Network Intrusion Detection System using Suricata
DVWA Security Testing Lab Report

Author: Prajwal V

Environment: Kali Linux | Suricata IDS | DVWA | Apache | MySQL

1. Introduction

This project implements a Host-Based Intrusion Detection System (IDS) using Suricata to monitor, detect, and analyze malicious network activities. The system is tested against a deliberately vulnerable web application (DVWA) to simulate real-world cyber attacks and validate detection capabilities.

2. Objectives

- Understand IDS/IPS concepts
- Deploy Suricata for live packet inspection
- Write custom detection rules
- Simulate attacks safely in a lab
- Analyze logs and alerts
- Demonstrate threat detection skills

3. Background Study

An Intrusion Detection System (IDS) monitors network traffic to identify suspicious activities. Signature-based IDS uses predefined rules to detect known attack patterns. Suricata is an open-source IDS/IPS engine capable of deep packet inspection, protocol analysis, and real-time alerting.

4. Tools and Technologies Used

Kali Linux
Suricata IDS
DVWA (Damn Vulnerable Web Application)
Apache Web Server
MySQL Database
Nmap
Browser-based testing

5. System Architecture

Attacker → DVWA Web Server → Suricata IDS → Log Files (fast.log / eve.json)

Suricata captures packets from the network interface and compares them against custom rules to generate alerts.

6. Environment Setup

1. Installed Suricata
2. Configured rule path in suricata.yaml
3. Created custom local.rules
4. Started Apache and MySQL
5. Deployed DVWA
6. Started Suricata in monitoring mode

7. Custom Rule Design

Rules were written to detect:
- ICMP Ping
- SYN packets
- Port scanning
- Brute force login attempts
- SQL injection
- Command injection

Each rule uses content matching, TCP flags, or thresholds.

8. Attack Simulation and Testing

8.1 Port Scanning

Nmap SYN scans were executed to probe open ports. Suricata generated alerts indicating potential scanning behavior.

8.2 Brute Force Login

Repeated login attempts were made on DVWA login page. Multiple HTTP requests triggered brute force detection rules.

8.3 SQL Injection

Payloads such as ' OR '1'='1 were injected into input fields. Suricata detected malicious SQL patterns and raised alerts.

8.4 Command Injection

Special characters such as ';' were injected to execute commands. Suricata detected suspicious content and flagged the attack.

9. Results

Port Scan – Detected
SYN Scan – Detected
Brute Force – Detected
SQL Injection – Detected
Command Injection – Detected

All attacks were successfully identified by custom Suricata rules.

10. Log Analysis

Alerts were monitored using:
/var/log/suricata/fast.log
/var/log/suricata/eve.json

These logs provided timestamps, source IP, destination IP, protocol, and rule ID.

11. Challenges Faced

- Kernel interface issues with af-packet
- Duplicate rule IDs
- Rule syntax errors
- Monitoring localhost traffic

These were resolved by correcting rule paths, fixing signatures, and restarting Suricata.

12. Skills Acquired

Network traffic analysis
Linux administration
IDS rule writing
Cyber attack simulation
Security monitoring
Log investigation

13. Conclusion

The project successfully demonstrates the deployment of an Intrusion Detection System using Suricata. Custom signatures effectively detected multiple real-world attack scenarios, providing practical knowledge in defensive cybersecurity.

14. Future Enhancements

- Integrate ELK/Splunk dashboards
- Implement IPS blocking
- Add anomaly-based detection
- Deploy in enterprise networks