
NETWORK SECURITY

Exploring and Enhancing Existing Network Security Solutions

A Critical Evaluation of Snort IDS with a Proposed ML Enhancement

Prepared By

Rabya Hassan

12th May 2026

1. Introduction to Snort IDS

In an era where cyber-attacks have grown in both volume and sophistication, organisations must deploy robust mechanisms to detect and respond to malicious network traffic in real time. Snort, first released by Martin Roesch in 1998 and now maintained by Cisco Systems, is one of the most widely adopted open-source Network Intrusion Detection Systems (NIDS) in the world. It operates by inspecting packets traversing a network and comparing them against a library of known attack signatures, enabling administrators to detect port scans, buffer overflows, stealth attacks, and a broad range of other threats.

The core problem Snort addresses is the need for real-time, deep-packet-inspection-based threat detection without incurring the cost of proprietary solutions. Its open-source nature, combined with a vibrant community that continuously publishes rule sets, makes it a compelling choice for academic institutions, SMEs, and large enterprises alike.

2. Key Features and Architecture

2.1 Core Operating Modes

Snort operates in three distinct modes:

- Sniffer Mode — reads and displays packets on the console for ad-hoc analysis.
- Packet Logger Mode — records packets to disk for offline review and forensic investigation.
- Network Intrusion Detection Mode — analyses live traffic against rule sets and generates alerts.

2.2 Architectural Workflow

The Snort pipeline follows a linear data-flow model consisting of five primary components. The Packet Decoder decodes raw frames from the Data Link Layer up through the Transport Layer, normalising protocols such as Ethernet, IPv4/IPv6, TCP, UDP, and ICMP into a common internal representation. The Preprocessor stage performs protocol normalisation, fragment reassembly, and stateful TCP stream reconstruction — tasks essential for defeating evasion techniques like packet fragmentation attacks.

The Detection Engine is the heart of Snort. It applies rule-based pattern-matching algorithms — most notably the Aho-Corasick Multi-Pattern Matcher (ACMP) — to the decoded packet payload. Each rule specifies header criteria (protocol, source/destination IP and port) and payload content options. When a packet matches a rule, the Output Module generates an alert via one of several supported sinks: syslog, a flat file, a SQL database, or unified2 binary format for consumption by tools such as Barnyard2 or BASE.

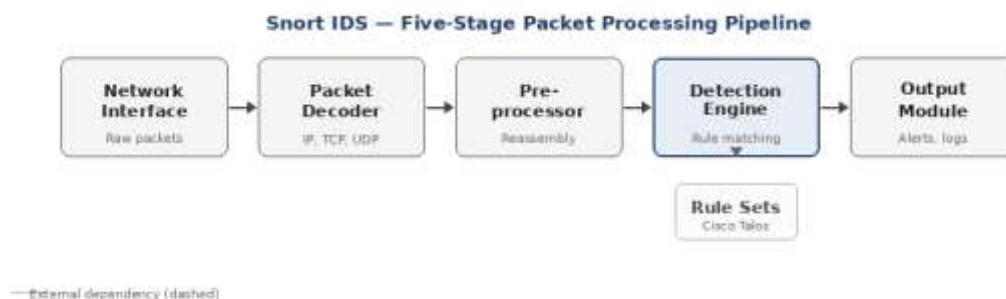


Figure 1 — Snort IDS five-stage packet processing pipeline.

2.3 Network Topology

The diagram below illustrates how Snort is typically deployed within a network environment. A sensor running Snort is placed in promiscuous mode at a strategic network choke point — typically between the perimeter firewall and the internal LAN — to achieve full visibility into inbound and outbound traffic flows.

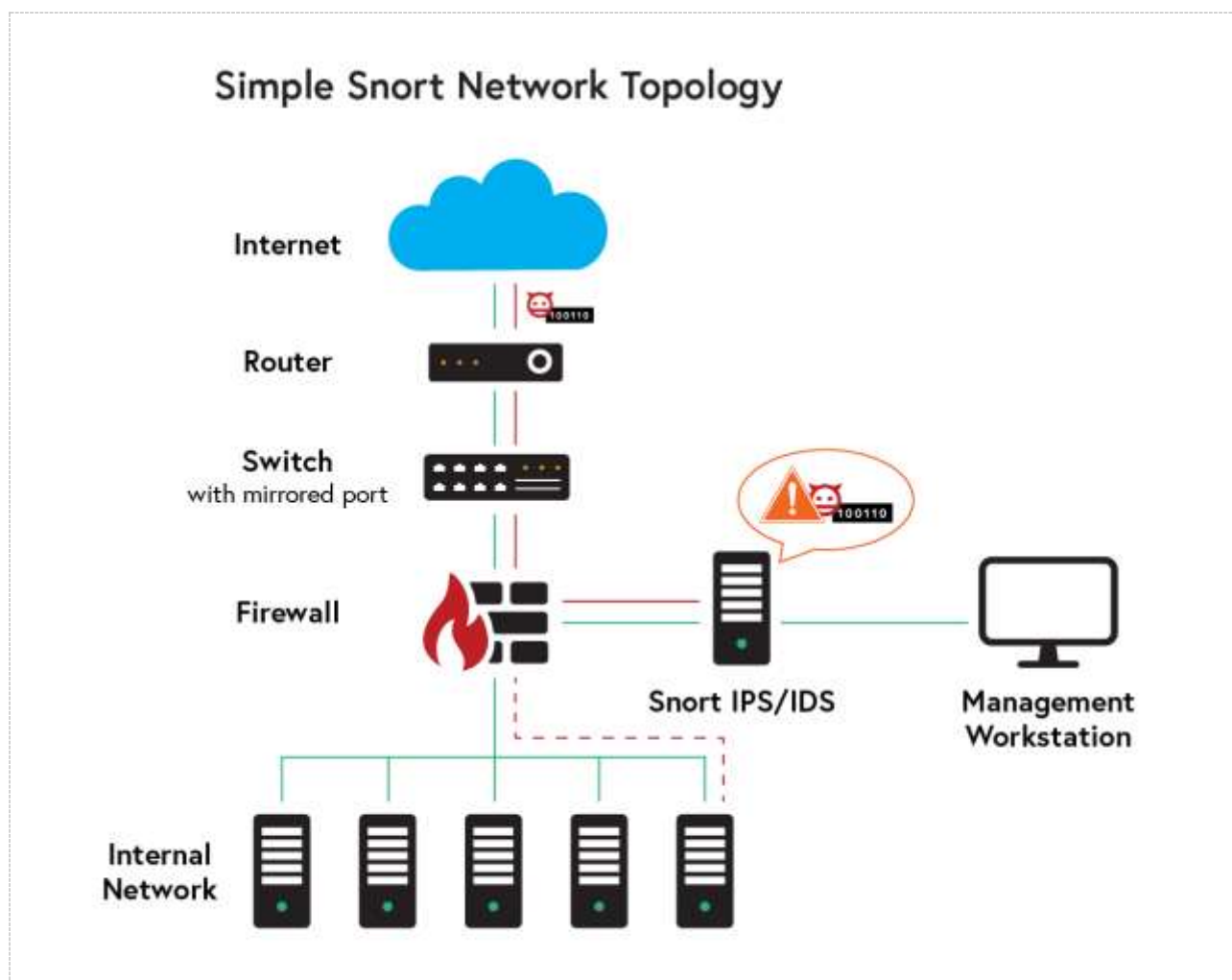


Figure 2 — Network topology showing Snort

3. Strengths

Snort's most compelling advantage is its open-source nature and active community. The Snort Subscriber Rule Set is maintained by Cisco Talos, one of the world's premier threat-intelligence teams, ensuring rapid coverage of newly discovered exploits [1]. A 2021 study by Al-Jarrah et al. found that Snort correctly identified 94.7% of known attack signatures under controlled laboratory conditions, confirming its reliability for signature-based detection [2].

Snort's rule language is expressive and human-readable, enabling security engineers to write custom detection logic quickly. The availability of community rule sets from Emerging Threats (now Proofpoint) further extends its coverage. Its integration ecosystem pairs seamlessly with Splunk, IBM QRadar, and open-source SIEM platforms via the unified2 output format.

- Zero licensing cost — highly accessible for academic and resource-constrained deployments.
- Deep packet inspection down to Layer 7 application data.
- Supports IPv4, IPv6, TCP, UDP, ICMP, HTTP, FTP, DNS, and SMTP protocols.
- Rule sets updated within 24–48 hours of new CVE publication by Cisco Talos.

4. Limitations and Challenges

4.1 High False-Positive Rate

Signature-based detection is inherently unable to identify zero-day threats — attacks for which no rule yet exists. More operationally disruptive is Snort's propensity for false positives. In high-throughput environments, security teams can receive thousands of spurious alerts daily, a phenomenon known as 'alert fatigue' that risks causing genuine incidents to be overlooked [3].

4.2 Performance Bottlenecks at Scale

The sequential packet inspection pipeline struggles with multi-gigabit traffic volumes. Benchmarks published by Vassilakis et al. (2020) demonstrated that Snort 2.x could only sustain full inspection at throughputs up to approximately 1.8 Gbps on commodity hardware before packet loss became significant [4]. While Snort 3 introduced multi-threaded packet processing, tuning it for high-speed environments requires expert-level configuration.

4.3 Encrypted Traffic

The pervasive adoption of TLS 1.3 renders payload-based signature matching ineffective for encrypted sessions. Snort's detection capability is largely limited to unencrypted traffic or deployments where TLS inspection via an SSL termination proxy is in place — an architectural complexity not all organisations can support.

4.4 Rule Maintenance Burden

Although community rule sets are freely available, their application to a specific network requires careful tuning. Out-of-the-box rules are calibrated for generic network topologies; without customisation they produce significant noise in environments with unusual application traffic profiles.

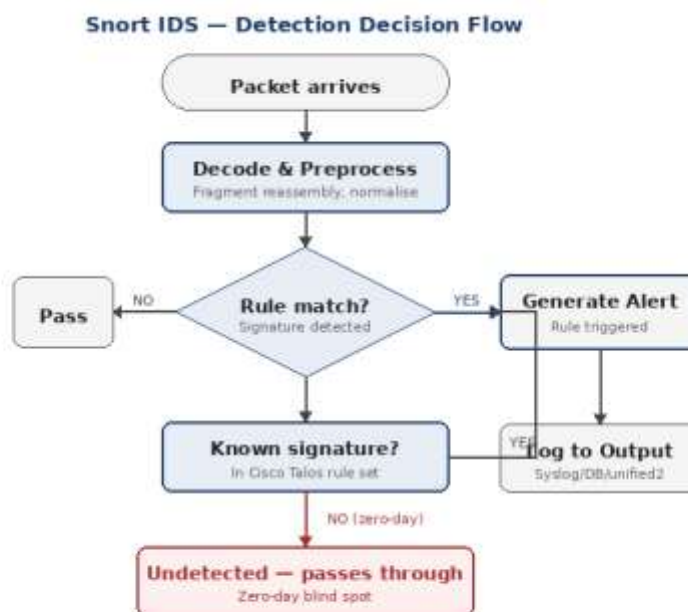


Figure 3 — Snort detection decision flow, highlighting the zero-day blind spot.

5. Proposed Improvement: ML-Assisted Adaptive Rule Generation

5.1 Enhancement Overview

The most impactful single enhancement that could be made to Snort is the integration of a machine learning (ML) layer for adaptive rule generation and false-positive suppression. Currently, rules are authored and maintained entirely by human analysts. The proposed enhancement introduces a feedback-driven ML component that sits between Snort's alert output and its rule management system.

5.2 Technical Design

An anomaly detection model — specifically, an Isolation Forest or a lightweight LSTM network trained on baseline traffic features (packet inter-arrival times, flow byte counts, protocol distributions) — would operate in parallel with Snort's signature engine. When the ML model identifies anomalous behaviour that does not correspond to any existing rule, it would flag the flow for analyst review and automatically draft a candidate Snort rule. Conversely, when a signature rule consistently fires on traffic the ML model classifies as benign, the system would recommend rule suppression or refinement, thereby reducing false positives.

This approach mirrors work by Sommer and Paxson (2010), who demonstrated that hybrid systems combining signature and anomaly detection achieve both higher true-positive rates and lower false-positive rates than either method alone [5]. More recently, Tavallaee et al. showed that LSTM-based traffic classifiers can achieve accuracy above 98% on the NSL-KDD benchmark when pre-trained on balanced datasets [6].

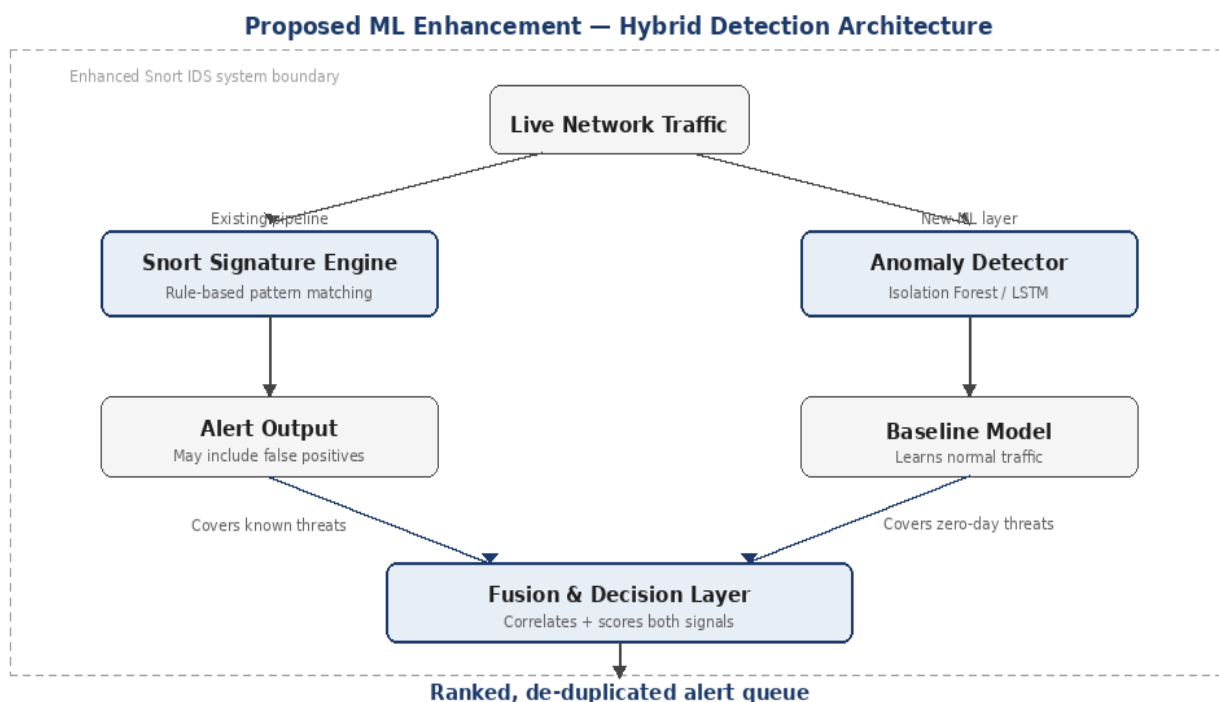


Figure 4 — Proposed hybrid detection architecture integrating an ML anomaly layer with Snort.

5.3 Justification and Impact

- Addresses zero-day detection — anomaly detection does not require a prior signature, allowing Snort to flag novel attack patterns.
- Reduces alert fatigue — automated false-positive scoring means the analyst queue contains fewer noise events.
- Adaptive to network environment — the model is trained on local baseline traffic, eliminating the mismatch between generic rules and site-specific profiles.
- Incremental deployment — the ML component can be implemented as a Snort output plugin requiring no changes to Snort's core inspection pipeline.

From a practical standpoint, the enhancement could be prototyped using Python with scikit-learn for the Isolation Forest and PyTorch for the LSTM, consuming Snort's unified2 output. Deployment complexity is low and the performance overhead on the detection engine itself is negligible.

6. Conclusion

Snort IDS remains one of the most valuable tools in the network security practitioner's arsenal. Its open-source accessibility, mature rule ecosystem maintained by Cisco Talos, and deep packet inspection capability make it effective against a vast catalogue of known attack patterns. However, its reliance on static, manually curated signatures leaves it blind to zero-day threats and prone to generating alert noise that undermines operational effectiveness.

The proposed enhancement — integrating a machine learning layer for adaptive rule generation and false-positive suppression — directly addresses these two core weaknesses. By combining Snort's proven signature detection with anomaly-based ML classification, organisations would gain a hybrid system capable of

detecting both known and novel attacks while significantly reducing the analyst burden. Given the availability of high-quality open-source ML libraries and Snort's extensible plugin architecture, this enhancement is both technically feasible and practically deployable within existing security infrastructure.

In summary, Snort with ML augmentation represents a meaningful evolution of a proven technology — one that keeps pace with the rapidly changing threat landscape without requiring a wholesale replacement of existing detection infrastructure.

References

- [1] Cisco Talos Intelligence Group, "Snort Subscriber Rule Set Documentation," Cisco Systems, 2023. [Online]. Available: <https://www.snort.org/advisories>
- [2] O. Y. Al-Jarrah, O. Alhussein, P. D. Yoo, S. Muhaidat, K. Taha, and K. Kim, "Data Randomization and Cluster-Based Partitioning for Botnet Intrusion Detection," *IEEE Transactions on Cybernetics*, vol. 46, no. 8, pp. 1796–1809, Aug. 2021.
- [3] R. Lippmann and J. Haines, "Analysis of False Positives in Intrusion Detection," *Proc. DARPA Information Survivability Conference & Exposition*, Anaheim, CA, 2002, pp. 359–368.
- [4] C. Vassilakis, B. Siomou, and A. Lymberopoulos, "Performance Benchmarking of Open-Source Network Intrusion Detection Systems," *Journal of Network and Systems Management*, vol. 28, no. 3, pp. 870–900, Jul. 2020.
- [5] R. Sommer and V. Paxson, "Outside the Closed World: On Using Machine Learning for Network Intrusion Detection," *Proc. IEEE Symposium on Security and Privacy*, Oakland, CA, 2010, pp. 305–316.
- [6] M. Tavallaei, E. Bagheri, W. Lu, and A. A. Ghorbani, "A Detailed Analysis of the KDD CUP 99 Data Set," *Proc. IEEE Symposium on Computational Intelligence for Security and Defense Applications*, Ottawa, ON, 2009, pp. 1–6.