Intrusion Detection System Using Snort: Attack Detection Lab

Author: Jagruth P
Date: 6th Jun 2025

📘 Abstract:

This project presents a practical implementation of a Network Intrusion Detection System (NIDS) using **Snort**. A multi-VM lab was configured using Ubuntu (Snort IDS), Kali Linux (Attacker), and Metasploitable (Victim). The goal was to simulate common cyberattacks and detect them using Snort's custom rules.

The project involved port scanning, ICMP flood, brute-force attacks, and SQL injection. Snort was configured to analyze traffic and raise alerts when suspicious behavior was detected. The exercise enhanced understanding of real-world network threats, IDS rule creation, and log analysis. Screenshots, detection logic, and explanations are included to demonstrate Snort's capability in monitoring and detecting malicious activity.

1. Introduction

Cybersecurity requires proactive defense mechanisms to detect threats. One such mechanism is Intrusion Detection Systems (IDS), which monitor traffic for signs of malicious activity. **Snort**, an open-source NIDS, is capable of packet sniffing, traffic analysis, and alert generation.

This project aimed to configure and use Snort to detect multiple types of attacks. A lab was created using virtualization tools, simulating real-world traffic scenarios between an attacker (Kali), a vulnerable target (Metasploitable), and an IDS (Ubuntu). This documentation outlines the setup, rules written, attacks simulated, and Snort's response.

2. Lab Setup

Component	Details
Ubuntu VM	Snort IDS, Interface: ens33
Kali VM	Attacker tools: Nmap, Hydra, curl
Metasploitable	Vulnerable target with FTP, DVWA, Mutillidae
Network Mode	Bridged Adapter (all VMs same subnet)

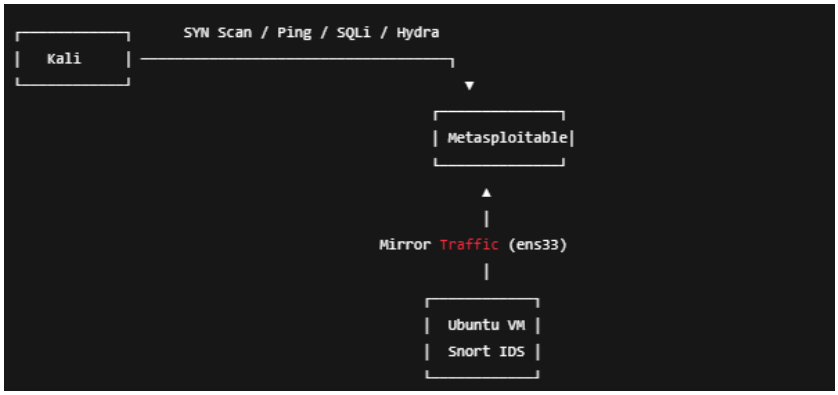


Fig 2.1: Lab Structure

- Installed via APT on Ubuntu

- Interface: `ens33`

- Config file: `/etc/snort/snort.conf`

- Rules file: `/etc/snort/rules/local.rules`

Ubuntu IP: 192.168.146.135

Kali IP: 192.168.146.136

Metasploitable IP: 192.168.146.129

3. Custom Snort Rules Used

SID	Purpose	Description
1000001	Port Scan Detection	Detects SYN scans on ports 1–1024
1000002	ICMP Flood Detection	Detects oversized ICMP requests
1000004	SQL Injection	Detects HTTP requests with ' or 1=1
1000005	FTP Brute Force	Detects repeated connection attempts

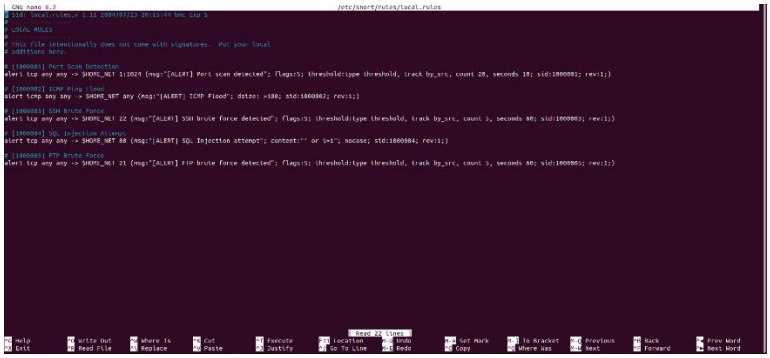


Fig 3.1: Local Snort rules

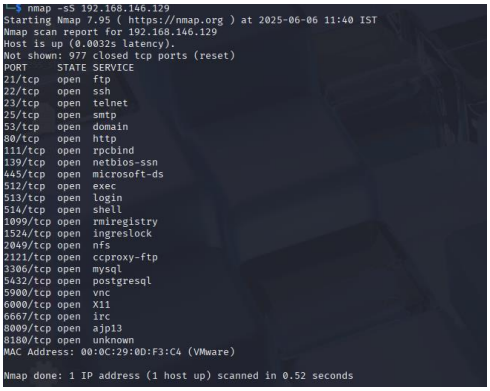
Each rule is stored in local.rules. The Snort configuration was validated using:

```
sudo snort -T -c /etc/snort/snort.conf -i ens33
```

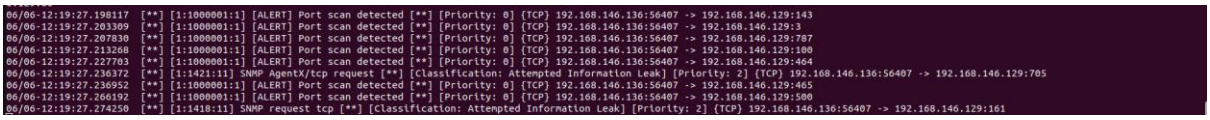
4. Attack Simulations

📌 Port Scan

- **Command:** nmap -sS 192.168.146.129
- **Snort Alert:** [1000001] Port scan detected
- **Screenshots:**



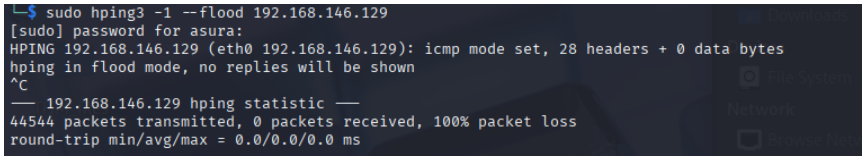
Screenshot 1: Nmap scan result from Kali



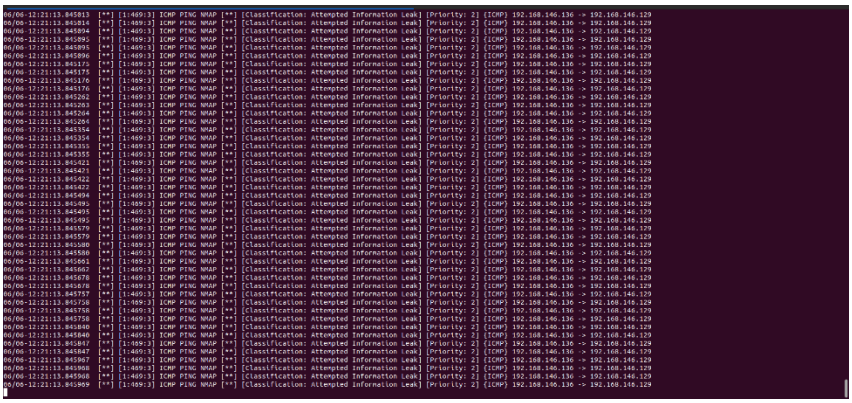
Screenshot 2: Snort console showing port scan alert

📌 ICMP Flood

- **Command:** hping3 -1 --flood 192.168.146.129
- **Snort Alert:** [1000002] ICMP Flood
- **Screenshots:**



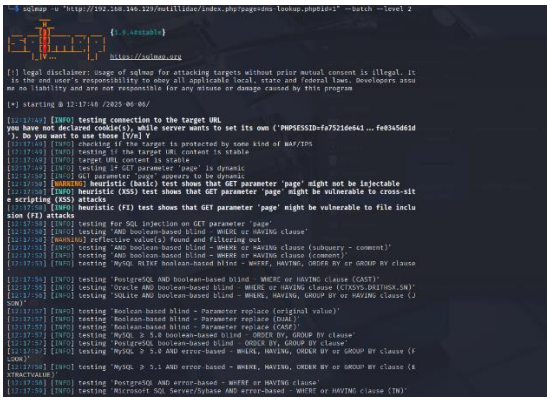
Screenshot 1: ICMP flood from Kali



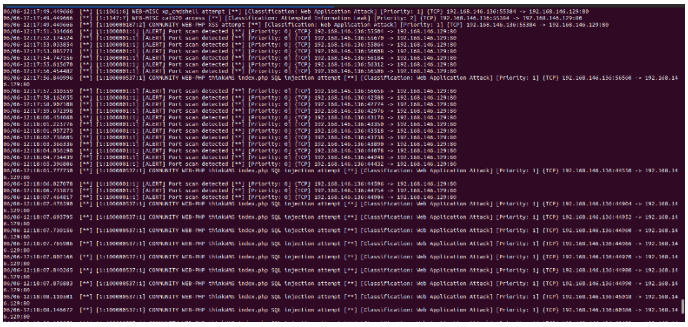
Screenshot 2: Snort console showing ICMP Flood alert

📌 SQL Injection (Fake & Real)

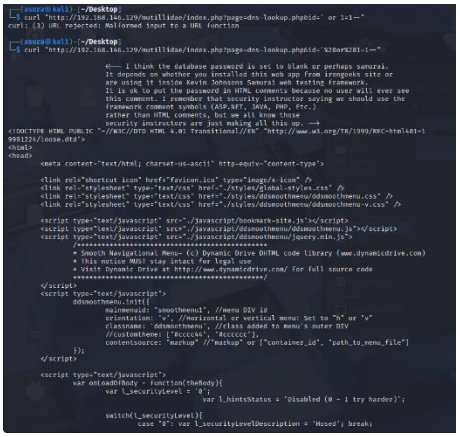
- **Command (curl - fake):** curl "http://192.168.146.129/mutillidae/index.php?page=dns-lookup.php&id=' or 1=1--"
- **Command (sqlmap - real):** sqlmap -u "http://192.168.146.129/mutillidae/index.php?page=dns-lookup.php&id=1" --batch --level 2
- **Snort Alert:** [1000004] SQL Injection attempt
- **Screenshots:**



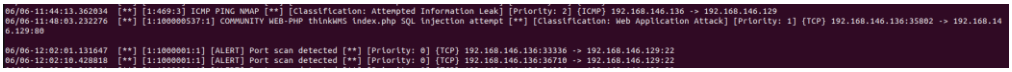
Screenshot 1:SQL (real) Injection Attack from Kali using sqlmap



Screenshot 2: Snort console showing web app attack alert



Screenshot 3: HTTP flood from Kali



Screenshot 4: Snort console showing Fake SQLi alert

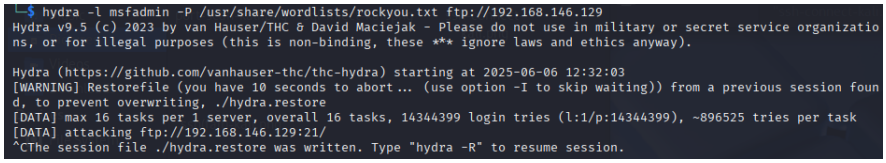
📌 FTP Brute Force (used instead of SSH)

- **Command:**

  hydra -l msfadmin -P /usr/share/wordlists/rockyou.txt ftp://192.168.146.129 -t 4

- **Snort Alert:** [1000005] FTP brute force detected

- **Screenshots:**



Screenshot 1: FTP attack(ongoing) from Kali

Screenshot 2: Snort console showing FTP Bad Login alert

5. Observations

- Port scans triggered alerts within seconds using SYN flags.
- ICMP floods generated multiple alerts due to oversized packets.
- SQLi alerts matched ' or 1=1 in raw HTTP payloads.
- FTP brute-force was used in place of SSH (due to OpenSSH restrictions).
- All traffic was monitored via ens33 from Ubuntu Snort VM.

6. Challenges Faced

SSH Brute Force Blocked: Kali's OpenSSH client (v9+) did not support Metasploitable's outdated ssh-dss keys. Workarounds like ~/.ssh/config and /etc/ssh/ssh_config failed.

✅ **Workaround Used:** Switched to FTP brute-force (vsftpd) which behaves similarly for login attempts and can be detected using a Snort threshold rule.

7. Conclusion

Snort effectively detected all four attack types when properly configured. This project helped solidify the fundamentals of intrusion detection, rule writing, and traffic simulation. The practical experience gained through VM lab setup and live testing is highly relevant for SOC roles, red team simulations, and secure network design.