International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14, Issue 11, CONFLUENCE - 2026

DeepGuard-NIDS: A Secure, Failover-Resilient Network Intrusion Detection and Real-Time Threat Analytics Platform

Aadya N.K
Dept. of Computer Applications
BMS College of Commerce and Management
Bengaluru, India
aadya2846@gmail

Adithi S.Rao
Dept. of Computer Applications
BMS College of Commerce and Management
Bengaluru, India
Chaitras04112006@gmail.com

Under the Guidance of: Shrinkhala Kumari

Abstract—Modern organizational infrastructure relies heavily on continuous network connectivity, exposing enterprise environments to high-volume, dynamic cyber threats that static rule-based systems fail to mitigate. This paper presents DeepGuard-NIDS, a modular web-based network intrusion detection platform developed for real-time traffic packet inspection, threat classification, and automated anomaly response. The system integrates socket-level flow monitoring, dynamic feature extraction, internal alert logs, multi-channel notifications, and an automated threat analysis dashboard under a role-based access control (RBAC) model. Its central design contribution is a resilient detection gate and a dynamic dual-engine database failover that transitions seamlessly from cloud PostgreSQL to local SQLite during network dropouts, ensuring uninterrupted logging and threat screening. Evaluated on real-time traffic simulations and standard benchmarks, the system exposes 117 management endpoints across 16 data models, maintaining high accuracy with zero server crashes under continuous operation.

***Index Terms*—network intrusion detection, cyber security, role-based access control, deep learning, anomaly detection, database failover, Flask.**

I. INTRODUCTION

Colleges and enterprise networks depend on several interacting workflows: monitoring incoming network traffic, detecting malicious packet payloads, triggering security alerts, distributing threat reports, and managing administrator privileges. When these run in separate logging utilities or standalone tools, security operations centers (SOC) struggle to enforce security policies consistently and maintain continuous uptime.

At modern academic and enterprise IT departments, this problem is sharpened by the necessity of high-availability monitoring: intrusion detection engines must maintain continuous operation even when primary cloud connectivity or central database servers face outages. Enforcing security policies manually or relying on fragile single-point database links leaves systems vulnerable, so DeepGuard-NIDS was built to apply automated detection and maintain an auditable record of every detected anomaly.

The contributions of this paper are: (i) an integrated web and network security architecture covering real-time sniffing, threat logging, and administrator workflows; (ii) a high-availability dual-engine database abstraction that prevents data loss during cloud database disconnections; (iii) a layered security design with strict role-based access control (RBAC) and sanitized upload handling; and (iv) comprehensive empirical evaluation across multiple operational modules.

The rest of the paper covers background (Section II), architecture (III), module design (IV), testing (V), and conclusions (VI).

II. BACKGROUND AND RELATED WORK

Network Intrusion Detection Systems (NIDS) commonly combine packet inspection, signature matching, and anomaly classification. Commercial and open-source platforms often isolate packet inspection from administrative reporting, forcing security staff to consolidate alerts manually. Web frameworks such as Flask make it practical to build lightweight, enterprise-ready systems that embed threat screening and policy enforcement directly into a unified portal [1].

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14, Issue 11, CONFLUENCE - 2026

Access control in the proposed platform follows the role-based model [2], where permissions attach to roles (e.g., Analyst, Administrator, Auditor) rather than individual users. Web-application hardening strictly adheres to the OWASP risk framework [3], particularly regarding broken access control and file upload security. Location and network checks employ standard cryptographic and routing algorithms, while kiosk and API authentication rely on keyed-hash message authentication (HMAC) [4].

III. SYSTEM ARCHITECTURE

The platform is structured as a multi-layered web and network engine. Security Analysts, Administrators, and Auditors interact with a Flask 3.1.3 application layer over HTTPS and WebSockets. Analysts authenticate via secure cookie sessions, while network agents send encrypted tokens, establishing parallel security guards across all endpoints.

Below the application layer, specialized engines handle distinct responsibilities: the Network Engine manages packet capture and inspection; the Detection Engine processes feature vectors and anomaly classifications; and the Operations Engine handles multi-channel alerts, user management, and system auditing. Table I summarizes the technology stack.

TABLE I
TECHNOLOGY STACK

Layer	Technology
Language	Python 3.10+
Web Framework	Flask 3.1.3, Flask-SocketIO
Data Access	Flask-SQLAlchemy 3.1.1
Database	PostgreSQL (Cloud), SQLite (Local Fallback)
Security	Werkzeug PBKDF2-SHA256, Flask-Login, HMAC
Analytics / Mail	Scikit-Learn, FPDF2, Flask-Mail

A. Data Model

The relational database schema contains 16 primary data models. Core entities include User, Role, NetworkSession, PacketLog, ThreatAlert, FirewallRule, and AuditTrail. Unique composite constraints prevent duplicate logging while maintaining explicit sequence tracking across packet streams.

B. Dual-Engine Database

DeepGuard-NIDS connects to a primary PostgreSQL instance hosted in the cloud. If network latency spikes or a connection timeout occurs, the system automatically redirects queries to a local SQLite database engine. Self-healing schema synchronization ensures data consistency once primary connectivity is restored.

C. Role-Based Access Control

Roles include Analyst, Administrator, and Auditor. Analysts monitor live network streams and triage alerts, Administrators configure network rules and threshold values, and Auditors inspect immutable system logs. Unauthorized role endpoint access yields an HTTP 403 response.

IV. MODULE DESIGN

A. Real-Time Packet Sniffing & Stream Analysis

A multi-threaded packet capture engine captures live PCAP traffic, extracting key statistical features including flow duration, packet headers, payload sizes, and flag patterns. Flow windows are normalized into feature vectors for real-time anomaly evaluation.

B. Threat Classification Engine

Feature vectors are processed through trained machine learning models (Random Forest and Deep Neural Networks) to classify traffic into normal behavior or specific attack vectors such as Denial of Service (DoS), Port Scanning, or Unauthorized Access. High-confidence threats trigger immediate automated rule recommendations.

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14, Issue 11, CONFLUENCE - 2026

C. Automated Alert Gate & Notifications

When an anomaly score exceeds configured threshold limits, the alert gate dispatches immediate notifications across WebSockets to the live dashboard and sends email alerts for high-severity incidents. Every alert generation event writes an unalterable log to the audit trail.

D. File Upload and System Security

Uploaded artifacts (e.g., custom network logs or firewall rules) are stored in private directories outside the web root under randomized UUID identifiers. File headers are validated against signature bytes to block executable or script injection attempts.

V. TESTING AND RESULTS

System functionality and security were evaluated using automated unittest suites. A total of 53 unit tests passed with 100% success, covering RBAC verification, threat classification accuracy, database failover continuity, and file security bounds.

TABLE II
AUTOMATED TEST SUITES

Suite	Tests	Coverage / Focus
Threat Analytics	8	Feature extraction, accuracy metrics, 403 checks
Alerts & Logging	10	Access control, event lifecycle, isolation
Failover Engine	10	Postgres-to-SQLite transition, audit continuity
Rule Management	10	RBAC validation, rule locking, integrity
System Endpoints	5	Endpoint availability, session management
All Modules	10	117 endpoints, stress test, zero crashes

A comprehensive smoke test called all 117 registered endpoints under load. The platform maintained 100% operational stability with zero server crashes or database lockouts.

VI. CONCLUSION AND FUTURE WORK

DeepGuard-NIDS demonstrates that real-time threat monitoring, administrative workflows, and fail-safe architecture can be effectively combined in a unified platform. The system's dual-engine database failover ensures continuous security operations even during central network disruptions. Future work includes integrating deep learning transformers for encrypted packet inspection and conducting extensive live enterprise field trials.

ACKNOWLEDGMENT

The authors thank Shrinkala Kumari for invaluable mentorship and guidance, as well as the faculty and management of BMS College of Commerce and Management for providing institutional resources and technical requirements.

REFERENCES

[1] M. Grinberg, Flask Web Development, 2nd ed. Sebastopol, CA, USA: O'Reilly Media, 2018.

[2] R. S. Sandhu, E. J. Coyne, H. L. Feinstein, and C. E. Youman, "Role-based access control models," Computer, vol. 29, no. 2, pp. 38–47, Feb. 1996.

[3] OWASP Foundation, "OWASP Top 10:2021," 2021. [Online]. Available: https://owasp.org/Top10/

[4] H. Krawczyk, M. Bellare, and R. Canetti, "HMAC: Keyed-hashing for message authentication," IETF, RFC 2104, Feb. 1997.

(This work is licensed under a Creative Commons Attribution 4.0 International License.)