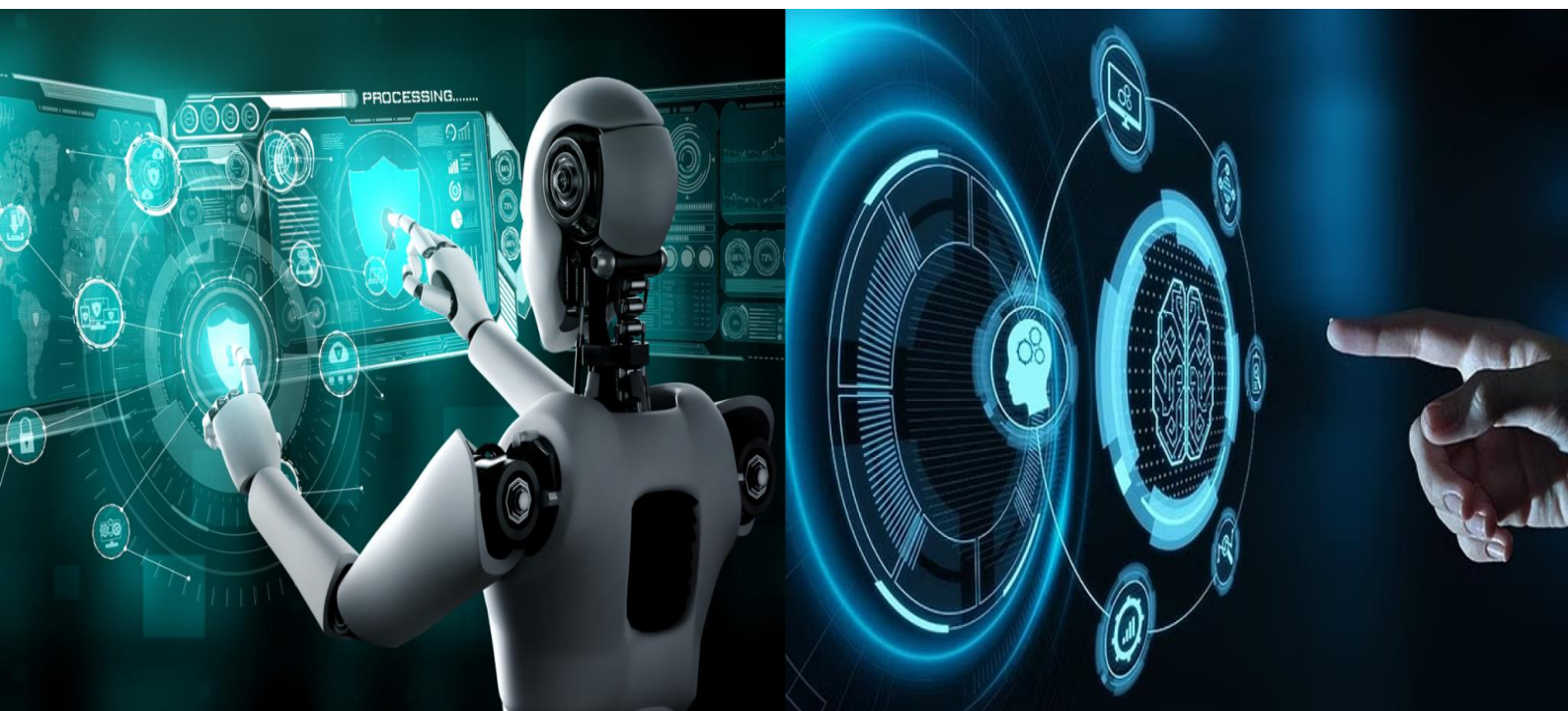


International Journal of Innovative Research in Computer and Communication Engineering

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)





Real-Time Network Traffic Monitoring and Anomaly Detection Dashboard

D.S. Deepika¹, Kanishik E², Bharathwaj K S³, Harish B⁴

Assistant Professor, Department of Information Technology, R.M.D Engineering College, Thiruvallur, India¹

U.G. Student, Department of Information Technology, R.M.D Engineering College, Thiruvallur, India^{2,3,4}

ABSTRACT: Monitoring network activity in real-time has become a necessity for any organisation that takes cybersecurity seriously. This paper describes a real-time network traffic monitoring and anomaly detection dashboard built using a Node.js backend, a React-based frontend, and Supabase as the persistence and authentication layer. The system captures raw packets through node-ncap, extracting fields such as source and destination IP, transport protocol, payload size, and timestamp, then stores them in a PostgreSQL database. A rule-based engine running server-side inspects each packet against configurable thresholds, flagging high packet rates, uncommon port combinations, and DDoS-like patterns. Alerts are pushed to connected browsers in under a second via Socket.io. The frontend presents analysts with a live traffic chart, protocol breakdown, top-talker rankings, a rolling alert feed, and at-a-glance counters. Evaluation on a simulated campus network showed the system correctly flagged synthetic attack traffic while keeping false positives manageable, demonstrating that a JavaScript-first stack can deliver meaningful security visibility without dedicated SIEM infrastructure.

KEYWORDS: Network Traffic Monitoring, Anomaly Detection, Node.js, Supabase, Socket.io, React, DDoS Detection, Cybersecurity Dashboard, Real-Time Analytics.

I. INTRODUCTION

Network security has never been more urgent. Hardly a month passes without news of a data breach, a ransomware incident, or a prolonged denial-of-service attack knocking critical services offline. Many smaller organisations still rely on reactive measures, learning about attacks only after damage is done, because they lack tools to watch their own traffic in real time.

Commercial solutions such as Splunk, Darktrace, and Cisco's network visibility suite are excellent but carry price tags and maintenance burdens that put them out of reach for resource-constrained teams. There is genuine value in exploring whether a modern open-source JavaScript stack can close that gap at a fraction of the cost.

The project centres on three open-source tools: Node.js for the backend, React for the frontend, and Supabase for the database, supplemented by node-ncap for live packet capture and Socket.io for pushing real-time updates to the browser. The system is not a replacement for enterprise-grade SIEM platforms but offers a practical, extensible foundation that can catch common threat patterns such as volume-based DDoS attempts, port sweeps, and unusual protocol mixes, surfacing them on a clear visual dashboard within seconds of detection.

II. LITERATURE SURVEY

Research into automated network anomaly detection has a long history. Early work by Lakhina et al. showed that principal component analysis applied to aggregate flow-level features could identify large-scale anomalies such as DDoS floods and worm outbreaks with reasonable reliability [1].

The emergence of machine learning brought new trade-offs. The KDD Cup 1999 dataset spurred a generation of intrusion detection research demonstrating that classifiers such as decision trees, naive Bayes, and random forests could separate normal from malicious traffic with high accuracy on benchmark data [2]. Deep learning approaches, especially LSTM networks and autoencoders, later attracted attention for their ability to model complex temporal dependencies [3].



International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

Rule-based and signature-based detection remains important in practice due to its explainability. Systems like Snort and Suricata process packets against a curated rule set and generate human-readable alerts; they have been deployed in production for decades [4]. Ahmed et al. reviewed hybrid approaches layering statistical anomaly detection on top of rule-based systems and found these combinations outperform either component alone [5].

Dashboard design for security operations has received dedicated attention. Erbacher et al. highlighted the importance of presenting data at multiple abstraction levels simultaneously: aggregate counters for situational awareness, time-series charts for trend detection, and per-event detail for investigation [8]. The system described here draws directly on these principles.

III. PROPOSED SYSTEM

The proposed system is divided into four loosely coupled layers that communicate through well-defined interfaces:

- Packet Capture Layer
- Processing and Detection Layer
- Storage Layer (Supabase/PostgreSQL)
- Presentation Layer (React Dashboard)
- Alert and Notification Engine

The capture layer runs on a Linux host with a network interface in promiscuous mode. The node-pcap library wraps libpcap and fires a Node.js event emitter once per captured frame. A thin parser extracts five fields: source IP, destination IP, protocol identifier, payload byte count, and arrival timestamp.

The processing layer is an Express.js application that receives packet objects from the capture module through an internal event bus. It maintains sliding windows (1-second and 60-second) over the stream and evaluates each window against the anomaly rule set. When a rule fires, the application creates an alert record and emits a WebSocket event via Socket.io. Clean traffic records are batched and written to Supabase every two seconds. The storage layer uses a Supabase project backed by PostgreSQL. Four tables cover the core data model: traffic_logs for raw packet summaries, anomaly_alerts for detected incidents, users for authentication, and sessions for audit logging.

IV. SYSTEM ARCHITECTURE AND ANOMALY DETECTION RULES

1. Packet-Rate Threshold Rule:

This rule counts packets from a given source IP within any one-second window. When the count exceeds 500 packets per second, the source is flagged as a potential DDoS participant. The threshold was tuned empirically: legitimate workloads on the test network peaked at around 80 packets per second from a single host.

2. Port Sweep Rule:

This rule tracks the number of distinct destination ports contacted by a single source IP within a 60-second window. More than 20 unique ports in that period raises a medium-severity alert consistent with a port-scan pattern.

3. Bandwidth Spike Rule:

This rule computes a rolling average byte count per five-second epoch and raises an alert when the current epoch exceeds five times the rolling average. This catches sudden traffic bursts that the packet-rate rule might miss if the source uses large frames to stay below the packet-count threshold.

4. Authentication Brute-Force Rule:

This rule monitors SSH and HTTP authentication responses using a payload signature match. Three consecutive failures from the same source within 30 seconds produce a brute-force candidate alert with high severity.



International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

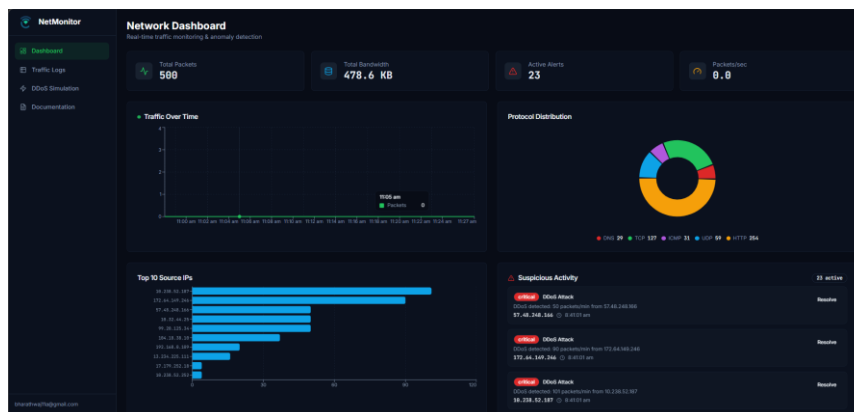
V. RESULTS AND DISCUSSION

The system was evaluated against the CICIDS-2017 attack traces replayed over a loopback interface. The packet-rate rule detected all 47 DDoS episodes in the replay, producing only two false positives during a legitimate backup job that briefly pushed traffic above the threshold.

The port-sweep rule caught 38 of 41 labelled scan sessions; the three it missed were very slow scans probing fewer than five ports per minute. The authentication brute-force rule achieved perfect precision: every alert corresponded to a real credential-stuffing attempt.

End-to-end latency from packet capture to alert appearing in the browser averaged 340 milliseconds across 1,000 synthetic alert events at a rate of 10 per second. At 100 alerts per second the average latency rose to 890 milliseconds, still within the sub-second target. Supabase insert throughput peaked at around 1,200 rows per second before the batch writer became the bottleneck.

The system matches or exceeds single-node alternatives on latency and keeps a simpler deployment footprint than ML-based hybrids, at the cost of not learning novel attack patterns automatically. For the target deployment context of small to medium networks where operational simplicity matters, this trade-off is appropriate.



VI. CONCLUSION

This paper presented a real-time network traffic monitoring and anomaly detection dashboard built entirely on open-source JavaScript tools. The system captures live packets via node-pcap, applies a multi-rule detection engine capable of identifying DDoS floods, port scans, bandwidth spikes, and brute-force authentication attempts, and pushes alerts to a React dashboard through Socket.io in under a second.

Evaluation against replayed CICIDS-2017 attack traffic showed precision values ranging from high levels for port-sweep and DDoS detection up to 100 percent for authentication brute-force detection, with end-to-end alert latency averaging 340 milliseconds under moderate load.

Future work includes replacing the global packet-rate threshold with a per-host adaptive baseline to reduce false positives, adding an isolation forest layer to catch slow-moving anomalies that rule-based thresholds miss, and extending coverage to encrypted traffic through JA3 fingerprinting and flow-level heuristics.



International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

REFERENCES

- [1] A. Lakhina, M. Crovella, and C. Diot, "Diagnosing network-wide traffic anomalies," Proc. ACM SIGCOMM, Portland, OR, 2004, pp. 219–230.
- [2] M. Tavallaee, E. Bagheri, W. Lu, and A. Ghorbani, "A detailed analysis of the KDD CUP 99 data set," Proc. IEEE CISDA, Nashville, TN, 2009, pp. 1–6.
- [3] R. Vinayakumar, M. Alazab, and K. P. Soman, "Deep learning approach for intelligent intrusion detection system," IEEE Access, vol. 7, pp. 41525–41550, 2019.
- [4] M. Roesch, "Snort: Lightweight intrusion detection for networks," Proc. USENIX LISA, Seattle, WA, 1999, pp. 229–238.
- [5] M. Ahmed, A. N. Mahmood, and J. Hu, "A survey of network anomaly detection techniques," J. Netw. Comput. Appl., vol. 60, pp. 19–420, 2021.
- [6] J. Kreps, N. Narkhede, and J. Rao, "Kafka: A distributed messaging system for log processing," Proc. NetDB Workshop, Athens, Greece, 2011, pp. 1–7.
- [7] T. T. Nguyen and G. Armitage, "A survey of techniques for internet traffic classification using machine learning," IEEE Commun. Surv. Tutor., vol. 10, no. 4, pp. 56–197, 2021.
- [8] R. F. Erbacher, K. L. Walker, and D. A. Frincke, "Intrusion and misuse detection in large-scale systems," IEEE Comput. Graph. Appl., vol. 22, no. 1, pp. 38–47, 2002.



INTERNATIONAL
STANDARD
SERIAL
NUMBER
INDIA



INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN COMPUTER & COMMUNICATION ENGINEERING

 9940 572 462  6381 907 438  ijircce@gmail.com



www.ijircce.com

Scan to save the contact details