EncryptEdge Labs

# Cybersecurity Analyst Internship

# Task Report

atalmamun@gmail.com

Task No: 13



Credit: Offensive Security

EncryptEdge Labs

# Table of Contents


# 1.0 EncryptEdge Labs Internship Task Report

## 1.1 Introduction

Intrusion Detection Systems (IDS) play a critical role in modern cybersecurity strategies, acting as a safeguard against unauthorized access, malicious activities, and other security breaches within a network or host system. As cyber threats continue to evolve in complexity and frequency, the need for proactive monitoring tools like IDS becomes increasingly essential. This task explores the fundamental concepts of IDS, their classification, and practical implementation using open-source tools. Through both theoretical study and hands-on experience, the goal is to gain a solid understanding of how IDS helps in identifying, alerting, and potentially mitigating intrusions before they can cause significant harm.

## 1.2 Objective

The primary objectives of this task are:

- To understand the role and significance of Intrusion Detection Systems in cybersecurity.
- To differentiate between Host-based IDS (HIDS) and Network-based IDS (NIDS), including their respective benefits and limitations.
  To explore and compare detection techniques such as signature-based and anomaly-based methods.
- To configure and test an IDS tool (Snort) within a virtualized environment and analyze its ability to detect simulated threats.
- To complete hands-on labs for practical exposure to IDS concepts and implementation.

## 1.3 Requirements

To complete this task, the following tools and environments are required:

- **Operating System:** Kali Linux (or a similar Linux-based virtual machine)
- **IDS Tool:** Snort (or any other open-source IDS)
- **Virtual Environment:** VirtualBox, VMware, or another VM platform to simulate the network environment
- **Online Lab Platform:** TryHackMe (for IDS and Snort labs)
- **Internet access:** For downloading tools, accessing resources, and completing online labs

# 2.0 Theoretical Understanding of Intrusion Detection Systems

Intrusion Detection Systems (IDS) are essential tools in cybersecurity that monitor and analyze network or system activities for signs of malicious behavior or policy violations. They serve as an additional layer of defense, offering early detection of potential intrusions and allowing system administrators to respond before damage is done. IDS solutions are broadly categorized into Host-based and Network-based systems, each with distinct use cases, advantages, and limitations.

## 2.1 Role of IDS in Cybersecurity

The primary function of an IDS is to detect suspicious activity and alert system administrators to potential threats. This enables real-time monitoring of network traffic or host-level operations to identify known attack patterns or deviations from normal behavior. IDS tools do not typically block attacks (unless paired with an Intrusion Prevention System, or IPS), but they are critical in recognizing early signs of compromise.

Key roles of IDS in cybersecurity include:

- Monitoring network traffic or host activity for anomalies.
- Alerting administrators about unauthorized access attempts, malware, or data breaches.
- Helping in forensic analysis by logging suspicious events.
  Supporting compliance with cybersecurity standards and regulations.

By providing visibility into network or host activity, IDS tools help organizations improve their overall security posture and incident response capabilities.

## 2.2 Host-Based IDS (HIDS) vs. Network-Based IDS (NIDS)

| Feature | Host-Based IDS (HIDS) | Network-Based IDS (NIDS) |
|---|---|---|
| **Monitoring Scope** | Individual hosts or endpoints | Entire network traffic across multiple devices |
| **Installation Point** | Installed on each monitored system | Deployed at strategic points within the network |
| **Data Analyzed** | System logs, file integrity, application behavior | Network packets, protocol headers, traffic patterns |
| **Advantages** | Detailed insights into specific system behavior<br><br>Can detect internal threats<br><br>Effective against encrypted threats (if decrypted on host) | Broad visibility of traffic<br><br>Centralized monitoring<br><br>Efficient in detecting widespread attacks |

| | | |
|---|---|---|
| **Limitations** | Resource-intensive | Cannot inspect encrypted traffic |
| | Difficult to manage at scale | May miss host-level anomalies |
| | Limited visibility beyond host | Blind to local events on individual machines |
| **Use Cases** | Detecting privilege escalation, unauthorized file changes, insider threats | Monitoring for port scans, malware communications, DDoS attempts |

### 2.3 Significance of IDS in Network Security

An effective IDS serves as an early warning system, reducing the risk of undetected breaches and minimizing response time. In today's threat landscape, where attackers often bypass traditional defenses, IDS solutions provide critical insight and forensic data that can be used to investigate incidents and strengthen defenses. Whether deployed at the host or network level, IDS systems contribute to layered security by adding detection capabilities that complement firewalls, antivirus software, and other protective measures.

## 3.0 Intrusion Detection Techniques

Intrusion Detection Systems utilize various techniques to identify potential threats and malicious behavior. Two of the most commonly used detection methods are **signature-based detection** and **anomaly-based detection**. Each approach has its own strengths and weaknesses, and many modern IDS tools incorporate both for comprehensive coverage.

## 3.1 Signature-Based Detection

Signature-based detection works by identifying known patterns of malicious activity, often referred to as *signatures*. These can include specific sequences of bytes in network packets, command strings used in exploits, or predefined behaviors associated with malware.

**Advantages:**

- **High accuracy for known threats:** Very effective at detecting well-documented attacks.
  **Low false positives:** Because it matches exact patterns, it rarely flags legitimate behavior as malicious.
  **Efficient performance:** Typically requires less processing power compared to anomaly detection.

**Limitations:**

- **No detection of zero-day attacks:** Cannot recognize new, unknown, or modified threats.
- **Frequent updates required:** Signatures need to be updated regularly to stay current.
- **Reactive rather than proactive:** Only responds to known threats, offering limited protection against novel attacks.

**Real-World Example:** A signature-based IDS like Snort can detect the *Code Red* worm by matching a specific HTTP request pattern associated with the worm's propagation method. When a packet matches this pattern, the IDS generates an alert.

## 3.2 Anomaly-Based Detection

Anomaly-based detection identifies threats by monitoring behavior and flagging deviations from a defined "normal" baseline. This technique uses statistical models, machine learning, or heuristics to determine what constitutes typical activity.

**Advantages:**

- **Detects unknown threats:** Can identify new or evolving attack patterns that do not match any known signature.
- **Adaptive:** Capable of learning and evolving with network behavior.
- **Useful for insider threats:** Can detect unusual behavior from authorized users.

**Limitations:**

- **High false positive rate:** Normal but uncommon activity may be flagged as malicious.
- **Complex configuration:** Requires careful tuning and ongoing management.
- **Resource-intensive:** Generally requires more processing power and data analysis.

**Real-World Example:** Anomaly detection might identify an internal user's machine suddenly transferring large volumes of data to an unfamiliar external IP address outside business hours. Although this action doesn't match a known attack signature, it deviates significantly from the user's normal behavior and triggers an alert.

## 3.3 Comparison Summary

| Detection Method | Strengths | Weaknesses | Best Used For |
|---|---|---|---|
| **Signature-Based** | High accuracy for known threats<br>Low false positives | Cannot detect new/unknown attacks<br>Needs constant updates | Known malware and attack pattern detection |

| | | | |
|---|---|---|---|
| **Anomaly-Based** | Detects novel or unknown threats<br><br>Adaptive to environment | Prone to false positives<br><br>Requires tuning and resources | Unusual behavior, insider threats, zero-day attacks |

# 4.0 Practical Implementation of IDS Tool (Snort)

In this section, Snort — an open-source Intrusion Detection System — was installed, configured, and tested on a Kali Linux virtual machine. The objective was to simulate real-world attack scenarios, observe how Snort detects various threats, and analyze its response based on defined rules.

## 4.1 Installation of Snort

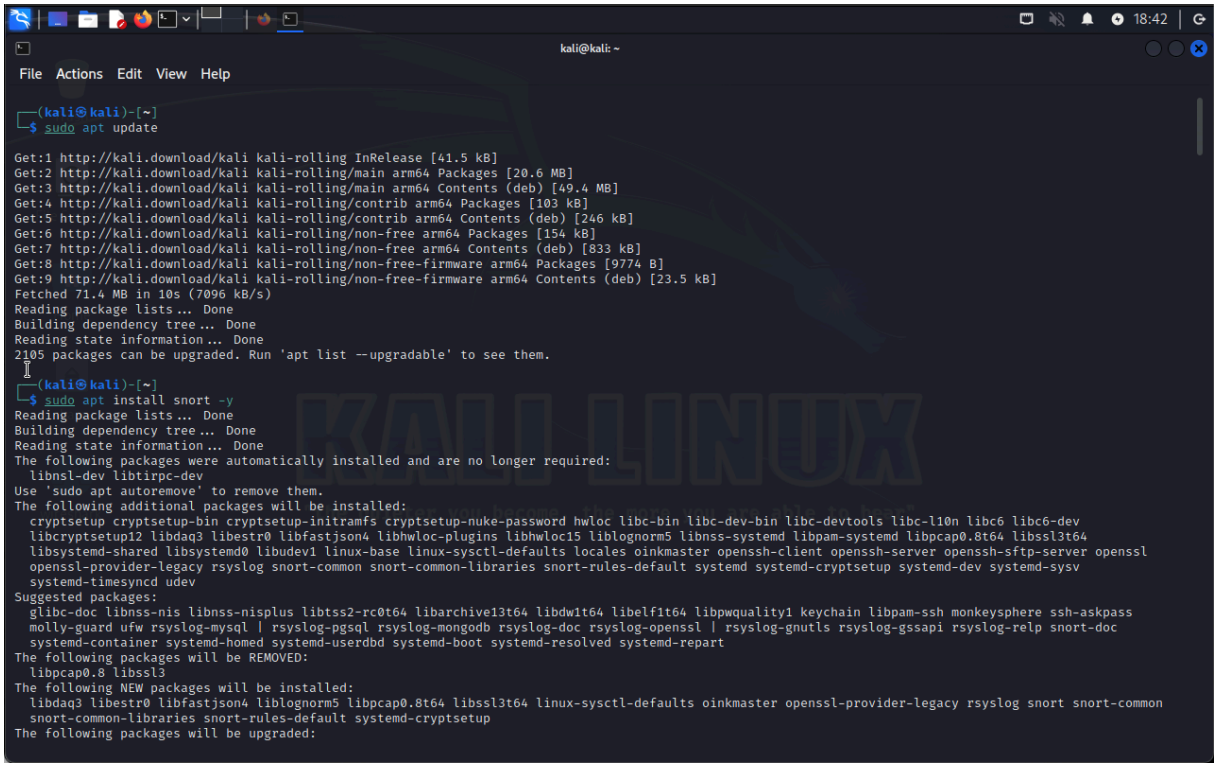
Snort was installed on Kali Linux using the terminal. The installation involved downloading the necessary dependencies, setting up configuration files, and verifying the installation.

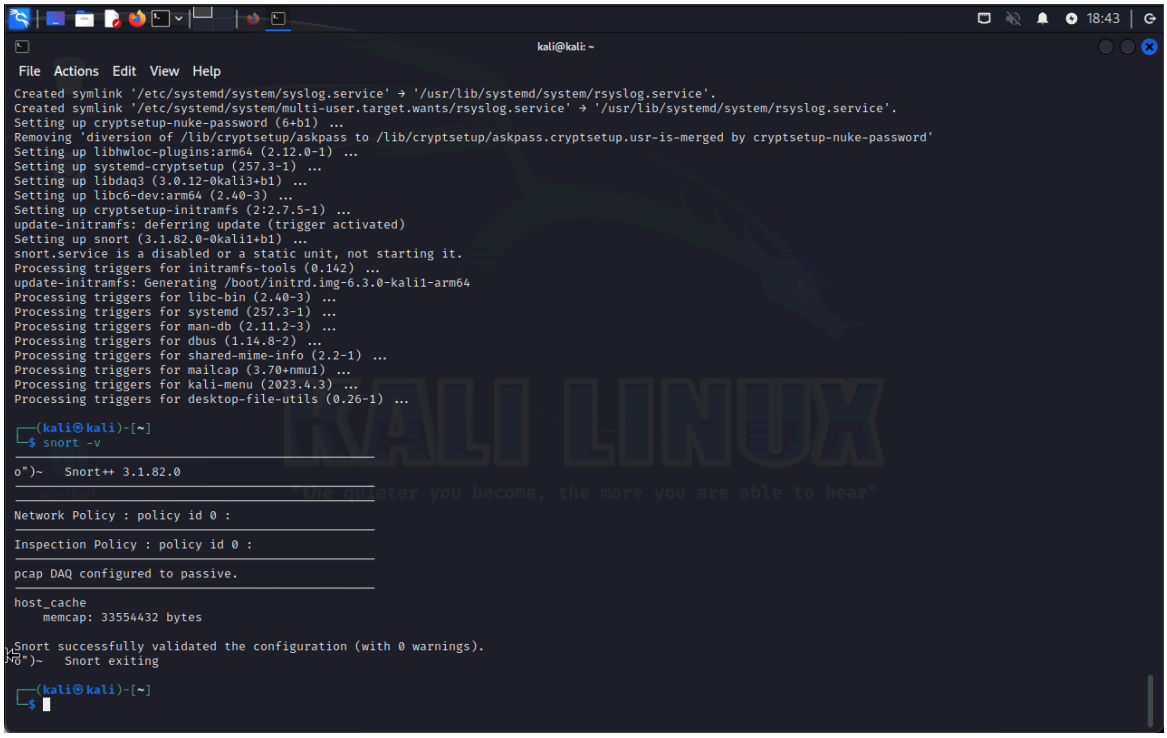
**Installation Steps:**

Update the system: `sudo apt update && sudo apt upgrade`

Install required packages: `sudo apt install snort -y`

Verify Snort installation: `snort -V`


18:42
kali@kali: ~
File Actions Edit View Help
┌──(kali㉿kali)-[~]
└─$ sudo apt update
Get:1 http://kali.download/kali kali-rolling InRelease [41.5 kB]
Get:2 http://kali.download/kali kali-rolling/main arm64 Packages [20.6 MB]
Get:3 http://kali.download/kali kali-rolling/main arm64 Contents (deb) [49.4 MB]
Get:4 http://kali.download/kali kali-rolling/contrib arm64 Packages [103 kB]
Get:5 http://kali.download/kali kali-rolling/contrib arm64 Contents (deb) [246 kB]
Get:6 http://kali.download/kali kali-rolling/non-free arm64 Packages [154 kB]
Get:7 http://kali.download/kali kali-rolling/non-free arm64 Contents (deb) [833 kB]
Get:8 http://kali.download/kali kali-rolling/non-free-firmware arm64 Packages [9774 B]
Get:9 http://kali.download/kali kali-rolling/non-free-firmware arm64 Contents (deb) [23.5 kB]
Fetched 71.4 MB in 10s (7096 kB/s)
Reading package lists ... Done
Building dependency tree ... Done
Reading state information ... Done
2105 packages can be upgraded. Run 'apt list --upgradable' to see them.
┌──(kali㉿kali)-[~]
└─$ sudo apt install snort -y
Reading package lists ... Done
Building dependency tree ... Done
Reading state information ... Done
The following packages were automatically installed and are no longer required:
libnsl-dev libtirpc-dev
Use 'sudo apt autoremove' to remove them.
The following additional packages will be installed:
cryptsetup cryptsetup-bin cryptsetup-initramfs cryptsetup-nuke-password hwloc libc-bin libc-dev-bin libc-devtools libc-l10n libc6 libc6-dev
libcryptsetup12 libdaq3 libestr0 libfastjson4 libhwloc-plugins libhwloc15 liblognorm5 libnss-systemd libpam-systemd libpcap0.8t64 libssl3t64
libsystemd-shared libsystemd0 libudev1 linux-base linux-sysctl-defaults locales oinkmaster openssh-client openssh-server openssh-sftp-server openssl
openssl-provider-legacy rsyslog snort-common snort-common-libraries snort-rules-default systemd systemd-cryptsetup systemd-dev systemd-sysv
systemd-timesyncd udev
Suggested packages:
glibc-doc libnss-nis libnss-nisplus libtss2-rc0t64 libarchive13t64 libdw1t64 libelf1t64 libpwquality1 keychain libpam-ssh monkeysphere ssh-askpass
molly-guard ufw rsyslog-mysql | rsyslog-pgsql rsyslog-mongodb rsyslog-doc rsyslog-openssl | rsyslog-gnutls rsyslog-gssapi rsyslog-relp snort-doc
systemd-container systemd-homed systemd-userdbd systemd-boot systemd-resolved systemd-repart
The following packages will be REMOVED:
libpcap0.8 libssl3
The following NEW packages will be installed:
libdaq3 libestr0 libfastjson4 liblognorm5 libpcap0.8t64 libssl3t64 linux-sysctl-defaults oinkmaster openssl-provider-legacy rsyslog snort snort-common
snort-common-libraries snort-rules-default systemd-cryptsetup
The following packages will be upgraded:


18:43
kali@kali: ~
File Actions Edit View Help
Created symlink '/etc/systemd/system/syslog.service' → '/usr/lib/systemd/system/rsyslog.service'.
Created symlink '/etc/systemd/system/multi-user.target.wants/rsyslog.service' → '/usr/lib/systemd/system/rsyslog.service'.
Setting up cryptsetup-nuke-password (6+b1) ...
Removing 'diversion of /lib/cryptsetup/askpass to /lib/cryptsetup/askpass.cryptsetup.usr-is-merged by cryptsetup-nuke-password'
Setting up libhwloc-plugins:arm64 (2.12.0-1) ...
Setting up systemd-cryptsetup (257.3-1) ...
Setting up libdaq3 (3.0.12-0kali3+b1) ...
Setting up libc6-dev:arm64 (2.40-3) ...
Setting up cryptsetup-initramfs (2:2.7.5-1) ...
update-initramfs: deferring update (trigger activated)
Setting up snort (3.1.82.0-0kali1+b1) ...
snort.service is a disabled or a static unit, not starting it.
Processing triggers for initramfs-tools (0.142) ...
update-initramfs: Generating /boot/initrd.img-6.3.0-kali1-arm64
Processing triggers for libc-bin (2.40-3) ...
Processing triggers for systemd (257.3-1) ...
Processing triggers for man-db (2.11.2-3) ...
Processing triggers for dbus (1.14.8-2) ...
Processing triggers for shared-mime-info (2.2-1) ...
Processing triggers for mailcap (3.70+nmu1) ...
Processing triggers for kali-menu (2023.4.3) ...
Processing triggers for desktop-file-utils (0.26-1) ...
┌──(kali㉿kali)-[~]
└─$ snort -v
o")~ Snort++ 3.1.82.0
Network Policy : policy id 0 :
Inspection Policy : policy id 0 :
pcap DAQ configured to passive.
host_cache
memcap: 33554432 bytes
Snort successfully validated the configuration (with 0 warnings).
o")~ Snort exiting
┌──(kali㉿kali)-[~]
└─$

## 4.2 Snort Configuration for Intrusion Detection

Snort was configured to detect basic network attacks such as port scans and suspicious packet payloads. Custom rules were written and tested using a sample configuration.

**Steps Taken:**

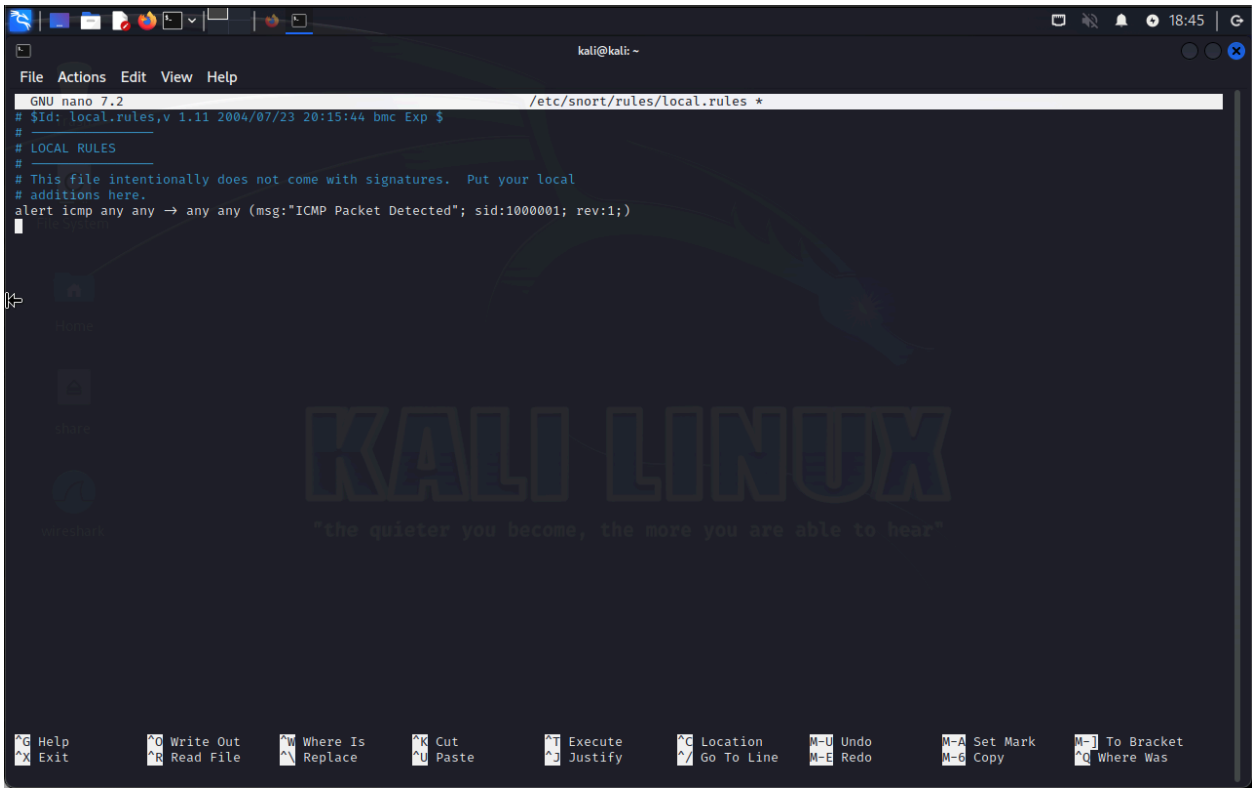
Created a test configuration file: `sudo nano /etc/snort/rules/local.rules`

Added custom Snort rule to detect ICMP ping:

```
alert icmp any any -> any any (msg:"ICMP Packet Detected"; sid:1000001; rev:1;)
```

Edited the Snort configuration file to include the custom rule:

```
include $RULE_PATH/local.rules
```



### 4.3 Simulating Intrusion Scenarios

To test the Snort setup, several simulated attacks were carried out:

**Ping scan using Nmap:**

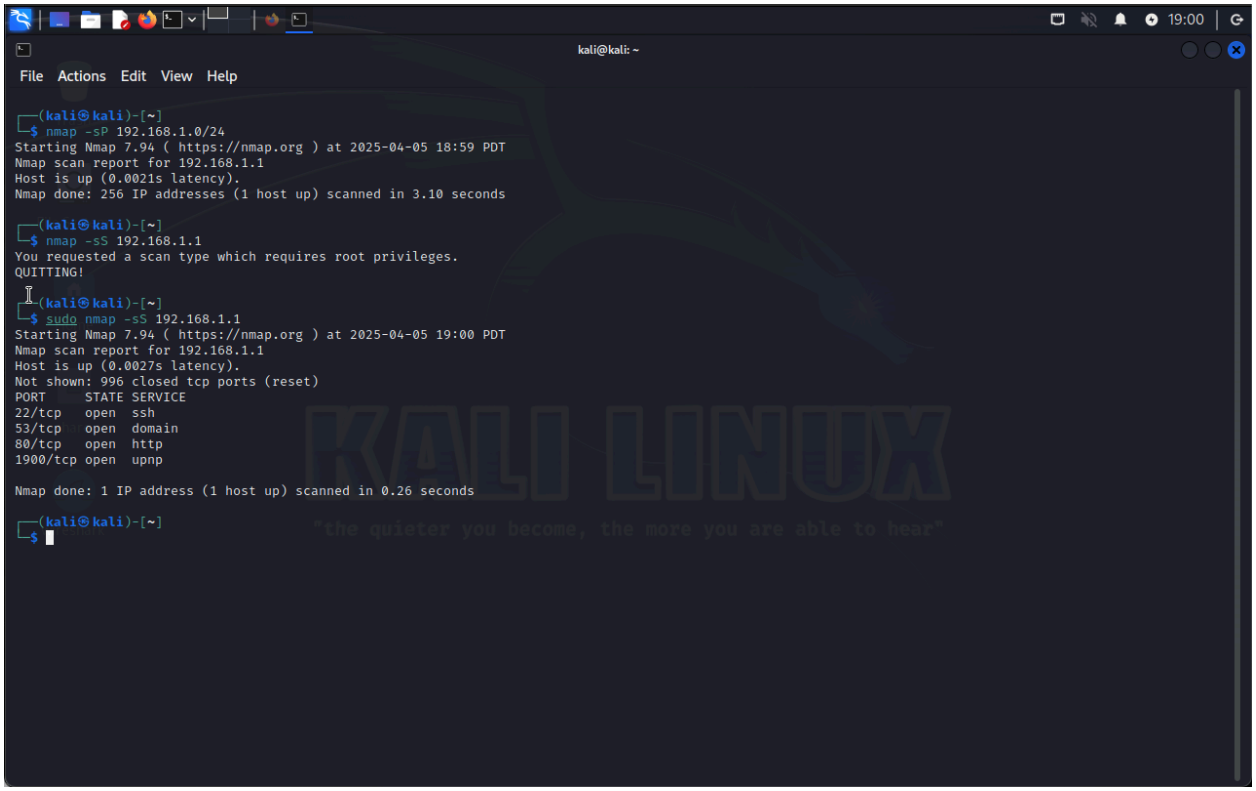
```
nmap -sP 192.168.1.0/24
```

**Port scan:**

```
nmap -sS 192.168.1.105
```

Snort was run in IDS mode to monitor and capture alerts:

```
sudo snort -A console -q -c /etc/snort/snort.conf -i eth0
```



### 4.4 Analysis of Detected Intrusions

The alerts generated by Snort were analyzed to determine the effectiveness of the rule set and configuration. Each alert provided detailed information including the source IP, destination IP, protocol used, and the triggered rule.

**Sample Alert Breakdown:**

- **Message:** ICMP Packet Detected
  **SID:** 1000001
- **Classification:** Misc activity
  **Priority:** 3

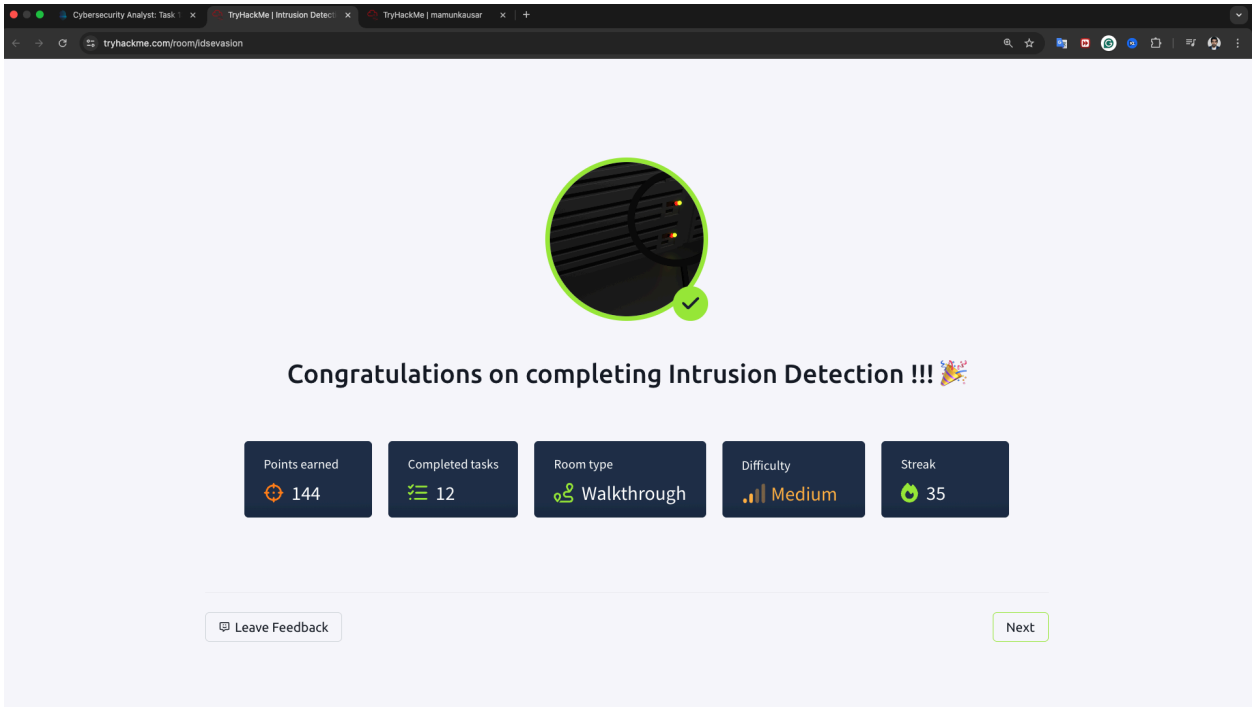
These results confirmed that Snort was functioning correctly and capable of identifying basic forms of suspicious activity in real-time.

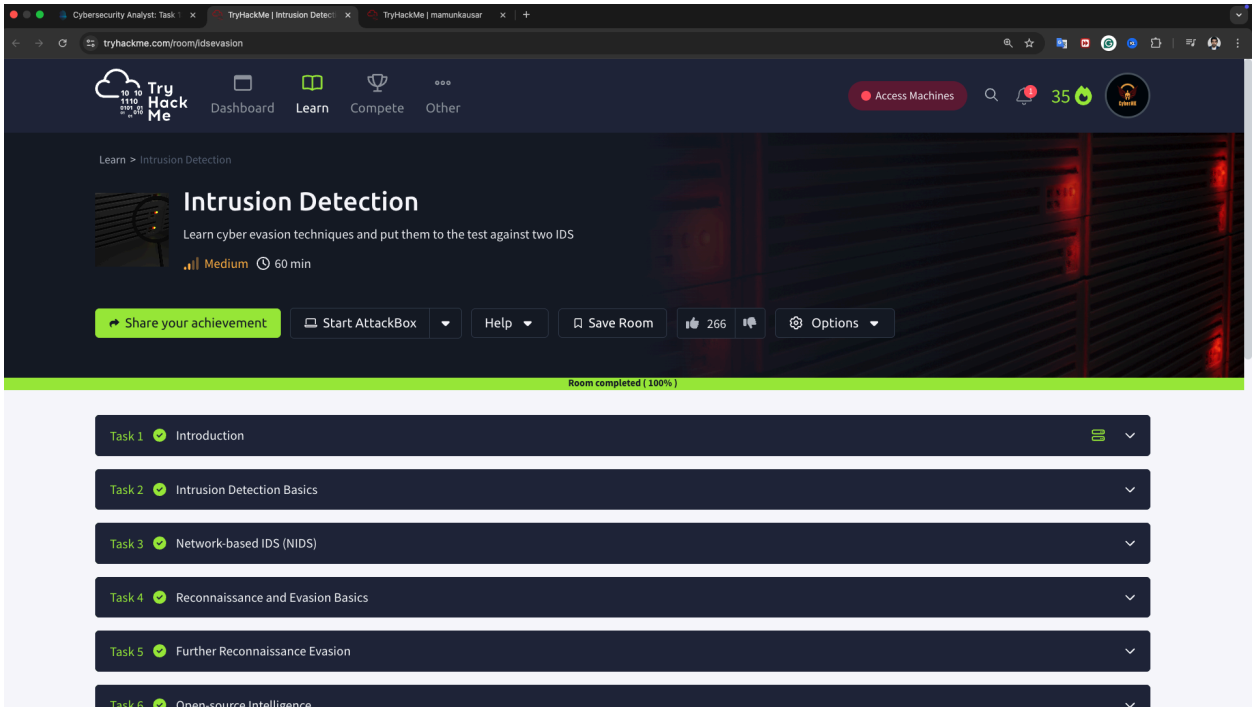
## 5.0 Lab Completion Proof

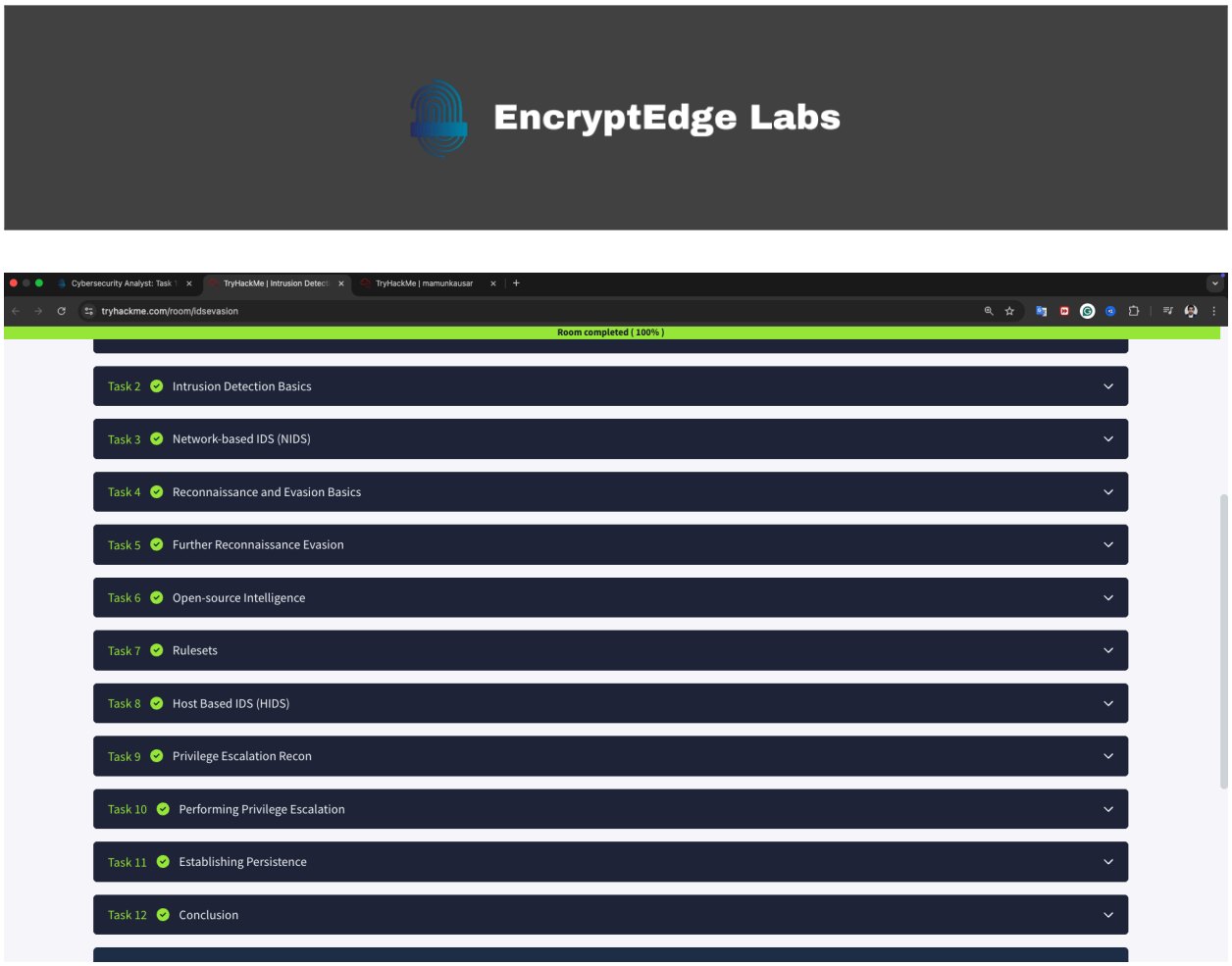
As part of this task, two hands-on labs from TryHackMe were completed to reinforce theoretical understanding and practical application of IDS tools and techniques. These labs provided interactive environments to simulate attacks, configure Snort, and observe how IDS detects malicious behavior.

### 5.1 TryHackMe Lab: Intrusion Detection System

This lab introduced the core concepts of Intrusion Detection Systems, including different types of IDS, detection methods, and basic evasion techniques. It included several exercises focused on identifying threats and interpreting alert outputs.


tryhackme.com/room/idsevasion
Congratulations on completing Intrusion Detection !!! 🎉
Points earned
144
Completed tasks
12
Room type
Walkthrough
Difficulty
Medium
Streak
35
Leave Feedback
Next

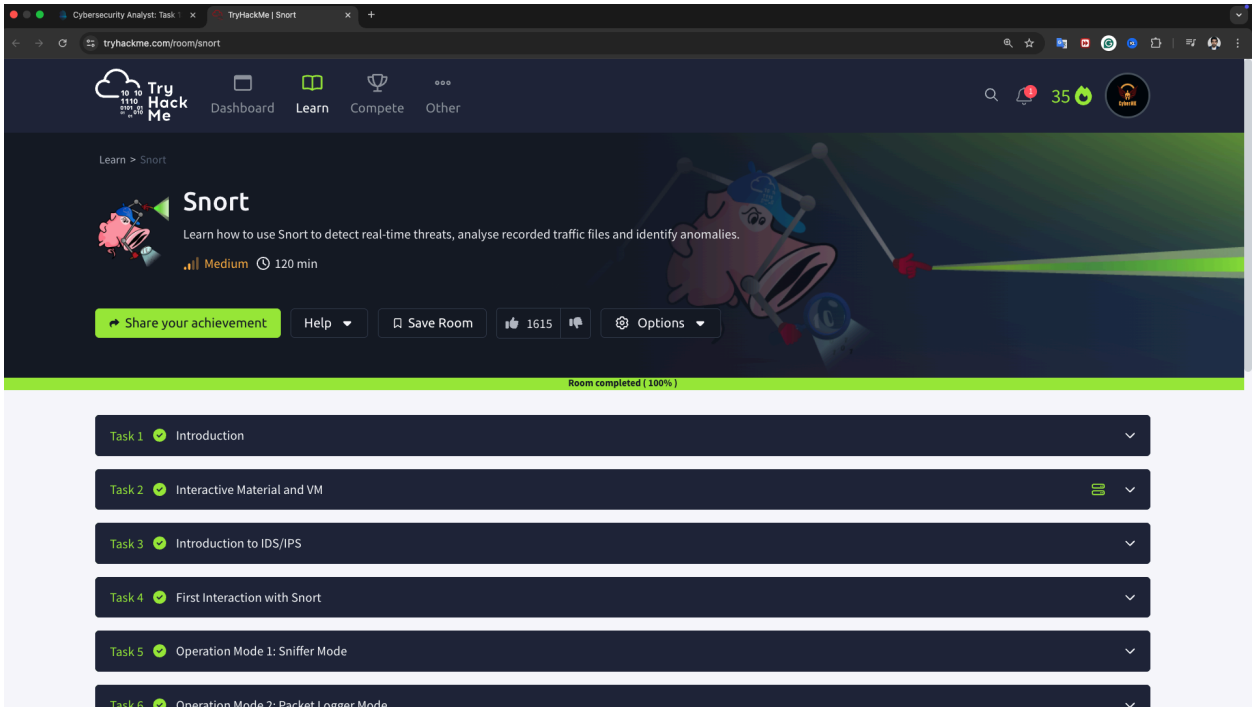

tryhackme.com/room/idsevasion
Try Hack Me
Dashboard
Learn
Compete
Other
Access Machines
35
Learn > Intrusion Detection
Intrusion Detection
Learn cyber evasion techniques and put them to the test against two IDS
Medium
60 min
Share your achievement
Start AttackBox
Help
Save Room
266
Options
Room completed ( 100% )
Task 1 Introduction
Task 2 Intrusion Detection Basics
Task 3 Network-based IDS (NIDS)
Task 4 Reconnaissance and Evasion Basics
Task 5 Further Reconnaissance Evasion

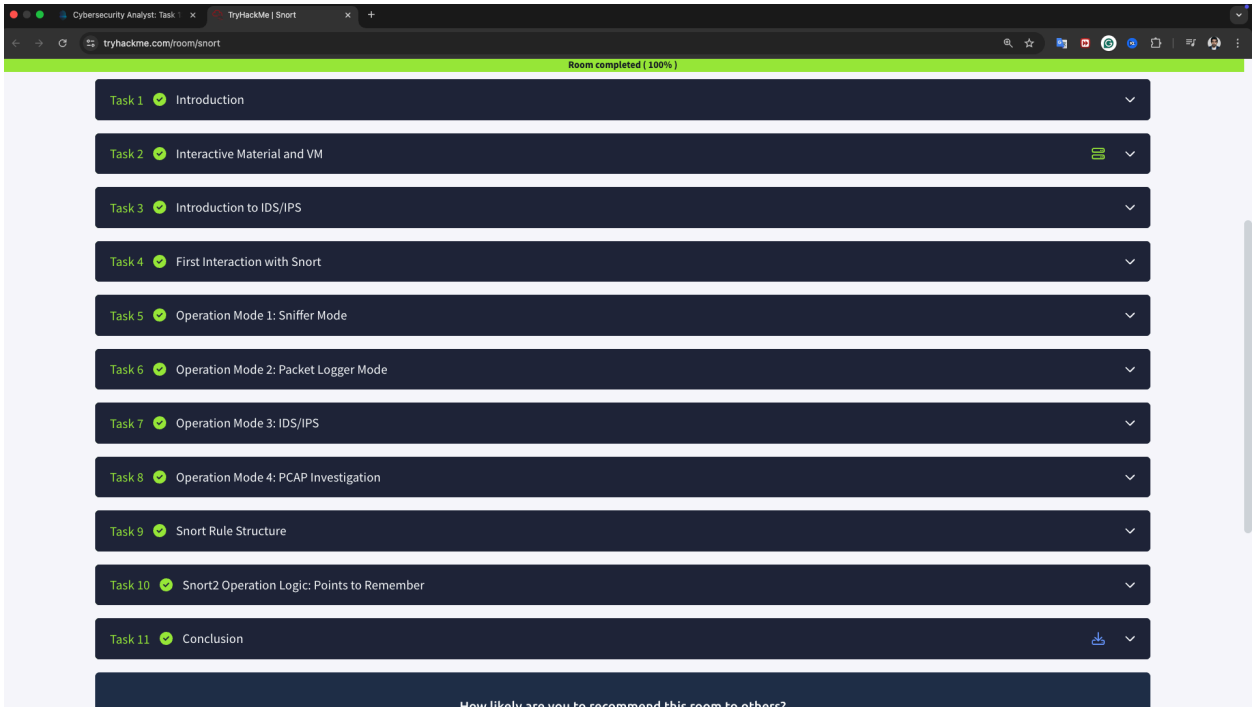


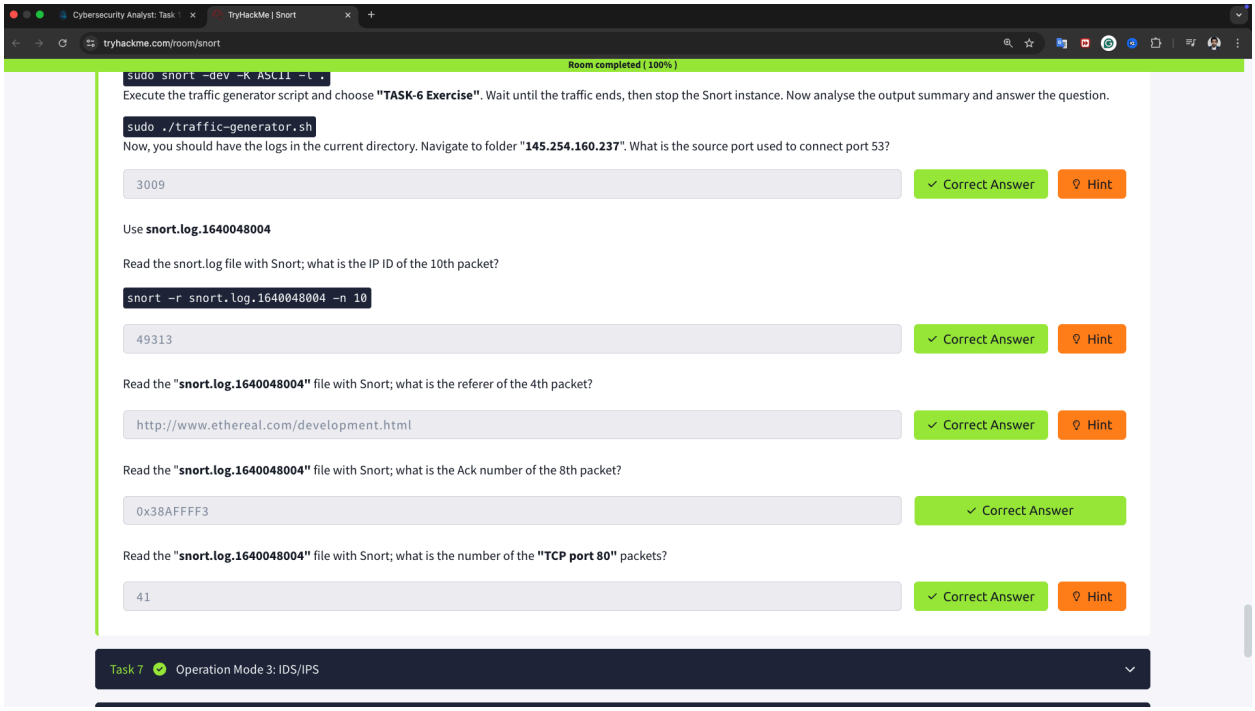

## 5.2 TryHackMe Lab: Snort

The Snort lab provided an in-depth, hands-on walkthrough of installing and configuring Snort in a controlled environment. It included tasks like:

- Writing and testing Snort rules
- Running Snort in packet-logging and alert modes
- Analyzing alert outputs from simulated attacks


Cybersecurity Analyst: Task
TryHackMe | Snort
tryhackme.com/room/snort
Try Hack Me
Dashboard
Learn
Compete
Other
35
Learn > Snort
Snort
Learn how to use Snort to detect real-time threats, analyse recorded traffic files and identify anomalies.
Medium
120 min
Share your achievement
Help
Save Room
1615
Options
Room completed ( 100% )
Task 1 Introduction
Task 2 Interactive Material and VM
Task 3 Introduction to IDS/IPS
Task 4 First Interaction with Snort
Task 5 Operation Mode 1: Sniffer Mode


Cybersecurity Analyst: Task
TryHackMe | Snort
tryhackme.com/room/snort
Room completed ( 100% )
Task 1 Introduction
Task 2 Interactive Material and VM
Task 3 Introduction to IDS/IPS
Task 4 First Interaction with Snort
Task 5 Operation Mode 1: Sniffer Mode
Task 6 Operation Mode 2: Packet Logger Mode
Task 7 Operation Mode 3: IDS/IPS
Task 8 Operation Mode 4: PCAP Investigation
Task 9 Snort Rule Structure
Task 10 Snort2 Operation Logic: Points to Remember
Task 11 Conclusion




These labs helped solidify the understanding of IDS architecture, rule creation, and real-time monitoring with Snort. They also demonstrated the practical use of IDS tools in identifying threats and securing network infrastructure.

**EncryptEdge Labs**

**This Internship Task report was developed on [April, 06, 2025]**

**By:**

**atalmamun@gmail.com**