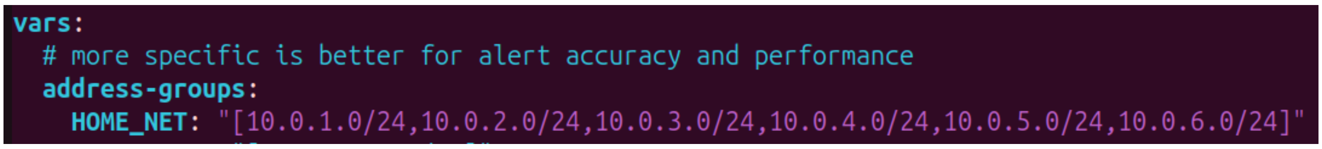
(2) Configure Suricata

Suricata is a powerful IDS/IPS, meaning it can not only detect threats but can actively block them. Its an open source tool that inspects network traffic for any malicious activity via rules. Rules are instruction that help the engine match certain string, ip address be it source or destination, ports and different packet sizes which can either lead to an alert creation or it being blocked outright.

Suricata stores its main configuration file in the location `/etc/suricata/suricata.yaml`. Here we will make a copy of the original yaml file and work on that, such that if anything goes wrong the main config file is still there.

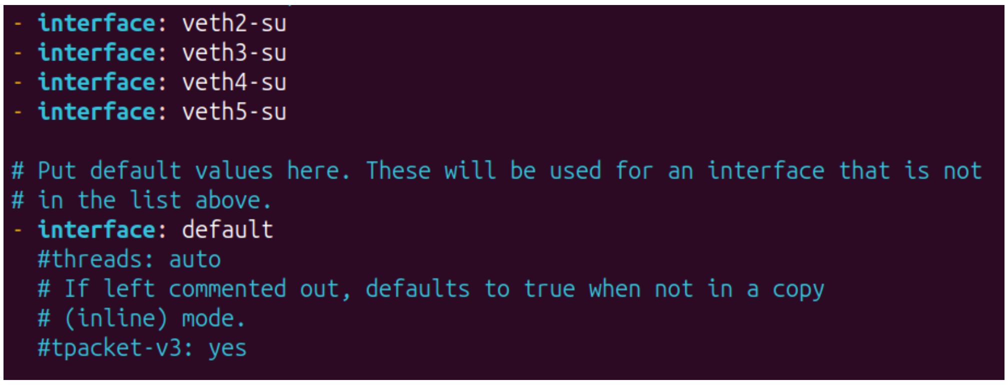
In the configuration files, first comes the variables section where the HOME_NET variable is defined, here we will use our subnets used in the given task for greater accuracy. This is what it will look like after changing it



```
vars:
  # more specific is better for alert accuracy and performance
  address-groups:
    HOME_NET: "[10.0.1.0/24,10.0.2.0/24,10.0.3.0/24,10.0.4.0/24,10.0.5.0/24,10.0.6.0/24]"
```

The default log directory will be set as is, scrolling down we see the types section where our concern would be the "alerts" and the "tls" section where changes can be made for advanced configurations. Sections like "tcp-data" and "http-body-data" are for Deep Packet Inspection or a feature similar to follow tcp stream in a tool called Wireshark. In the logging section we can see the option of syslog, this can be enabled to centralize alerts for a future vector of combining Suricata together with a SIEM tool.

In the af-packet section where the different interfaces are mentioned we will use the interfaces in the Suricata namespace veth1-su through veth5-su from where the traffic will originate from, like this



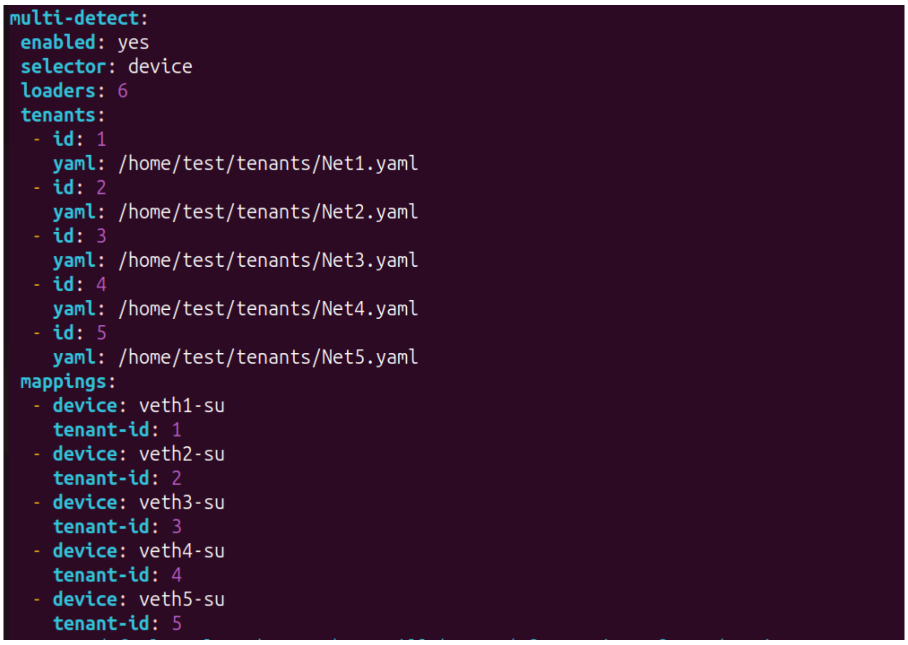
```
- interface: veth2-su
- interface: veth3-su
- interface: veth4-su
- interface: veth5-su

# Put default values here. These will be used for an interface that is not
# in the list above.
- interface: default
  #threads: auto
  # If left commented out, defaults to true when not in a copy
  # (inline) mode.
  #tpacket-v3: yes
```

Now as this task requires different rulesets applying to traffic from different namespaces, we will use a feature of suricata called Multi-tenancy. This is not available by default and we have to add the section manually, using the `multi-detect` section.

The multi-detect section we will use will look like this

It is important to be careful about indentations in yaml as it is indentation sensitive language

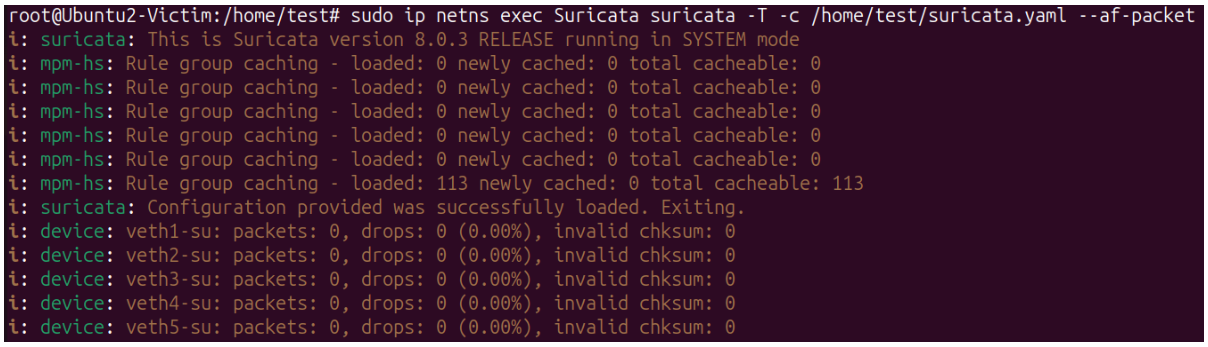


```
multi-detect:
 enabled: yes
 selector: device
 loaders: 6
 tenants:
  - id: 1
    yaml: /home/test/tenants/Net1.yaml
  - id: 2
    yaml: /home/test/tenants/Net2.yaml
  - id: 3
    yaml: /home/test/tenants/Net3.yaml
  - id: 4
    yaml: /home/test/tenants/Net4.yaml
  - id: 5
    yaml: /home/test/tenants/Net5.yaml
 mappings:
  - device: veth1-su
    tenant-id: 1
  - device: veth2-su
    tenant-id: 2
  - device: veth3-su
    tenant-id: 3
  - device: veth4-su
    tenant-id: 4
  - device: veth5-su
    tenant-id: 5
```

Selector is set to device as we are using network interfaces, loaders is set to 6 to have 6 independent detection engine instance to handle traffic. Tenants section includes the IDs and their associated rules applicable to only that particular namespace. the mappings section is where we assign the tenant id to the device/interface which we want suricata to monitor and use the associated rules on that namespace. We Test this configuration using the command.

```
sudo ip netns exec Suricata suricata -T -c /home/test/suricata.yaml --af-packet
```

Here we use "-T" to test configuration, "-c" for specific location of the config file and "--af-packet" forces suricata to use the af-packet section. The output looks like this



```
root@Ubuntu2-Victim:/home/test# sudo ip netns exec Suricata suricata -T -c /home/test/suricata.yaml --af-packet
i: suricata: This is Suricata version 8.0.3 RELEASE running in SYSTEM mode
i: mpm-hs: Rule group caching - loaded: 0 newly cached: 0 total cacheable: 0
i: mpm-hs: Rule group caching - loaded: 0 newly cached: 0 total cacheable: 0
i: mpm-hs: Rule group caching - loaded: 0 newly cached: 0 total cacheable: 0
i: mpm-hs: Rule group caching - loaded: 0 newly cached: 0 total cacheable: 0
i: mpm-hs: Rule group caching - loaded: 0 newly cached: 0 total cacheable: 0
i: mpm-hs: Rule group caching - loaded: 113 newly cached: 0 total cacheable: 113
i: suricata: Configuration provided was successfully loaded. Exiting.
i: device: veth1-su: packets: 0, drops: 0 (0.00%), invalid chksum: 0
i: device: veth2-su: packets: 0, drops: 0 (0.00%), invalid chksum: 0
i: device: veth3-su: packets: 0, drops: 0 (0.00%), invalid chksum: 0
i: device: veth4-su: packets: 0, drops: 0 (0.00%), invalid chksum: 0
i: device: veth5-su: packets: 0, drops: 0 (0.00%), invalid chksum: 0
```

This shows us that all devices are visible by suricata and that it can monitor the traffic on those interfaces. This sums up the configuration of the tool suricata with respect to the given task, next we move to writing rules to detect various types of network traffic and create alerts.