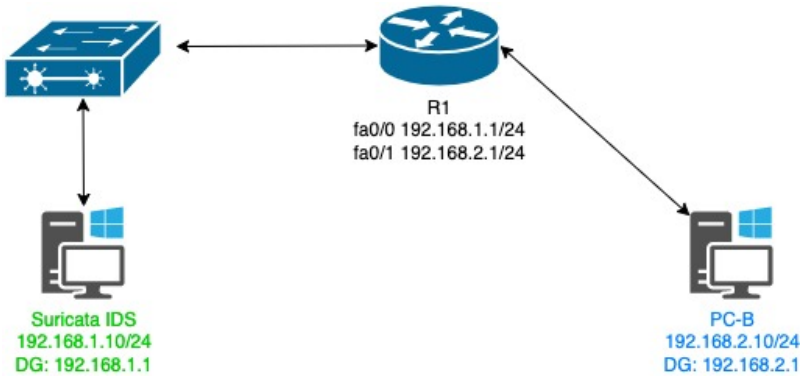
SURICATA IDS SETUP

Topology Setup



	Suricata IDS	PC-B (Blue)
Adapter 1	NAT	
Adapter 2	VMWare12	VMWare11
Adapter 2 IP	192.168.1.10	192.168.2.10
Adapter 2 Subnet	255.255.255.0	255.255.255.0
Adapter 2 Default-gateway	192.168.1.1	192.168.2.1

- Build the topology as shown above
- Load R1 script (In Moodle) into the Router and enable fa0/0 and fa0/1 interfaces
- Set the two VMs to use the VMWare adapters and IP addresses of PC-B as shown in the table.
- Test connectivity throughout the network using *ping*. If everything does not connect, troubleshoot before moving on.

1. Suricata Installation

Tasks:

- Take a snapshot of the Ubuntu 22.04 VM and name the snapshot 'Base Image'
- Boot the VM and set the IP address as above
- Open a Terminal and start by installing prerequisite packages and adding the Suricata repository to our list
- Update **ubuntu** and install **suricata**
- Install **curl** and **json reader**

Commands:

```
sudo apt-get install software-properties-common
sudo add-apt-repository ppa:oisf/suricata-stable
sudo apt-get update && sudo apt-get install suricata -y
sudo apt install curl
sudo apt-get install jq
```

- Since we want Suricata to be running all the time we will set it to start on bootup

```
sudo systemctl enable suricata.service
```

2. Check status/start/stop

Tasks:

- Before we configure and modify Suricata for our requirements we will check it is installed correctly and able to run
- Start suricata, check its status and then stop the IDS running so we can configure it

Commands:

```
sudo systemctl start suricata
sudo systemctl status suricata.service
sudo systemctl stop suricata
```

3. Configure Suricata

Tasks:

- Suricata configuration files and rules are stored in */etc/suricata* and */etc/suricata/rules* respectively. Start by listing these files to ensure they are present
- Next configure Suricata by editing *suricata.yaml*
- Set the appropriate variables listed below to set internal and external Ips and interface
- The *communit-id* is used for event correlation. It can be useful when using tools like **zeek** or importing logs in json format. When enabled, Suricata will create records so they can be matched to events in zeek. The seed needs to be unique.
- Under *Default-rule-path* you will the file where Suricata rules and stored (we will download these later). Specify a new file called *local.rules* that we will use for cus-

tom rules that we will create. We will need to create this file later in directory */var/lib/suricata/rules*

- Save the **yaml** file and exit

Commands:

 # the yaml file is the suricata configuration file

ls -la /etc/suricata

 # below rules directory is set of prepackaged rules for suricata. This may not be created at this stage so do not worry.

ls -al /etc/suricata/rules

ls -al /var/lib/suricata/rules

edit the yaml file and set the below variables
can also use CTRL-W to search for text in nano editor

sudo nano -c /etc/suricata/suricata.yaml

```
Vars:
  HOME_NET  "x.x.x.x/24"                  # Check your local ip address is in the list
  EXTERNAL_NET "!HOME_NET"                #

af-packet:
  Interface: xxx                          # change to you interface (ens33)

pcap:
  Interface: xxx                          # change to you interface (ens33)

Community-id: true                        # change to true


Default-rule-path: /var/lib/suricata/rules    # possibly line 1919

Rule-files:
 - suricata.rules
 - local.rules                            # add local.rules - will create this later
```

Save and exit yaml file (CTRL X, Y, enter)

4. Update Suricata configuration

Tasks:

- Start be downloading the current maintained rules list for Suricata. These are fetched from http://emergingthreats.net
- (Optional) list further sources for downloading additional rules and specify to use these as required. Some of these will require a subscription. In the example command below it would pull down additional rules related to windows malware from a different repository from the default Suricata location
- Update Suricata to use the new rules

 # ignore any ERROR about protocols that are not enabled – we are not using them

sudo suricata-update
sudo ls -al /var/lib/suricata/rules
sudo suricata-update list-sources
sudo suricata-update enable-source malsilo/win-malware
sudo suricata-update

5. Test Suricata configuration

Tasks:

- Test the newly configured suricata functions correctly. Here *-T* tells suricata we want to test the configuration and *-v* runs verbose. Suricata will output log files logs (*fast.log* and *eve.json*) in directory */var/log/suricata* These are log files we can look to review events Suricata has logged
- Note: Suricata can be run as a background Daemon using *-D*
- We can also run Suricata by specifying configuration file and rules files as shown below. When doing this replace the variables e.g. replace eth0 with your interface
- (Optional) Default way to run Suricata with the normal rule set

Commands:

 # ignore any ERROR about local.rules as we haven't created the file yet
if threads are created on step two this means it is working so CTRL-C to stop

sudo suricata -T -c /etc/suricata/suricata.yaml -v
sudo suricata -c suricata.yaml -s signatures.rules -i ens33

```
sudo systemctl start suricata.service
```

6. Test Suricata maintained rules

Tasks:

- We are now ready to test suricata when running live.
- We can use curl to request a page. This test facility is designed to return the request as *root* user simulating an attack as though someone has got root access

Commands:

```
sudo ls -al /var/lib/suricata/rules/
curl http://testmynids.org/uid/index.html     # should return a root username
sudo cat /var/log/suricata/fast.log           # this may not be configured. If so try below
```

```
sudo cat /var/log/suricata/eve.json           # just check there is data present at this stage
```

7. Create custom rules

Tasks:

- We now know that the maintained Suricata rule set we downloaded earlier works. We can proceed to create some custom rules that would be unique to our installation i.e. an organisations specific network setup they are tyring to protect
- First, stop Suricata running and create a simple rule to detect ICMP echo requests

  alert icmp any any -> $HOME_NET any (msg:"ICMP External Ping"; sid:1; rev:1;)

  This rule will log any pings coming from an external network to our Home network using any port
  SID is a signature id we can specify 1 as it is our first custom rule
  REV is revision which again we set to 1

- Test the configuration as we did before and restart Suricata
- Start Suricata running

Commands:

```
sudo systemctl stop suricata.service
sudo nano /var/lib/suricata/rules/local.rules

        alert icmp any any -> $HOME_NET any (msg:"ICMP External Ping"; sid:1; rev:1;)


sudo suricata -T -c /etc/suricata/suricata.yaml -v
sudo systemctl start suricata.service
```

8. Test new custom rule

Tasks:

- Test the new custom rule by pinging Suricata from PC-B on the external network
- Check both the *fast.log* and *eve.json* log files for this activity

Tasks:

```
sudo cat /var/log/suricata/fast.log

# if the log is empty try pinging from the router as suricata may already know about the sub-net of PC-B

sudo tail -F /var/log/suricata/eve.json | jq 'select(.event_type=="alert")'
```

- Take a snapshot of the Ubuntu 22.04 VM and name the snapshot 'Suricata Install' you can return to this in the future if needed.

9. OPTIONAL (If time permits)

- Research how to write a Suricata rule to detect malicious ssh activity. Configure equipment and perform a simulated ssh connection to the router – check logs for an appropriate entry.

- SUGGESTED ANSWER:

  Configure interface

configure vlan 1 on switch with ip for example 192.168.1.3

- setup up local user account

username admin privilege 15 secret cisco12345

- configure domain name

ip domain-name ccnasecurity.com

- Generate RSA encryption

crypto key generate rsa general-keys modulus 1024

- Setting up VTY Line

S1(config)# line vty 0 4
S1(config-line)# privilege level 15
S1(config-line)# exec-timeout 5 0
S1(config-line)# login local
S1(config-line)# transport input ssh
S1(config-line)# exit

- Setup span port on switch

monitor session 1 source interface (this will be port going to router)
monitor session 1 destination interface (this will be port to Suricata IDS)

- If Suricata is still running stop the service

sudo systemctl stop suricata.service

- Add new rule inside sudo nano /var/lib/suricata/rules/local.rules

Example but can do in one direction instead this was just a quick one since was running out of time

Alert ssh any any <> (msg:" test rule- SSH Has been Detected"; sid: 200; rev:1;)

- Test the configuration

sudo suricata -T -c /etc/suricata/suricata.yaml -v

- start Service again

sudo systemctl start suricata.service

Then on Blue Windows VM open putty like did in lab 1 and SSH to Vlan 1 ip address configured earlier.