc# cd
root@NIDS:~# exit
logout
nids@NIDS:~$ sudo systemctl restart wazuh-agent
nids@NIDS:~$
```

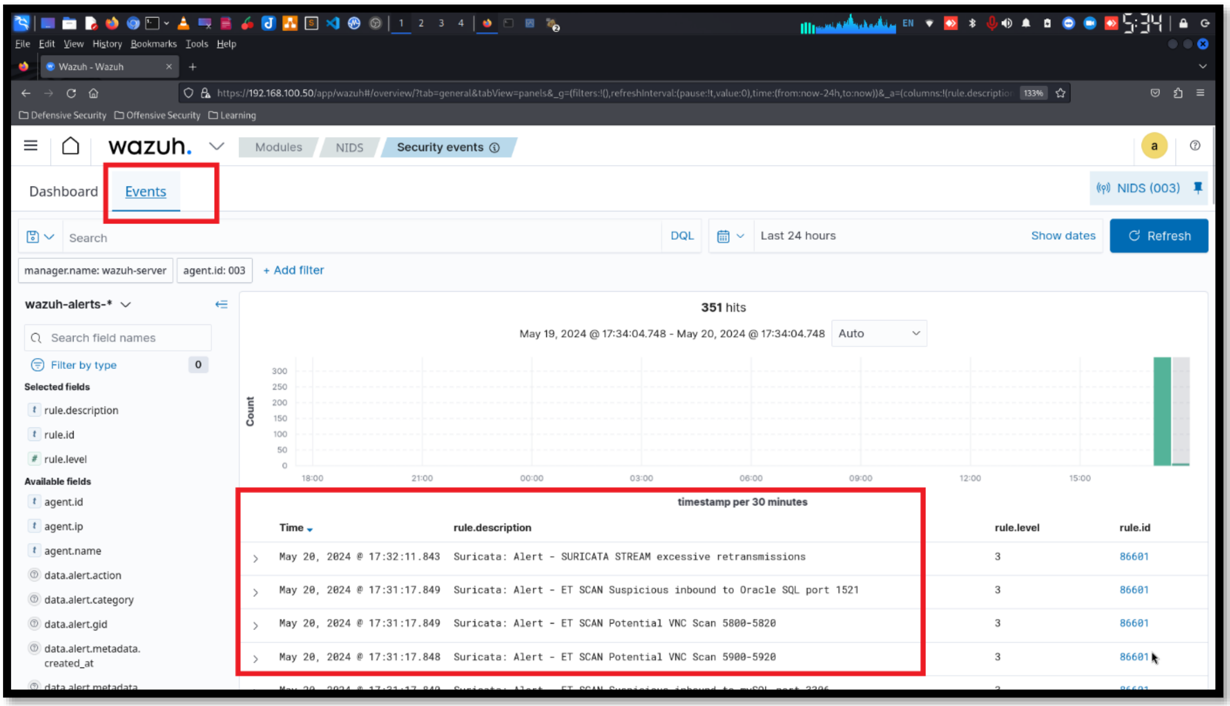
Now launch new terminal in kali linux and perform network scanning with Nmap tool. You can perform any command of nmap.



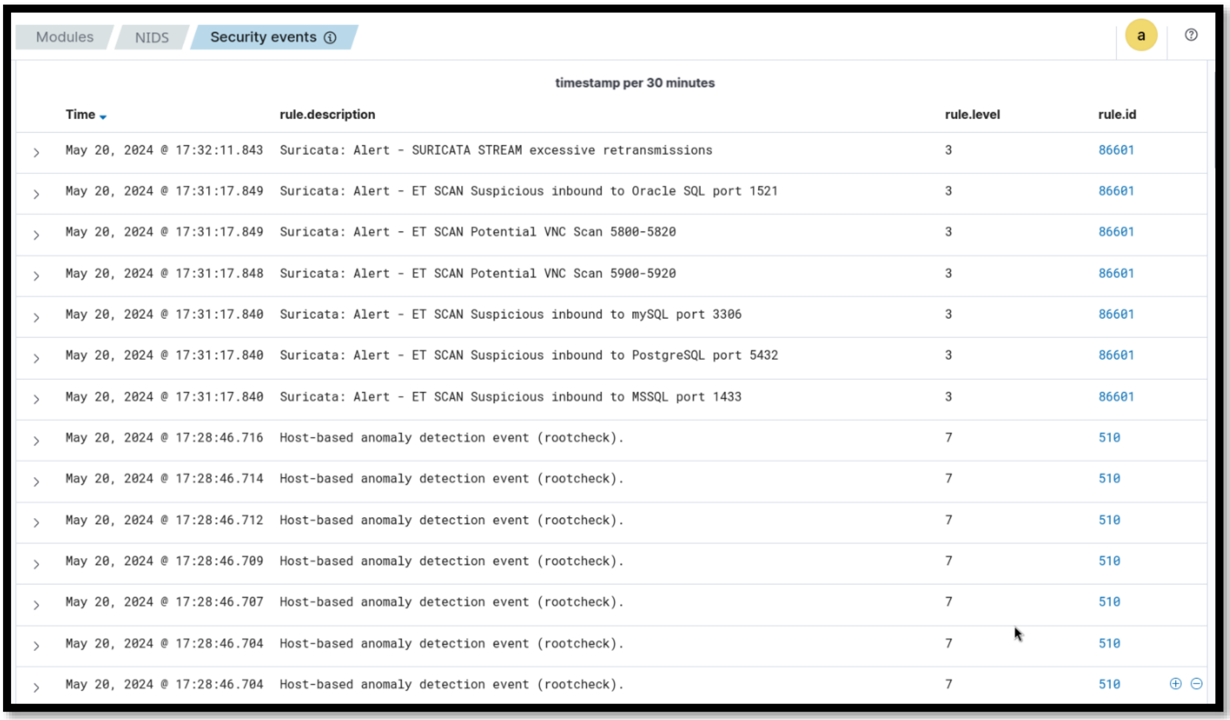
```
┌──(kali㉿kali)-[~]
└─$ sudo nmap -sT 192.168.100.23
[sudo] password for kali:
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-05-20 17:31 PKT
Nmap scan report for 192.168.100.23
Host is up (0.00037s latency).
All 1000 scanned ports on 192.168.100.23 are in ignored states.
Not shown: 1000 closed tcp ports (conn-refused)
MAC Address: 08:00:27:07:7E:8D (Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 0.27 seconds
```

Now go to wazuh dashboard and see the Events.

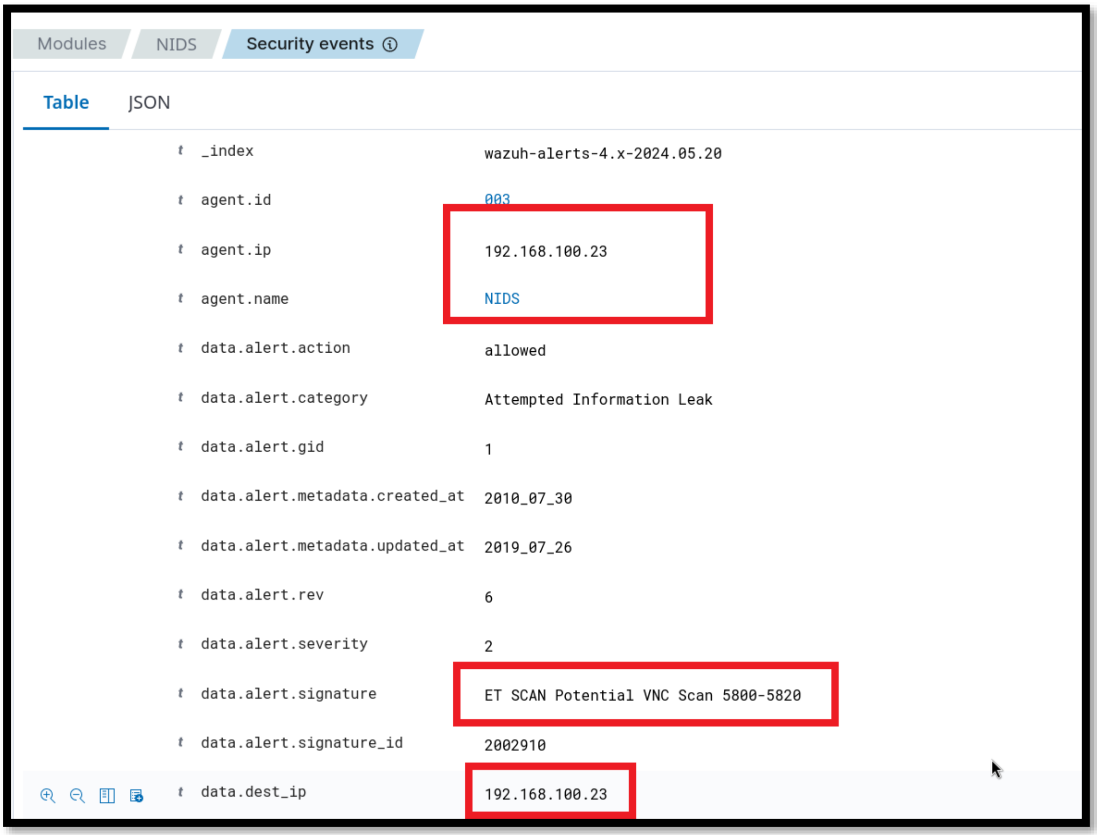




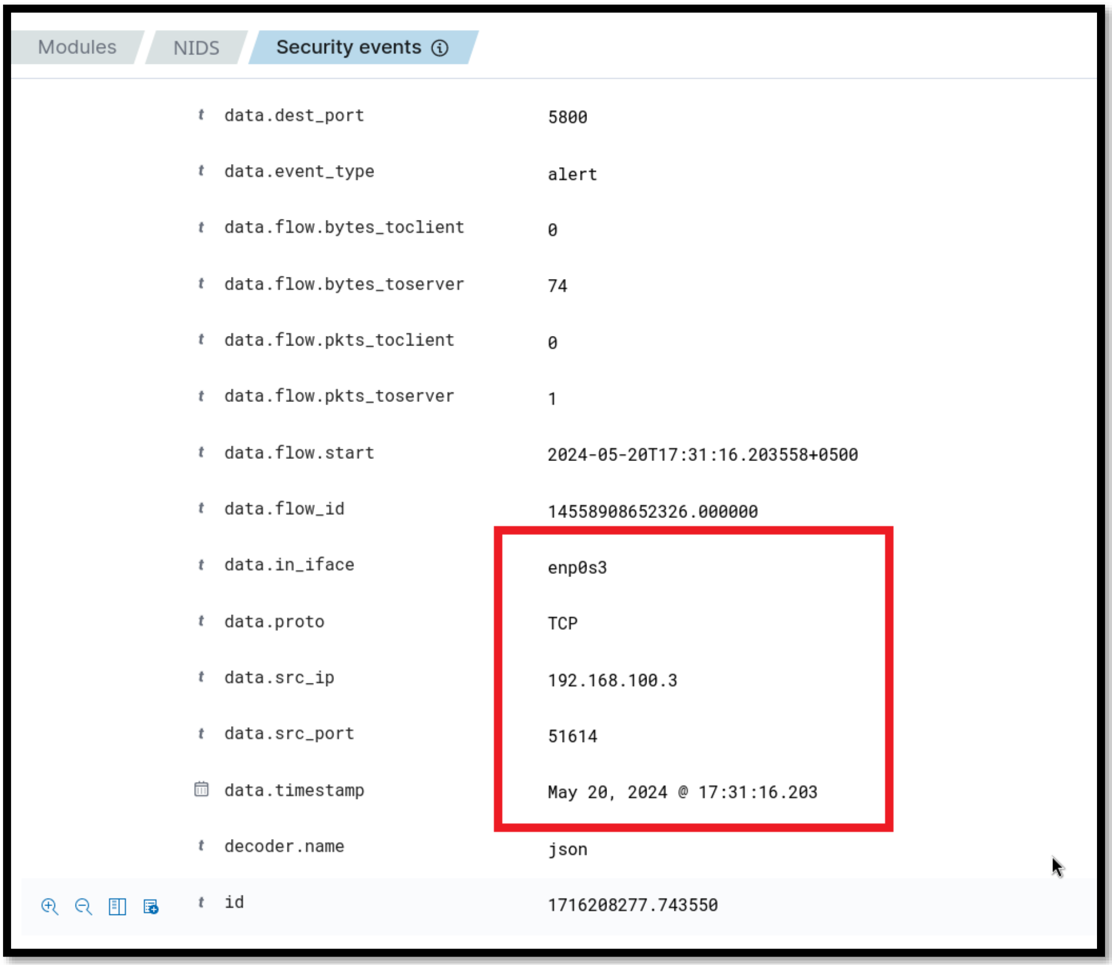
Time	rule.description	rule.level	rule.id
May 20, 2024 @ 17:32:11.843	Suricata: Alert - SURICATA STREAM excessive retransmissions	3	86601
May 20, 2024 @ 17:31:17.849	Suricata: Alert - ET SCAN Suspicious inbound to Oracle SQL port 1521	3	86601
May 20, 2024 @ 17:31:17.849	Suricata: Alert - ET SCAN Potential VNC Scan 5800-5820	3	86601
May 20, 2024 @ 17:31:17.848	Suricata: Alert - ET SCAN Potential VNC Scan 5900-5920	3	86601
May 20, 2024 @ 17:31:17.840	Suricata: Alert - ET SCAN Suspicious inbound to mySQL port 3306	3	86601



Time	rule.description	rule.level	rule.id
May 20, 2024 @ 17:32:11.843	Suricata: Alert - SURICATA STREAM excessive retransmissions	3	86601
May 20, 2024 @ 17:31:17.849	Suricata: Alert - ET SCAN Suspicious inbound to Oracle SQL port 1521	3	86601
May 20, 2024 @ 17:31:17.849	Suricata: Alert - ET SCAN Potential VNC Scan 5800-5820	3	86601
May 20, 2024 @ 17:31:17.848	Suricata: Alert - ET SCAN Potential VNC Scan 5900-5920	3	86601
May 20, 2024 @ 17:31:17.840	Suricata: Alert - ET SCAN Suspicious inbound to mySQL port 3306	3	86601
May 20, 2024 @ 17:31:17.840	Suricata: Alert - ET SCAN Suspicious inbound to PostgreSQL port 5432	3	86601
May 20, 2024 @ 17:31:17.840	Suricata: Alert - ET SCAN Suspicious inbound to MSSQL port 1433	3	86601
May 20, 2024 @ 17:28:46.716	Host-based anomaly detection event (rootcheck).	7	510
May 20, 2024 @ 17:28:46.714	Host-based anomaly detection event (rootcheck).	7	510
May 20, 2024 @ 17:28:46.712	Host-based anomaly detection event (rootcheck).	7	510
May 20, 2024 @ 17:28:46.709	Host-based anomaly detection event (rootcheck).	7	510
May 20, 2024 @ 17:28:46.707	Host-based anomaly detection event (rootcheck).	7	510
May 20, 2024 @ 17:28:46.704	Host-based anomaly detection event (rootcheck).	7	510
May 20, 2024 @ 17:28:46.704	Host-based anomaly detection event (rootcheck).	7	510

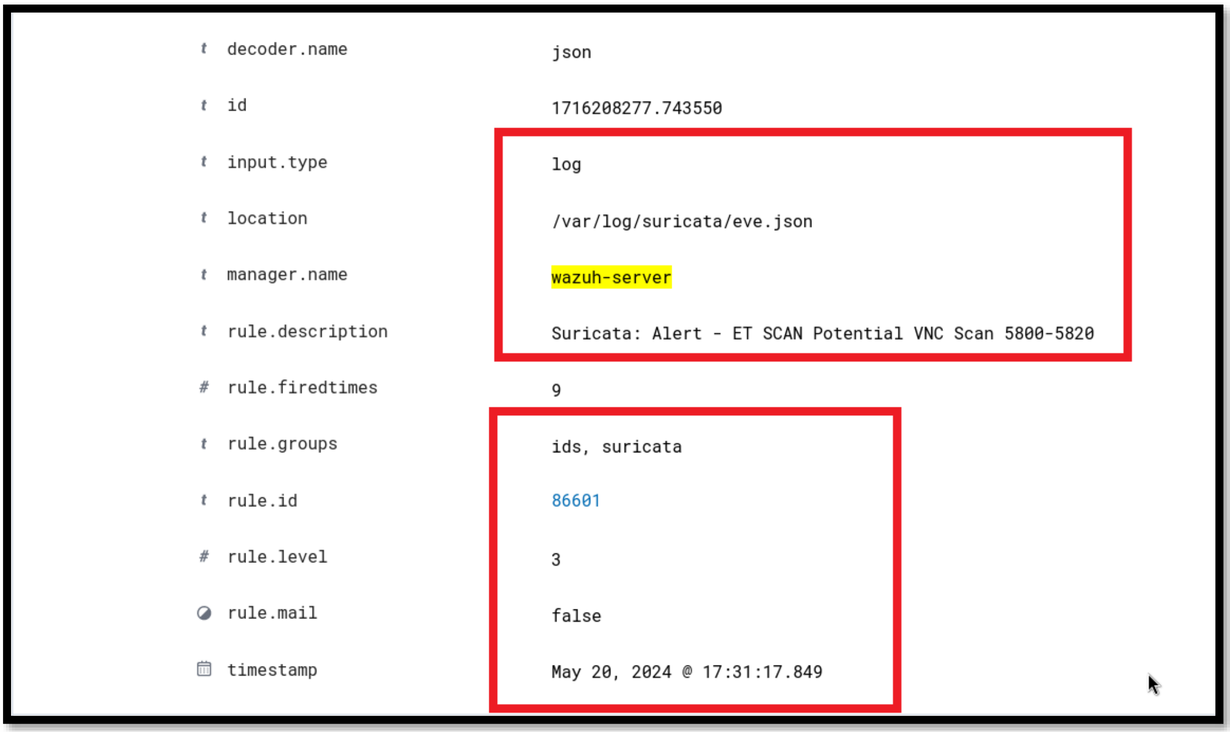


_index	wazuh-alerts-4.x-2024.05.20
agent.id	003
agent.ip	192.168.100.23
agent.name	NIDS
data.alert.action	allowed
data.alert.category	Attempted Information Leak
data.alert.gid	1
data.alert.metadata.created_at	2010_07_30
data.alert.metadata.updated_at	2019_07_26
data.alert.rev	6
data.alert.severity	2
data.alert.signature	ET SCAN Potential VNC Scan 5800-5820
data.alert.signature_id	2002910
data.dest_ip	192.168.100.23



data.dest_port	5800
data.event_type	alert
data.flow.bytes_toclient	0
data.flow.bytes_toserver	74
data.flow.pkts_toclient	0
data.flow.pkts_toserver	1
data.flow.start	2024-05-20T17:31:16.203558+0500
data.flow_id	14558908652326.000000
data.in_iface	enp0s3
data.proto	TCP
data.src_ip	192.168.100.3
data.src_port	51614
data.timestamp	May 20, 2024 @ 17:31:16.203
decoder.name	json
id	1716208277.743550

I highlighted interesting logs analysis fields here.



decoder.name	json
id	1716208277.743550
input.type	log
location	/var/log/suricata/eve.json
manager.name	wazuh-server
rule.description	Suricata: Alert - ET SCAN Potential VNC Scan 5800-5820
rule.firedtimes	9
rule.groups	ids, suricata
rule.id	86601
rule.level	3
rule.mail	false
timestamp	May 20, 2024 @ 17:31:17.849

SUMMARY

In summary, Integrating Wazuh and Suricata can provide a comprehensive security solution that leverages the strengths of both platforms. Suricata can be used to monitor network traffic for threats and intrusions, while Wazuh can collect and analyze logs from Suricata, correlating network-based events with host-based activities for a holistic view of the security landscape. This integration enables organizations to detect, investigate, and respond to threats more effectively by combining network and endpoint security data into a unified system.