The exercises are designed for students to finish in an individual capacity. The exercises are not designed to be completed in tutorial sessions but rather to give you some tasks and a starting point to continue and complete on your own.

1 Lab Tasks

The objective of this lab is to setup and use an Intrusion Detection System to monitor the traffic in the network. We will use the IDS in a passive configuration monitoring the traffic in the network. To capture the traffic and to send them to the IDS we will configure port-mirroring (or a SPAN port) on the switch.

Snort is an Open Source IDS/IPS. It uses a series of rules that help define malicious network activity and uses those rules to find packets that match against them and generates alerts for users. Snort can be deployed inline as an IPS to stop these packets. In this lab we deploy Snort in a passive configuration where it can only act as an IDS.

Port Mirroring is duplicating the traffic from one switch port (or multiple ports) and forwarding that duplicate traffic to another port, where it can be captured and analyzed by monitoring tools or devices.

Monitoring Port: IDS is configured with 2 (or more) network interfaces. The additional port added is the monitoring port. This is connected to the mirrored port in the switch and used receive a dump of the traffic which needs to be monitored. IDS/IPS can have multiple monitoring ports to collect traffic from different parts of large networks.

1. **Network Setup**

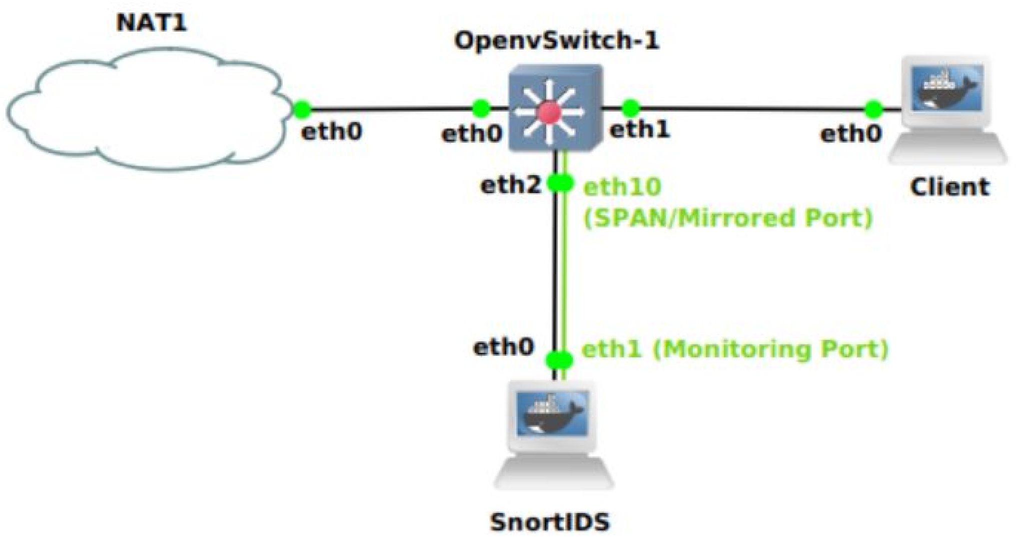


Figure 1: Network Diagram

Create a **new GNS3 project** and follow the below instructions to setup the four nodes as shown in the above diagram.

(a) Setting up **Open vSwitch (OVS) :**
Open vSwitch, sometimes abbreviated as OVS, is an open-source implementation of a distributed virtual multilayer switch. We use OVS in this lab as generic switches available in GNS3 do not have port-mirroring feature.

i. Open your host machine's web browser and download the appliance file from **this link**.

ii. Import the appliance file to GNS3.
File menu > Import appliance > Select the downloaded appliance file : **openvswitch.gns3a**

iii. OVS should be available in the available devices now. Add it to the project use drag and drop.

(b) Setting up the **NAT Cloud :** NAT cloud is used as a gateway to the Internet.

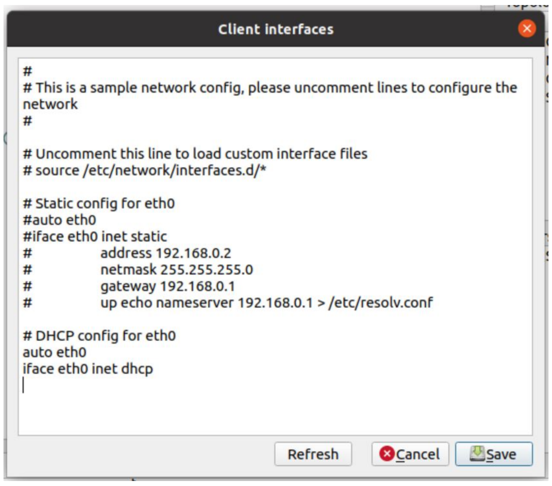
i. Drag and drop a NAT device to the project.

(c) Setting up the **Client :** We use this node to generate traffic to be inspected by the IDS.

i. Drag and drop a Ubuntu-24.04-plus-essentials device.

ii. Rename the device as Client. (Right-click > Configure > Name)

iii. Configure the network interface with DHCP



```
#
# This is a sample network config, please uncomment lines to configure the
network
#

# Uncomment this line to load custom interface files
# source /etc/network/interfaces.d/*

# Static config for eth0
#auto eth0
#iface eth0 inet static
#           address 192.168.0.2
#           netmask 255.255.255.0
#           gateway 192.168.0.1
#           up echo nameserver 192.168.0.1 > /etc/resolv.conf

# DHCP config for eth0
auto eth0
iface eth0 inet dhcp
```

Figure 2: Client IP configurations

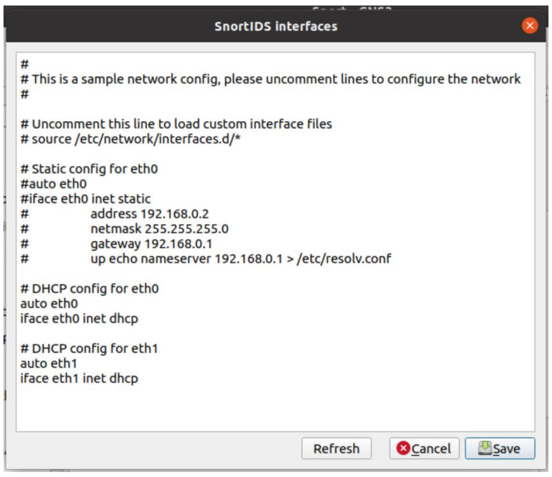
(d) Setting up the **SnortIDS :**

i. Drag and drop a Ubuntu-24.04-plus-essentials device.

ii. Rename the device as SnortIDS (Right-click > Configure > Name)

iii. Increase the number of Adapters to 2. (Right-click > Configure > Adapters)

iv. Configure both network interfaces with DHCP



```
#
# This is a sample network config, please uncomment lines to configure the network
#

# Uncomment this line to load custom interface files
# source /etc/network/interfaces.d/*

# Static config for eth0
#auto eth0
#iface eth0 inet static
#           address 192.168.0.2
#           netmask 255.255.255.0
#           gateway 192.168.0.1
#           up echo nameserver 192.168.0.1 > /etc/resolv.conf

# DHCP config for eth0
auto eth0
iface eth0 inet dhcp

# DHCP config for eth1
auto eth1
iface eth1 inet dhcp
```

Figure 3: SnortIDS IP configurations

(e) Configurations on the VM
Run the below command directly on the GNS3 VM shell.

```
sudo modprobe openvswitch
```

(f) Connecting the devices
Use the Link tool to connect the devices with the port numbers exactly as shown in the network diagram.

Example: eth2 port of OVS *MUST* connect to eth0 of SortIDS and eth10 of of OVS *MUST* connect to eth1 of SnortIDS.

2. **Configuring the devices**

   (a) Configuring the **Switch**
   Start the ovs-ctl using the below command.

   ```
   /usr/share/openvswitch/scripts/ovs-ctl start
   ```

   Create a Network bridge and add all required ports in to that bridge.

   ```
   ovs-vsctl add-br br0
   ```

   ```
   ovs-vsctl add-port br0 eth0 -- add-port br0 eth1 \
   -- add-port br0 eth2 -- add-port br0 eth10
   ```

   Use the following command to mirror all traffic IN and OUT from eth1 port to eth10 port on the switch. This dumps all traffic from Client to the monitoring port of the IDS.

   ```
   ovs-vsctl -- set Bridge br0 mirrors=@m \
   -- --id=@eth1 get Port eth1 \
   -- --id=@eth10 get Port eth10 \
   -- --id=@m create Mirror name=mymirror select-dst-port=@eth1 \
   select-src-port=@eth1 output-port=@eth10
   ```

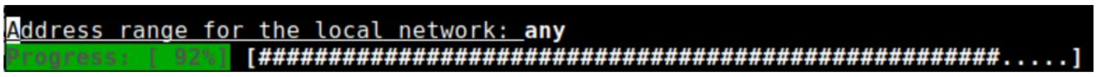
   (b) Configuring the **SnortIDS**

   ```
   apt update
   apt install snort python3 python3-pip -y
   pip3 install gdown
   ```

   If pip3 command gives you errors, run it with the `--break-system-packages` option.

   (c) Configuring the **Client**

   ```
   apt update; apt install nmap -y
   ```



```
Address range for the local network: any
Progress: [ 92%] [###########################################################.....]
```

Figure 4: Configure local network as *any*

3. **Configuring Snort**
   Snort comes with a default set of rules including the Community Ruleset which is developed by the Snort community. It is freely available to all users. These rule files are located at the location */etc/snort/rules/*.

   First of all we will disable these default rules in the snort.config file so we can write our own rules. To do this we have to comment out the the lines starting with **include RULE_PATH** in the Snort config file.

   Use the following command to create a copy of the snort.conf file with the default rules disabled.

   ```
   sed 's|include $RULE_PATH|#&|' /etc/snort/snort.conf \
   > /etc/snort/snort_custom.conf
   ```

Now use the below command to enable only the **local.rules** file where we will be writing our custom rules. (One command)

```
sed -i 's|#include $RULE_PATH/local.rules|include \
$RULE_PATH/local.rules|' /etc/snort/snort_custom.conf
```

Now we will write custom rules in the **/etc/snort/rules/local.rules** file.

```
nano /etc/snort/rules/local.rules
```

4. **Test the IDS with a simple rule** (Single command. Copy the both lines separately)

```
alert icmp any any -> any any (msg:"ICMP test detected"; GID:1;
sid:10000001; rev:001; classtype:icmp-event;)
```

Edit the file **/etc/snort/rules/local.rules**, and add the following line. This rule will give us alert when the IDS see any ICMP traffic in the network:

Run `snort` from command line:

```
snort -i eth1 -v -c /etc/snort/snort_custom.conf -A full
```

Now from the **Client** console ping the Google DNS server 8.8.8.8.

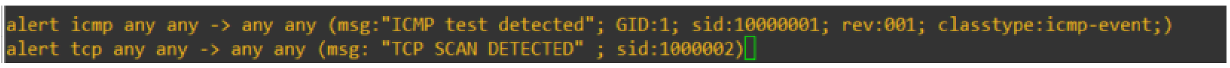
```
ping 8.8.8.8
```

After running the pings for about 10 seconds, stop Snort (Ctrl+C) and view the created alerts in the log file using the below command:

```
tail -n 50 /var/log/snort/alert
```

5. **Make rules for TCP scan (e.g. nmap)**
Let's make a rule which can detect any TCP scan on our system. Edit **/etc/snort/rules/local.rules**, add the following rule:

```
alert tcp any any -> any any (msg: "TCP SCAN DETECTED" ; sid:1000002)
```



```
alert icmp any any -> any any (msg:"ICMP test detected"; GID:1; sid:10000001; rev:001; classtype:icmp-event;)
alert tcp any any -> any any (msg: "TCP SCAN DETECTED" ; sid:1000002)
```

Figure 5: Local Rules

Now use Nmap on Client to scan any external IP and Snort will give us alerts about the TCP connection :

```
nmap 8.8.8.8
```

Stop Snort and view the log file to see the alerts.

```
tail -n 50 /var/log/snort/alert
```

Your Turn: The above rule generates an alert for any TCP traffic, not only scan traffic. How can you modify the above to alert only if 50 or more SYN packets are seen from the same source within 10 seconds? Implement the rule and test.

6. **Inspect offline traffic dump**
   Use the below command to download a dump of network traffic with some suspicious activities.

   ```
   gdown 117L6zjh4udhokJlIhsVEncz4_QEY7M0x
   ```

   Snort offers a parameter (-r) to specify the local traffic dump to be inspected. It will scan and apply the rule in the traffic dump file and output the inspection result. Notice that we are using the original **/etc/snort/snort.conf** where all in-built rules are also enabled.

   ```
   snort -c /etc/snort/snort.conf -r traffic.pcap -A full
   ```

   View the results in log file.

   ```
   tail -n 50 /var/log/snort/alert
   ```

2 Group Discussion and Reflection:

1. How can Snort be configured to function as an IPS instead of an IDS? Does this require changing Snort's placement in the network topology?
2. How do non-inline IPS systems operate in real-world environments?