**Lightweight NIDS Technical Report**

**Interns Name**: Dev Sharma (179), Prateek (166)  
**Date**: 17 Aug, 2025

**1. Detection Logic**

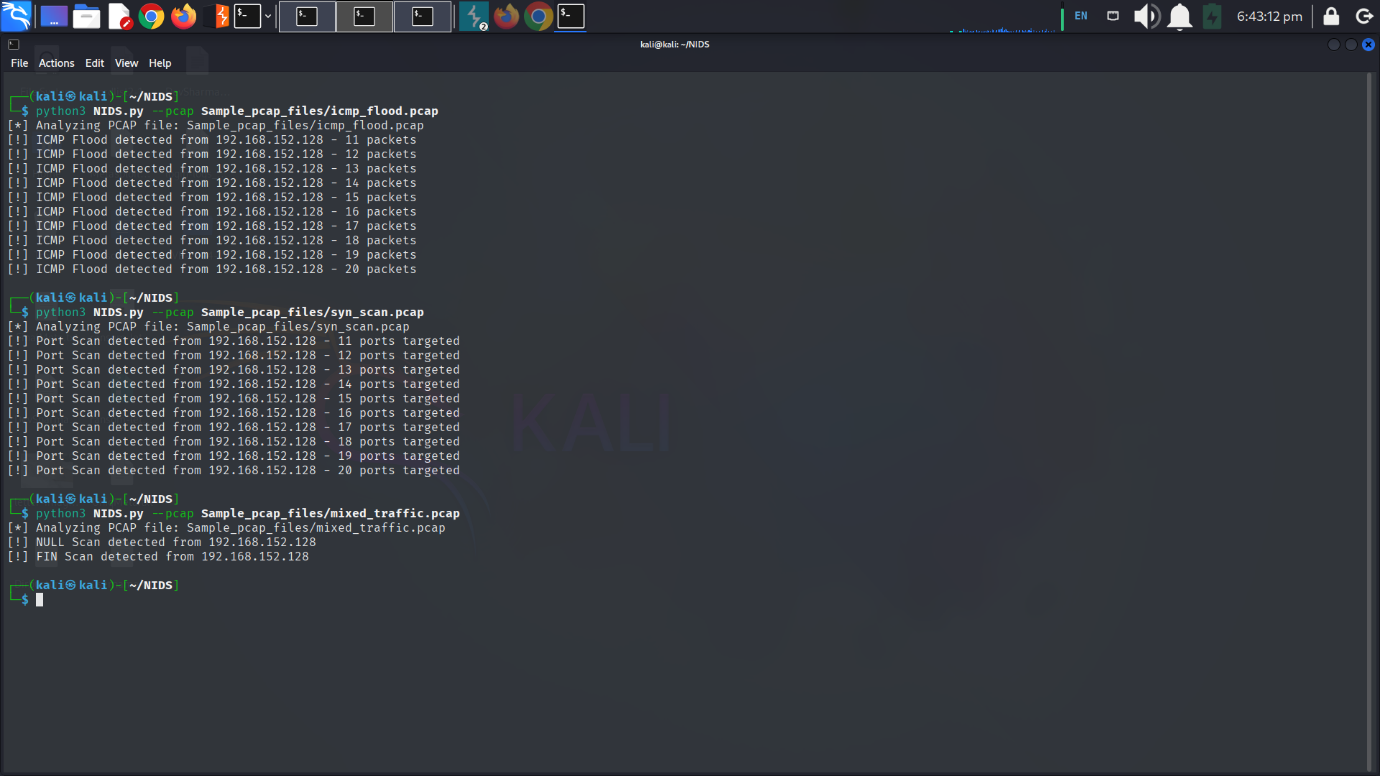
The NIDS uses a combination of **threshold-based** and **pattern-based** detection methods to identify suspicious network activity.

**Key Detection Methods**

| **Attack Type** | **Detection Logic** |
| --- | --- |
| **ICMP Flood** | Counts ICMP echo requests (Type 8) from a single source; triggers if threshold exceeded (default: 10 packets/5 sec). |
| **TCP SYN Scan** | Tracks SYN packets to multiple ports from one IP; flags if > threshold unique ports are targeted. |
| **Half-Open Connections** | Monitors SYN without ACK responses; alerts on excessive incomplete handshakes. |
| **NULL/FIN/XMAS Scans** | Checks TCP flags (e.g., NULL = no flags, FIN = FIN-only) to detect stealth scans. |
| **High-Rate SYNs** | Measures SYNs/second; triggers if exceeding a dynamic baseline (e.g., >50 SYNs/sec). |

**Technical Implementation**:

* Uses Scapy for packet dissection.
* State tracking via Python defaultdict for efficient counting.
* Time-windowed analysis to reduce memory usage.



**2. False-Positive Considerations**

**Common False Positives**

1. **Legitimate ICMP Traffic**:
   * Network monitoring tools (e.g., PingPlotter) may trigger ICMP flood alerts.  
     *Mitigation*: Whitelist trusted IPs or increase the threshold.
2. **Load Balancers/Cloud Services**:
   * Services like AWS ELB generate SYN traffic resembling scans.  
     *Mitigation*: Exclude known cloud IP ranges.
3. **VPNs/Proxies**:
   * Multiple users behind a single IP may appear as a scan.  
     *Mitigation*: Use rate-limiting instead of per-IP thresholds.
4. **Network Scanners (Nmap)**:
   * Authorized scans (e.g., IT audits) trigger alerts.  
     *Mitigation*: Add a "learning mode" to baseline normal traffic.

**Reducing False Positives**

* **Context-Aware Rules**: Combine multiple indicators (e.g., ICMP + SYN scans) to confirm attacks.
* **Whitelisting**: Support for config.yml to exclude trusted IPs/subnets.
* **Adaptive Thresholds**: Dynamically adjust thresholds based on historical traffic.

**3. Next-Step Improvements**

**Short-Term (1–3 Months)**

* **GUI Dashboard**: Web interface (Flask) for real-time alert visualization.
* **Enhanced Logging**: JSON output for integration with SIEMs (e.g., Splunk, ELK).
* **GeoIP Tagging**: Annotate alerts with source country (MaxMind DB).

**Medium-Term (3–6 Months)**

* **Machine Learning**: Anomaly detection using scikit-learn to baseline normal traffic.
* **Protocol Decoders**: Deep inspection of HTTP/DNS for C2 traffic patterns.
* **Containerization**: Docker support for easy deployment.

**Long-Term (6+ Months)**

* **Distributed Mode**: Agent-based deployment for multi-node monitoring.
* **Threat Intelligence Feeds**: Integrate abuse.ch or AlienVault OTX for known-bad IPs.
* **Automated Response**: Optional iptables blocking for critical alerts.

**Conclusion**

This lightweight NIDS balances simplicity with effective detection for common network threats. Future work will focus on reducing false positives through adaptive learning and expanding detection capabilities via protocol analysis. The modular design allows for incremental improvements while maintaining low resource usage.

**Recommendation**: Pilot-test in a lab environment to fine-tune thresholds before production deployment.

**Appendix**: Sample alert logic pseudocode available in /docs/alert\_rules.md.