Snort IDS Project : 1 Report

Title: Design and Implementation of Rule-Based Network Intrusion Detection System using Snort

Author: Taranjyot Kaur Kathoda

With the growing number of cyber attacks on both large and small-scale networks, Intrusion Detection Systems (IDS) have emerged as a significant component of network security. This project deals with the development of a rule-based Network IDS using Snort, a free IDS/IPS solution. The objective of this project is to develop and test customized Snort rules for identifying common network attacks and malicious activities. In a lab setting, 10 Snort rules were developed to identify attack patterns like ICMP flooding, HTTP misuse, DNS queries, insecure services, web-based attacks, and port scanning. The outcome of this project has demonstrated that Snort can successfully identify different types of network threats if configured correctly.

1. Introduction

Network security has become a critical concern due to the rapid growth of connected systems and internet-based services. Intrusion Detection Systems (IDS) help monitor network traffic and identify malicious or policy-violating activities. Snort is one of the most widely used open-source IDS/IPS tools due to its lightweight architecture, flexibility, and powerful rule-based detection engine.

This project aims to provide a practical implementation of Snort by configuring and testing custom detection rules. The focus is on understanding how Snort rules work internally and how they can be used to detect real-world attacks at the network level.

2. Objectives of the Project

- To understand the working of Snort as an IDS
- To configure Snort in a Linux-based environment
- To write and deploy custom Snort rules
- To detect and analyze different types of network attacks
- To validate the effectiveness of rule-based intrusion detection

3. Tools and Technologies Used

- **Snort** – Intrusion Detection System
- **Kali Linux** – IDS monitoring machine

- **Parrot OS / Linux Mint** – Attacker machine
- **Nmap** – Network scanning tool
- **Netcat (nc)** – Network utility
- **Curl** – HTTP request testing
- **Telnet / FTP / Nslookup** – Protocol testing tools

4. Snort Architecture Overview

Snort operates by capturing network packets and analyzing them using predefined rules. Each Snort rule consists of two main parts:

- **Rule Header**: Defines action, protocol, source IP/port, direction, and destination IP/port
- **Rule Options**: Defines the message, content to match, SID, revision number, and other metadata

Basic Rule Format:

action protocol src_ip src_port -> dest_ip dest_port (options)

5. Implementation and Rule Configuration

Rule 1: ICMP Ping Detection

Description: Detects ICMP echo requests (ping) which are often used for network reconnaissance.

Rule Structure: alert icmp any any -> 172.20.10.11 any (msg:"Ping is detected"; sid:1000001; rev:1;)

Attack Simulation: Continuous ping requests sent from attacker machine to target.

Outcome: Snort generated repeated alerts indicating ICMP traffic detection.

Rule 2: HTTP Traffic Detection

Description: Detects HTTP requests on port 80 to monitor unauthorized web access attempts.

Rule Structure: alert tcp any any -> 172.20.10.11 [80,443] (msg:"HTTP Traffic detected"; sid:1000002; rev:1;)

Attack Simulation: Curl command used to access HTTP service on the target machine.

Outcome: Snort successfully logged and alerted on HTTP traffic attempts.

Rule 3: DNS Query Detection

Description: Monitors DNS queries which can indicate reconnaissance or data exfiltration attempts.

Rule Structure: alert udp any any -> any 53 (msg:"DNS Query Detected"; sid:1000003; rev:1;)

Attack Simulation: Nslookup command executed against the target system.

Outcome: Multiple DNS query alerts were generated by Snort.

Rule 4: FTP Traffic Detection and Blocking

Description: Detects FTP traffic on port 21, which is insecure due to plaintext credentials.

Rule Structure: drop tcp any any -> 172.20.10.11 21 (msg:"FTP Traffic Blocked"; sid:1000006; rev:1;)

Attack Simulation: FTP connection attempt from attacker machine.

Outcome: Snort triggered alerts indicating FTP traffic detection.

Rule 5: Telnet Attempt Detection

Description: Detects Telnet access attempts on port 23, an insecure remote access protocol.

Rule Structure: alert tcp any any -> any 23 (msg:"TELNET Attempt Detected"; sid:1000005; rev:1;)

Attack Simulation: Telnet connection attempt to the target machine.

Outcome: Snort generated immediate alerts for Telnet attempts.

Rule 6: Unauthorized Port Access Detection (Port 9999)

Description: Detects traffic to a restricted or non-standard port to prevent backdoor access.

Rule Structure: block tcp any any -> 172.20.10.11 9999 (msg:"Blocking traffic to port 9999"; sid:1000008; rev:1;)

Attack Simulation: Netcat connection attempt to port 9999.

Outcome: Snort detected and alerted on unauthorized port access.

Rule 7: Cross-Site Scripting (XSS) Detection

Description: Detects XSS payloads embedded in HTTP requests.

Rule Structure: alert http any any -> $HOME_NET any (msg:"XSS Detected"; http_uri; content:"<script>"; sid:1000011; rev:11;)

Attack Simulation: Curl request containing <script>alert(1)</script> payload.

Outcome: Snort successfully detected the malicious script injection attempt.

Rule 8: SQL Injection Detection

Description: Identifies SQL Injection patterns in HTTP GET requests.

Rule Structure: alert http any any -> $HOME_NET any (msg:"SQLi Detected"; http_uri; content:"UNION"; nocase; content:"SELECT"; nocase; sid:1000010; rev:7;)

Attack Simulation: URL containing UNION SELECT SQL injection payload.

Outcome: Snort generated alerts indicating SQL Injection attempt.

Rule 9: Reverse Shell Attempt Detection

Description: Detects command execution attempts that may result in reverse shell access.

Rule Structure: alert tcp any any -> $HOME_NET any (msg:"Reverse Shell Attempt Detected"; http_raw_request; content:"/bin/bash"; sid:1000012; rev:3;)

Attack Simulation: Curl request containing /bin/bash command payload.

Outcome: Snort detected and alerted on the reverse shell attempt.

Rule 10: Nmap Stealth Scan Detection

Description: Detects SYN stealth scans commonly used during reconnaissance.

Rule Structure: alert tcp any any -> $HOME_NET any (msg:"Stealth Scan Detected"; flags:S; detection_filter:track by_src, count 10, seconds 5; sid:1000009; rev:1;)

Attack Simulation: nmap -sS scan launched against the target.

Outcome: Snort generated multiple alerts indicating stealth scan activity.

Rule 11: SSH Brute Force Detection

Description: Detects repeated SSH login attempts which may indicate a brute-force password attack on the target system.
Rule Structure: alert tcp any any -> any 22 (msg:"SSH Brute Force Attempt";

flow:to_server,established; detection_filter:track by_src, count 5, seconds 60; sid:1000004; rev:1;)

Attack Simulation: Multiple SSH login attempts executed using tools like Hydra or repeated manual password attempts from attacker machine.
Outcome: Snort generated alerts after multiple failed login attempts were detected within a short time window, indicating possible brute-force activity.

Rule 12: SMTP Traffic Rejection

Description: Detects and rejects SMTP traffic on port 25 to prevent unauthorized email relay or spam-based attacks.
Rule Structure: reject tcp any any -> 172.20.10.11 25 (msg:"SMTP Traffic Rejected"; sid:1000007; rev:1;)

Attack Simulation: Telnet or Netcat connection attempt made to port 25 to simulate unauthorized email service access.
Outcome: Snort rejected the SMTP connection attempt and logged the event, preventing potential misuse of the mail service.

6. Results and Analysis

All ten Snort rules were successfully triggered during testing. The IDS accurately detected various forms of malicious and suspicious traffic, demonstrating the effectiveness of rule-based detection. Proper rule tuning minimized false positives while maintaining high detection accuracy.

7. Limitations

- Signature-based detection may miss zero-day attacks
- Requires regular rule updates
- High traffic environments may generate large volumes of alerts

8. Future Enhancements

- Integration with SIEM tools
- Implementation of Snort in IPS mode
- Advanced rule tuning and thresholding
- Automation of alert analysis

9. Conclusion

This project successfully demonstrates the practical implementation of Snort as a rule-based Intrusion Detection System. By configuring and testing ten different Snort rules, the project highlights how IDS solutions can enhance network visibility and proactively detect threats. Snort proves to be a flexible, powerful, and cost-effective solution for network security monitoring.