# Task 5: NIDS Section

## Objectives:

- Understand how Security Onion utilizes Suricata to function as a NIDS.
- Explore how a SOC Analyst could utilize the **Alerts**, **Cases**, **Hunt**, **PCAP** Interfaces to investigate suspicious network traffic.
- Explore how a Security Onion function as an NIDS
    - Check the pre-existing rules in Suricata.
    - Add local rules to generate new alerts.
    - Tune the pre-existing rules to avoid Alert Fatigue.

### NIDS

NIDS stands for Network Intrusion Detection System. It is a means of monitoring network traffic, looking for specific activity, and generating alerts.

### Usage

Security Onion can run either **Snort** or **Suricata** as its Network Intrusion Detection System (NIDS). Security Onion generates NIDS (Network Intrusion Detection System) alerts by monitoring your network traffic and looking for specific fingerprints and identifiers that match known malicious, anomalous, or otherwise suspicious traffic. This is signature-based detection so you might say that it's similar to antivirus signatures for the network, but it's a bit deeper and more flexible than that. For this lab setup, NIDS alerts are generated by Suricata.

### Suricata

Suricata is a free and open source, mature, fast and robust network threat detection engine. Suricata, similar to Snort, inspects the network traffic using powerful and extensive rules and signature language for detection of complex threats.

## Task 5.1: Examine the Pre-existing Rules and the Cases Interface

**5.1.1:** From the Attacker VM ping the Server VM

**5.1.2:** Check the **Alerts Interface** and describe your observations in detail. Include the following information: the rule itself, the rule name, the rule category, rule action and and rule severity level

**5.1.3:** Can you explain the differences between rule category, rule UID, rule ruleset, and rule rev? Please provide examples and support your answer by checking online resources.

**5.1.4:** As a Cybersecurity Analyst who is monitoring the network, you think this alert requires further investigation. How would you escalate this alert to be investigated more in-depth? Describe the steps you would take to create a new case? assign the case to someone, and change the case status to 'In progress'. Change the severity level of this case to “Critical”, and set the priority of this case to 2. Lastly, set TLP to “Not for disclosure”

**5.1.5:** Can you describe the different levels of TLP (Traffic Light Protocol) security and provide examples of when an analyst should use each level? Additionally, can you explain the purpose of the PAP (Permissible Actions Protocol) and describe how the Analyst should utilize it? Please provide examples and support your answer by checking online resources.

**5.1.6:** While you are investigating this alert you notice that the number of instances of this specific alert is concerning. Therefore, make an observation about the count of this alert. Set the type of the observable to “Other”. Set this observable to “an indication of compromise”. Set the TLP accordingly.

**5.1.7:** You can find all NIDS rules of Security Onion in “/opt/so/rules/nids/all.rules”. Examine the alert generated by the ping command and find the rule.uuid. Use the rule uuid to search for the rule within the “all.rules” file.

## Task 5.2: Examine the Pre-existing Rules and the Hunt and PCAP interfaces

**5.2.1:** From the  Attacker VM, you should perform an NMAP scan to identify the services running on the Server VM.

**5.2.2:** From the Analyst VM,  check **Alerts Interface** and describe your observations

**5.2.3:** A Cybersecurity Analyst who uses Security Onion can have multiple other approaches to conduct a further investigation other than just the **Cases Interface**.

- Right click on any of the NMAP alerts, you will be able to see a set of actions: Hunt, CyberChef, Google, Virus Total.  Escalate this alert to the **Hunt Interface**.

**5.2.4:** Explain the difference between the **Dashboard Interface** and the **Hunt Interface**. Please provide examples and support your answer by checking online resources.

**5.2.5:** Once you are in the Hunt Interface, Check the Events Table at the end of the page. Maneuver the mouse pointer to “source.ip” and then right click on that alert. You could see that one of the actions provided is “Hunt”. Click on that action and explain your observation.

- To answer this question correctly,  it could be helpful to check the query that was generated in the query field and explain it.
- Also set the time frame to “1 month ago” and make sure to include the number of “Total Found” and explain what that number stands for in this case.

**5.2.6:** Explain how the “**Hunt**” feature (not the interface), could be beneficial for a Cybersecurity Analyst in general.

## Task 5.3:  Add Local Rules

**5.3.1:** Perform NMAP FIN Scan from the Attacker VM against the Server VM and check if there are any alerts triggered by this event.

**5.3.2:** Check the **Dashboard Interface**, write a query that displays only the network traffic with the source IP address of the Attacker VM. Set the time frame to the previous couple of minutes only. Explain the traffic shown in the Events Table? What is the difference between this table (in the **Dashboard Interface**) and the table in the **Alerts Interface**?

**5.3.3:** Let's add a new rule to detect NMAP FIN Scan to "/opt/so/saltstack/local/salt/idstools/local.rules".

- $ sudo su

**5.3.4:** Write your rule that if any machine within the IP Address of the Monitor Network was scanned through any port, using **NMAP FIN**, an alert will be generated with the message " Nmap Fin Scan ''.

**5.3.5:** There is a specific range of SID that is reserved for local rules. If you use a SID that is allocated for another group of rules, you will overwrite a pre-existing rule. Perform a quick search to find how SID are allocated and what is the range reserved for local use only.

**5.3.6:** From the manager, tell Salt to update:

- $ sudo salt-call state.highstate

**5.3.7:** Update rules:

- $ sudo so-rule-update
- At this window, if you correctly set sn SID that have not been previously used, you will notice that the number of added rules: 1

**5.3.8:**Restart Suricata (replacing $SENSORNAME_$ROLE as necessary):

- sudo salt $SENSORNAME_$ROLE  state.apply Suricata
- $SENSORNAME_$ROLE= securityonionlab_eval

**5.3.9:** You can then run NMAP FIN scan to generate traffic which should cause this rule to raise an alert. Test that the rule is working for both the Server and the User VMs to show that your rule will raise an alert for any machine within the Monitor Network not just for a specific IP Address.

## Task 5.4:  Tune a Pre-existing Rule

**5.4.1:** What is the meaning of the term 'Alert Fatigue' in the context of cybersecurity?

**5.4.2:** How can Alert Fatigue impact the effectiveness of a cybersecurity analyst's work, and what are some steps they can take to avoid it?

**5.4.3:** In addition to adding new local rules, SOC Analysts may have to update a pre-existing rule. Let's say that for this specific environment, the SOC Analysts are experiencing alert fatigue due to the high number of alerts generated by the NMAP scan.

- Pick one of the rules that are triggered by an NMAP Scam. Examine the details of that rule by checking the **Alerts Interface**
- Modify that rule and include it in the “local.rules” file. You should tune down the rule threshold.
- You are supposed to use the same SID. However, there is one of the rule features that you must modify in order for Suricata to use this modified version of the rule instead of the original one.
- Test your rule and explain your observations

**5.4.5:** In some cases the SOC Analyst may need to disable the entire rule. To perform this in Security Onion you can use “sudo so-rule”. Refer to the documentation of Security Onion and disable one of the pre-existing rules.

**5.4.6:** Show proof that the rule was successfully disabled.