# Intrusion Detection System

Name: Prateek(166), Dev Sharma(179)

## 1. Introduction

This report presents a lightweight Intrusion Detection System (IDS) built in Python with a PyQt5 GUI. The IDS is designed as a learning and research tool, allowing SOC analysts and security researchers to monitor live traffic or analyze PCAP files. The system emphasizes detection, alerting, and analysis without attempting to block traffic (unlike an IPS).

## 2. Features

The IDS integrates several core features:

- Real-time packet sniffing and rule-based detection
- PyQt5 GUI dashboard for live monitoring
- Visualizations: line, pie, and bar charts for activity insights
- Persistent logging (JSON and CSV)

## 3. Detection Logic

The IDS employs rule-based detection techniques, including:

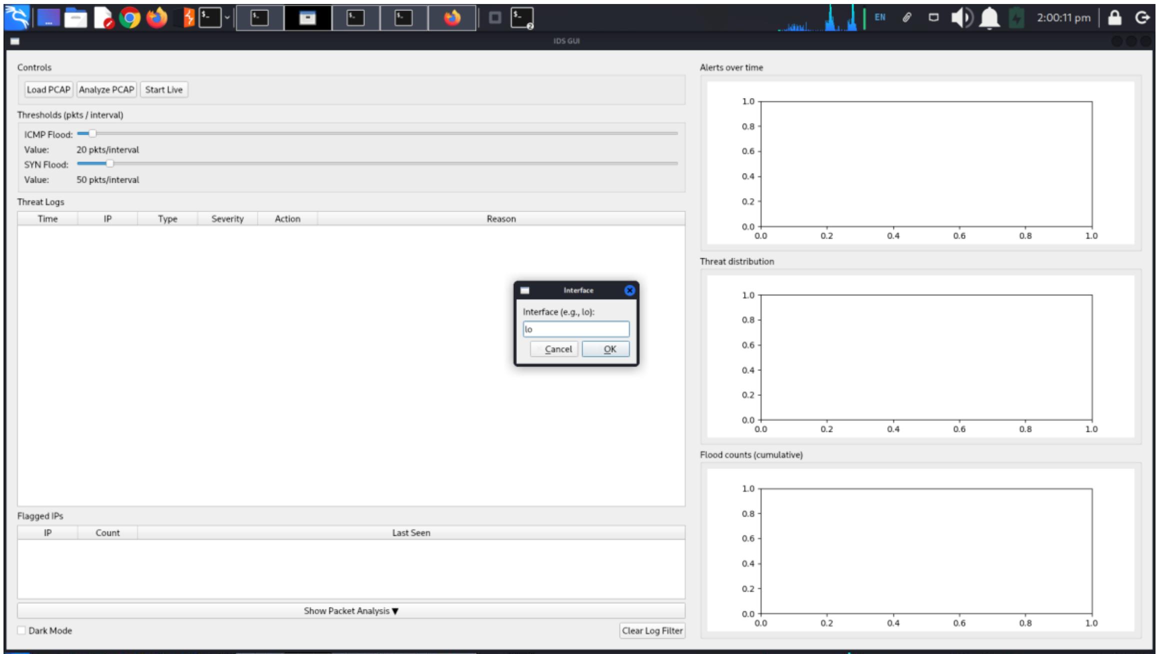
- ICMP Floods – alerts when ICMP echo requests exceed a threshold
- TCP SYN Floods – detects repeated SYNs without completing handshakes
- Port Scans – flags when multiple ports are probed in short succession
- Suspicious Payloads – matches known patterns such as SQL injection (`' OR 1=1`), XSS (`<script>`), and LFI attempts (`../../etc/passwd`)

## 4. Demo

Two PCAPs were generated and analyzed to validate detection:

- normal.pcap – benign traffic (simple TCP handshakes and HTTP GET requests)
- malicious.pcap – simulated ICMP flood, SYN scans, null/FIN/Xmas scans, and malicious HTTP payloads

Screenshots below illustrate detection results in the GUI:



Thresholds (pkts / interval)

ICMP Flood:
Value: 20 pkts/interval
SYN Flood:
Value: 50 pkts/interval

Threat Logs

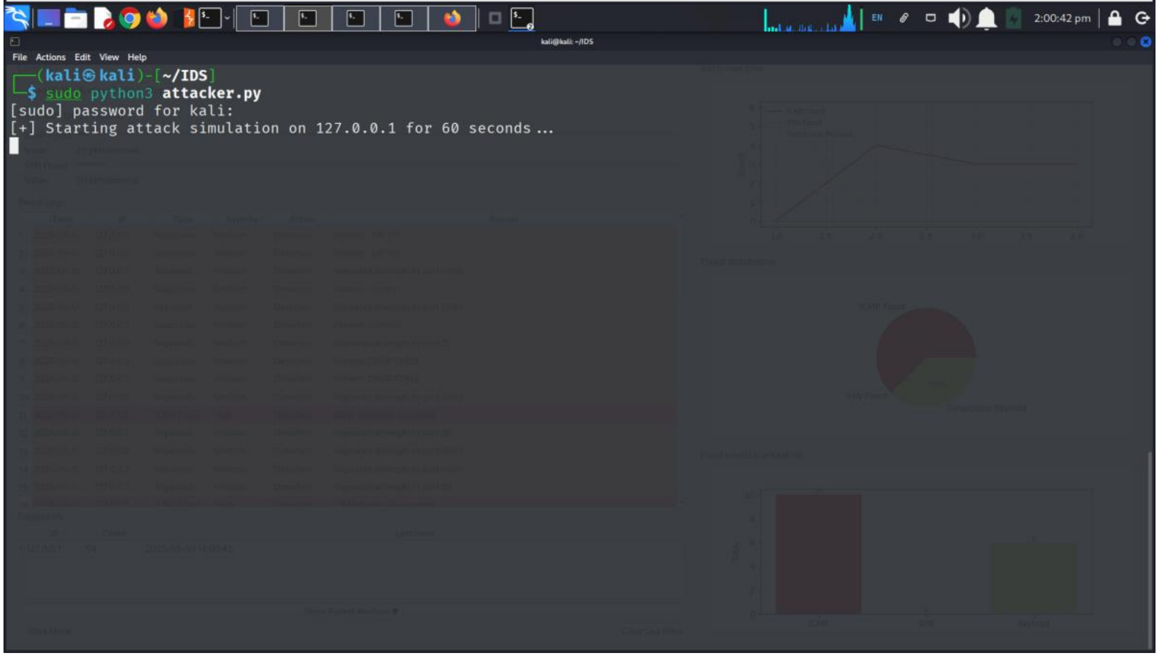
| Time | IP | Type | Severity | Action | Reason |
|---|---|---|---|---|---|

Interface (e.g., lo): lo

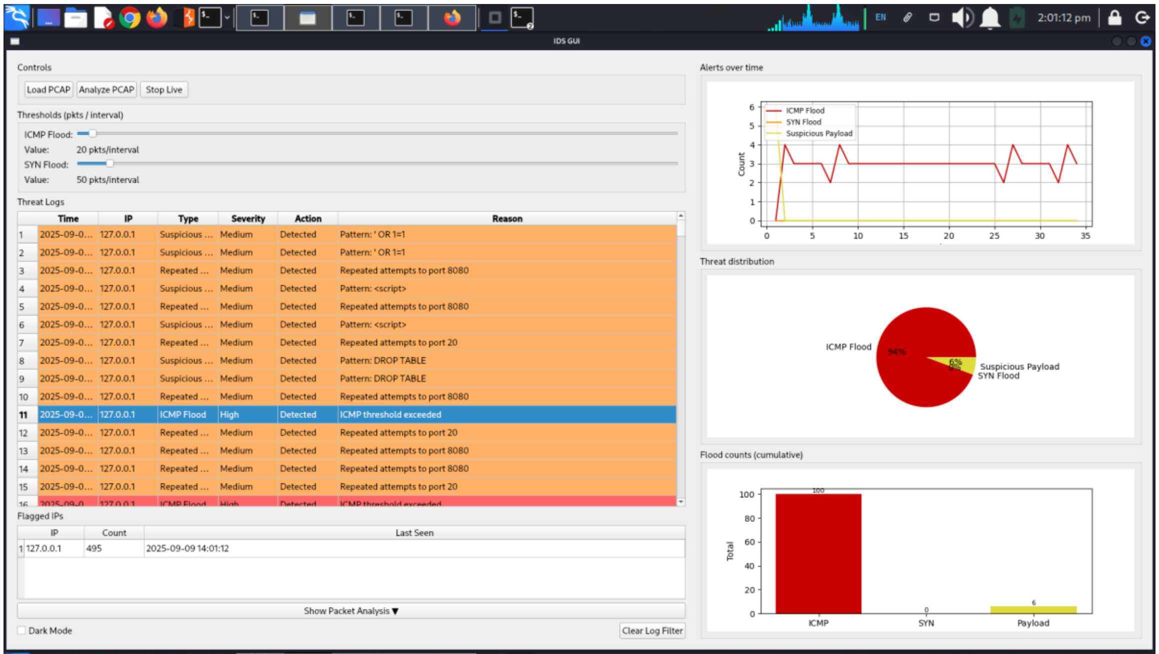
Flagged IPs

| IP | Count | Last Seen |
|---|---|---|

□ Dark Mode



```
┌──(kali㉿kali)-[~/IDS]
└─$ sudo python3 attacker.py
[sudo] password for kali:
[+] Starting attack simulation on 127.0.0.1 for 60 seconds ...
```



Thresholds (pkts / interval)

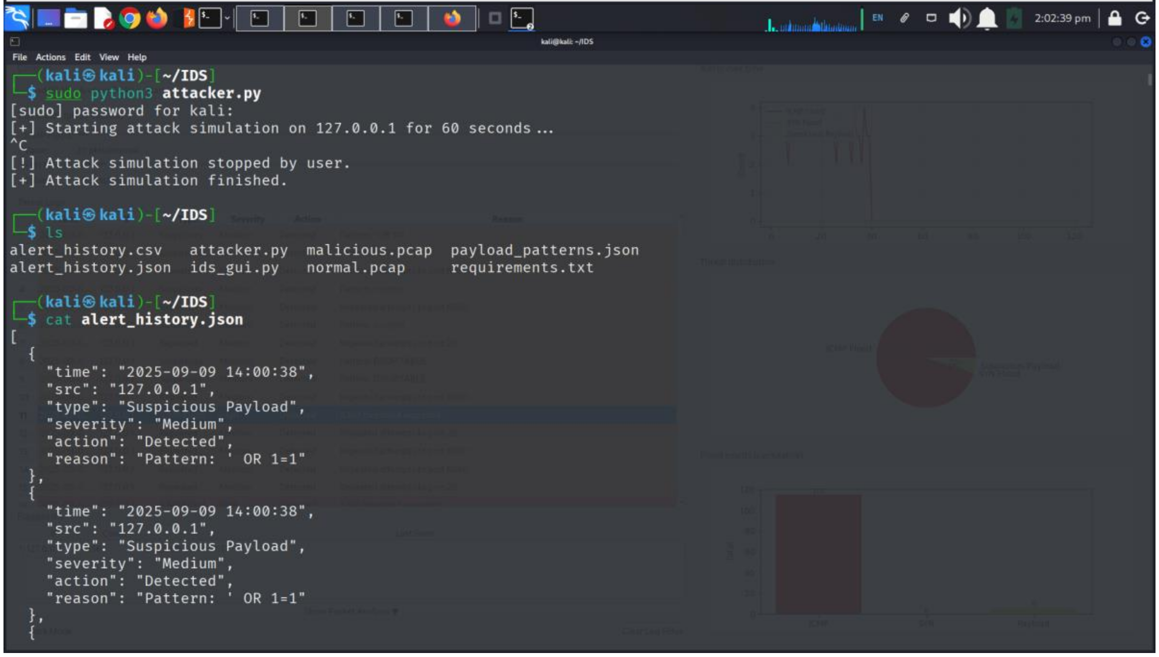
ICMP Flood:
Value: 20 pkts/interval
SYN Flood:
Value: 50 pkts/interval

Threat Logs

| | Time | IP | Type | Severity | Action | Reason |
|---|---|---|---|---|---|---|
| 1 | 2025-09-0... | 127.0.0.1 | Suspicious ... | Medium | Detected | Pattern: ' OR 1=1 |
| 2 | 2025-09-0... | 127.0.0.1 | Suspicious ... | Medium | Detected | Pattern: ' OR 1=1 |
| 3 | 2025-09-0... | 127.0.0.1 | Repeated ... | Medium | Detected | Repeated attempts to port 8080 |
| 4 | 2025-09-0... | 127.0.0.1 | Suspicious ... | Medium | Detected | Pattern: <script> |
| 5 | 2025-09-0... | 127.0.0.1 | Repeated ... | Medium | Detected | Repeated attempts to port 8080 |
| 6 | 2025-09-0... | 127.0.0.1 | Suspicious ... | Medium | Detected | Pattern: <script> |
| 7 | 2025-09-0... | 127.0.0.1 | Repeated ... | Medium | Detected | Repeated attempts to port 20 |
| 8 | 2025-09-0... | 127.0.0.1 | Suspicious ... | Medium | Detected | Pattern: DROP TABLE |
| 9 | 2025-09-0... | 127.0.0.1 | Suspicious ... | Medium | Detected | Pattern: DROP TABLE |
| 10 | 2025-09-0... | 127.0.0.1 | Repeated ... | Medium | Detected | Repeated attempts to port 8080 |
| 11 | 2025-09-0... | 127.0.0.1 | ICMP Flood | High | Detected | ICMP threshold exceeded |
| 12 | 2025-09-0... | 127.0.0.1 | Repeated ... | Medium | Detected | Repeated attempts to port 20 |
| 13 | 2025-09-0... | 127.0.0.1 | Repeated ... | Medium | Detected | Repeated attempts to port 8080 |
| 14 | 2025-09-0... | 127.0.0.1 | Repeated ... | Medium | Detected | Repeated attempts to port 8080 |
| 15 | 2025-09-0... | 127.0.0.1 | Repeated ... | Medium | Detected | Repeated attempts to port 20 |
| 16 | 2025-09-0 | 127.0.0.1 | ICMP Flood | High | Detected | ICMP threshold exceeded |

Flagged IPs

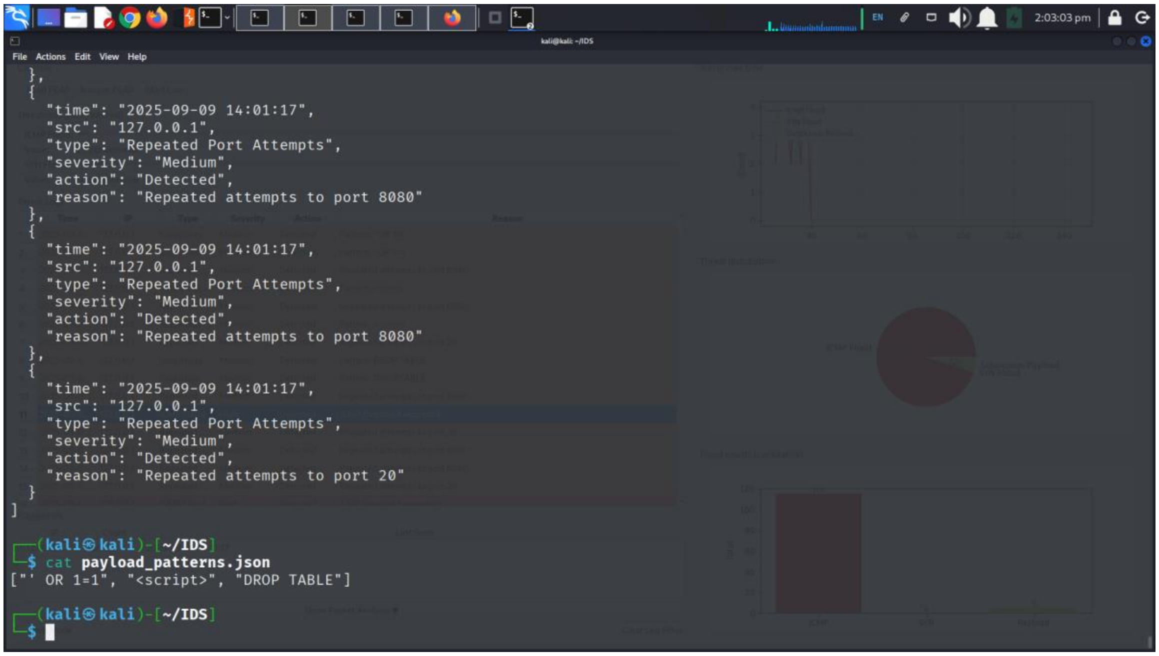
| | IP | Count | Last Seen |
|---|---|---|---|
| 1 | 127.0.0.1 | 495 | 2025-09-09 14:01:12 |



```
┌──(kali㉿kali)-[~/IDS]
└─$ sudo python3 attacker.py
[sudo] password for kali:
[+] Starting attack simulation on 127.0.0.1 for 60 seconds ...
^C
[!] Attack simulation stopped by user.
[+] Attack simulation finished.

┌──(kali㉿kali)-[~/IDS]
└─$ ls
alert_history.csv   attacker.py  malicious.pcap  payload_patterns.json
alert_history.json  ids_gui.py   normal.pcap     requirements.txt

┌──(kali㉿kali)-[~/IDS]
└─$ cat alert_history.json
[
  {
    "time": "2025-09-09 14:00:38",
    "src": "127.0.0.1",
    "type": "Suspicious Payload",
    "severity": "Medium",
    "action": "Detected",
    "reason": "Pattern: ' OR 1=1"
  },
  {
    "time": "2025-09-09 14:00:38",
    "src": "127.0.0.1",
    "type": "Suspicious Payload",
    "severity": "Medium",
    "action": "Detected",
    "reason": "Pattern: ' OR 1=1"
  },
  {
```



```
  },
  {
    "time": "2025-09-09 14:01:17",
    "src": "127.0.0.1",
    "type": "Repeated Port Attempts",
    "severity": "Medium",
    "action": "Detected",
    "reason": "Repeated attempts to port 8080"
  },
  {
    "time": "2025-09-09 14:01:17",
    "src": "127.0.0.1",
    "type": "Repeated Port Attempts",
    "severity": "Medium",
    "action": "Detected",
    "reason": "Repeated attempts to port 8080"
  },
  {
    "time": "2025-09-09 14:01:17",
    "src": "127.0.0.1",
    "type": "Repeated Port Attempts",
    "severity": "Medium",
    "action": "Detected",
    "reason": "Repeated attempts to port 20"
  }
]

┌──(kali㉿kali)-[~/IDS]
└─$ cat payload_patterns.json
["' OR 1=1", "<script>", "DROP TABLE"]

┌──(kali㉿kali)-[~/IDS]
└─$ 
```

## 5. False Positive Considerations

Rule-based detection can raise false positives under legitimate high-traffic conditions. For example, large ICMP test pings or legitimate port scanning by vulnerability scanners could trigger alerts. Thresholds must be tuned according to the environment. Payload detection may also over-flag benign inputs resembling attack patterns.

## 6. Next Steps / Improvements

Recommended improvements include:

- • Regex-based payload detection for more flexibility
- • GeoIP lookup for flagged IPs
- • Filtering alerts by severity
- • Richer packet analysis (headers + payload view)
- • Integration with SIEM platforms

## 7. Testing

Testing was conducted using:

- • PCAP-based replay (normal and malicious captures)
- • Simulated attacker script (lo_attacker.py) generating ICMP floods, SYN floods, and payload injections

Both methods successfully triggered alerts in the GUI.

## 8. Unit and Integration Tests

Basic unit tests should validate detection logic functions, e.g.:

- • Threshold check functions for ICMP/SYN floods
- • Port scan detection logic using crafted packet sequences
- • Payload pattern matcher against known malicious strings

Integration tests can replay PCAPs to validate the entire workflow, ensuring alerts are logged and displayed in the GUI.

## 9. Conclusion

The IDS provides a functional prototype for detecting common attacks such as floods, scans, and malicious payloads. While not production-grade, it demonstrates the core building blocks of detection systems and provides a foundation for further experimentation and enhancement.