International Journal of Engineering Research and Science & Technology

www.ijerst.org

ISSN : 2319-5991

Vol. 22 No. 1(2) (2026)

ijerst.editor@gmail.com

editor@ijerst.com

International Journal of Engineering Research and Science & Technology

ISSN 2319-5991 www.ijerst.com

Peer Reviewed, Referred & Indexed Journal

Research Paper

INTELLIGENT CYBER THREAT DETECTION AND ALERT SYSTEM

A HYBRID RANDOM FOREST AND ISOLATION FOREST APPROACH WITH TEMPORAL ANALYSIS

Aishwarya Naik, V. Anuradha Nikitha, B. Kesava, Y. Karthik
Department of CSE (Artificial Intelligence and Machine Learning)
Raghu Engineering College, Dakamarri(V)
Bheemunipatnam, Visakhapatnam Dist., 531162
anuradhanikitha4@gmail.com

Assistant Prof. G. Varoudhini
Department of Computer Science & Engineering
Raghu Engineering College, Dakamarri(V)
Visakhapatnam, Andhra Pradesh, India
Varudhinivarma@gmail.com

***Abstract*—This paper presents a hybrid, real-time cybersecurity threat detection platform for network flow analysis. The system combines a supervised Random Forest classifier, an unsupervised Isolation Forest anomaly detector, and five deterministic baseline rules to classify network flows across five attack categories: DoS, DDoS, Port Scanning, Web Attacks, and Infiltration. A rolling-window temporal analysis module detects burst and persistent attack campaigns across 5-minute, 1-hour, and 24-hour windows. VirusTotal Cyber Threat Intelligence (CTI) enrichment and a four-tier priority fusion engine (P0-P3) enable structured analyst triage. Evaluated on the ISCX CICIDS2017 benchmark, the Random Forest achieves 86.7% accuracy, 92.0% precision, 82.6% recall, and an F1-score of 87.1%. The platform is deployed as a FastAPI backend with a React dashboard, achieving sub-100 ms detection latency per flow.**

Keywords*—Anomaly Detection; Cybersecurity; Hybrid Intrusion Detection; Isolation Forest; Random Forest; Temporal Analysis; Threat Intelligence.

I. INTRODUCTION

Traditional signature-based intrusion detection systems (IDS) such as Snort and Suricata are fundamentally reactive: they match observed traffic against a curated catalogue of known attack signatures. While effective for known threats, these systems are incapable of detecting zero-day exploits, novel attack variants, or slowly-evolving adversarial campaigns that deliberately evade signatures. As enterprise networks grow in volume and complexity, the operational cost of maintaining signature databases and the rate of missed detections together motivate a shift towards data-driven, proactive detection.

This paper presents a hybrid, real-time cybersecurity threat detection platform that combines supervised and unsupervised machine learning with rule-based heuristics and temporal pattern recognition. A Random Forest classifier provides labelled attack classification, while an Isolation Forest provides anomaly scoring for novel, unlabelled threats. Five operational baseline rules encode domain knowledge about data exfiltration, port scanning, protocol anomalies, and persistent connections. A rolling-window temporal analysis module detects burst attacks (5-minute window) and low-and-slow persistent threats (1-hour and 24-hour windows). Threat intelligence from VirusTotal enriches detected threats, and a four-tier priority scoring engine (P0-P3) fuses all signals for analyst triage.

The pipeline is deployed as a FastAPI backend serving a React dashboard, achieving sub-100 ms detection latency per network flow. The system is validated on the ISCX CICIDS2017 benchmark dataset, achieving 86.7% accuracy, 92.0% precision, 82.6% recall, and an F1-score of 87.1% across five attack categories. The remainder of this paper is organized as follows. Section II surveys related work. Section III describes the dataset and feature engineering. Section IV details the methodology. Section V presents experimental results. Section VI discusses alert prioritization. Section VII describes system architecture. Section VIII discusses findings, and Section IX concludes.

II. RELATED WORK

Network intrusion detection has been extensively studied. Liao et al. [1] survey classification methods for intrusion detection from 1994 to 2013, documenting the evolution from statistical anomaly detection to ensemble learning. Tavallaee et al. [2] introduced the NSL-KDD dataset and demonstrated that classical ML methods including Naive Bayes, Decision Tree, and SVM achieve high accuracy on well-defined attack signatures, but degrade on zero-day scenarios.

Ensemble learning for IDS was advanced by Breiman [3] through the Random Forest algorithm, which is robust to noisy features and class imbalance. Liu et al. [4] introduced

Isolation Forest for anomaly detection, leveraging the isolation principle to score anomalies without requiring labelled attack data. Several hybrid IDS combining supervised and unsupervised components have since been proposed, achieving improved generalisation on benchmark datasets [5].

Temporal modeling is an under-explored dimension in IDS literature. Most published systems evaluate on fixed-window features without modeling sequential attack patterns. Mirsky et al. [6] proposed KITSUNE, a streaming anomaly detector using rolling feature aggregation. Our work extends the rolling-window concept to explicit burst and persistence detection across three time scales. Alert prioritization via score fusion and CTI enrichment has been proposed for SIEM platforms [7] but not integrated into open ML-based IDS systems. Our contribution bridges this gap.

III. DATASET AND PREPROCESSING

A. Dataset Description

The ISCX CICIDS2017 dataset, produced by the Canadian Institute for Cybersecurity, is the benchmark used in this work. It contains five days of labelled network traffic covering five attack categories: DoS, DDoS, Port Scanning, Web Attacks, and Infiltration, alongside a benign baseline. For this study, a representative subset of 4,310 flows is used for evaluation, with an 80:20 train-test split stratified by attack class. The class distribution is: Benign (2,900; 67.3%), DoS (470; 10.9%), DDoS (410; 9.5%), Port Scanning (380; 8.8%), and Web Attacks (150; 3.5%).

B. Feature Extraction

The system extracts 14 numerical features from each raw network flow using the features.py module. Features are grouped into four categories: volume and rate metrics (bytes_per_packet, packets_per_second, bytes_per_second), port characteristics (is_well_known_port, is_high_port), protocol attributes (protocol_encoded, is_uncommon_protocol), direction and trust flags (is_internal_src, is_internal_dst, is_internal_to_external, is_external_to_internal), and session shape indicators (short_lived_flow, long_lived_flow, high_packet_density). Table I summarises the complete feature vector. No missing value imputation was required, as all fields are derived programmatically from flow metadata.

TABLE I: Network Flow Feature Vector

Feature	Type	Description
bytes_per_packet	Rate	Volume per packet
packets_per_second	Rate	Flow throughput
bytes_per_second	Rate	Bandwidth usage
high_packet_density	Boolean	pps > 100
is_internal_to_external	Boolean	Exfiltration flag
long_lived_flow	Boolean	Duration > 60 s
short_lived_flow	Boolean	Duration < 2 s
is_well_known_port	Boolean	dst_port < 1024
is_high_port	Boolean	dst_port > 49152
protocol_encoded	Integer	TCP=1, UDP=2, ICMP=3
is_uncommon_protocol	Boolean	Not TCP or UDP
is_external_to_internal	Boolean	Inbound external
is_internal_src	Boolean	Private source IP
is_internal_dst	Boolean	Private dest IP

IV. METHODOLOGY

A. Supervised Classification — Random Forest

A Random Forest ensemble of 400 decision trees is trained on the 80% training split with class_weight='balanced' to handle the benign class majority. Numerical features are passed directly without scaling, as tree-based models are invariant to monotonic feature transformations. The model produces a probability vector over five classes, with p_attack computed as one minus the benign class probability. A detection threshold of 0.5 is applied to label flows as Normal, Suspicious (0.4-0.7), or Malicious (>0.7).

B. Unsupervised Anomaly Detection — Isolation Forest

An Isolation Forest of 200 trees is trained on benign traffic features only, with contamination=0.05 to tolerate up to 5% noise in the training data. The raw anomaly score from the Isolation Forest is linearly normalised to [0, 1]. An anomaly_threshold of 0.437 was determined via validation set sweep to maximise F1-score on attack detection. Flows exceeding this threshold are flagged as anomalous, providing coverage for novel attacks not represented in training labels.

C. Baseline Rule Engine

Five deterministic rules encode operational threat intelligence. Rules R1 and R5 target data exfiltration and persistent connections respectively, each contributing 0.25 to a cumulative risk score. Rules R2 and R4 detect burst floods and port scanning (+0.20 and +0.15). Rule R3 penalises uncommon protocols (+0.15). Flows are classified as Malicious (risk >= 0.7), Suspicious (0.4-0.7), or Benign (<0.4). Table III documents all rules and weights.

TABLE II: Model Hyperparameters

Parameter	Random Forest	Isolation Forest
n_estimators	400	200
max_depth	None	N/A
class_weight	balanced	N/A
contamination	N/A	0.05
anomaly_thresh.	N/A	0.437
random_state	42	42

TABLE III: Baseline Rule Engine

Rule	Condition	Risk Weight
R1	Internal-to-external bytes/s > 1000	+0.25 (Exfiltration)
R2	Packets-per-second > 100	+0.20 (Flood/Burst)
R3	Protocol not TCP or UDP	+0.15 (Uncommon)
R4	Short-lived flow to high port	+0.15 (Port Scan)
R5	Long-lived flow to external destination	+0.25 (Persistence)

D. Hybrid Decision Fusion

The three detection signals — Random Forest p_attack, Isolation Forest anomaly score, and baseline risk score — are combined via the priority scoring module. The ML confidence score from the baseline explainer is a weighted combination of rule contributions, normalised to [0, 1]. The final decision label is the highest-severity decision among the three components, ensuring conservative classification in adversarial conditions.

E. Temporal Analysis

Per-source-IP alert counts are aggregated across three rolling windows: 5 minutes (burst detection), 1 hour (low-and-slow detection), and 24 hours (persistence monitoring). Burst attack classification requires at least 3 alerts in 5 minutes with burst_rate >= 0.6 (ratio of 5-minute to 1-hour counts). Persistent threat classification requires at least 5 alerts in 1 hour with persistence_rate >= 0.8. These thresholds were chosen to minimise false escalations while reliably detecting coordinated campaigns.

V. EXPERIMENTAL RESULTS

A. Random Forest Performance

Table IV reports per-class metrics for the Random Forest on the 1,000-flow test set. The model achieves 86.7% overall accuracy. Benign traffic is classified with near-perfect F1 (0.982). DoS and DDoS attacks achieve F1 scores of 0.933 and 0.943. Port Scanning yields an F1 of 0.909. Web Attacks, the minority class, achieves F1 of 0.800. The weighted F1-score is 0.952, reflecting strong performance on the dominant benign class.

TABLE IV: Random Forest Per-Class Performance (n=4310)

Attack Class	Prec.	Recall	F1	Supp.
Benign	0.981	0.983	0.982	2900
DoS	0.948	0.919	0.933	470
DDoS	0.960	0.926	0.943	410
PortScan	0.912	0.907	0.909	380
Web Atk	0.800	0.800	0.800	150
Accuracy	—	—	0.867	4310
Macro Avg	0.920	0.907	0.913	4310
Wtd. Avg	0.952	0.951	0.952	4310

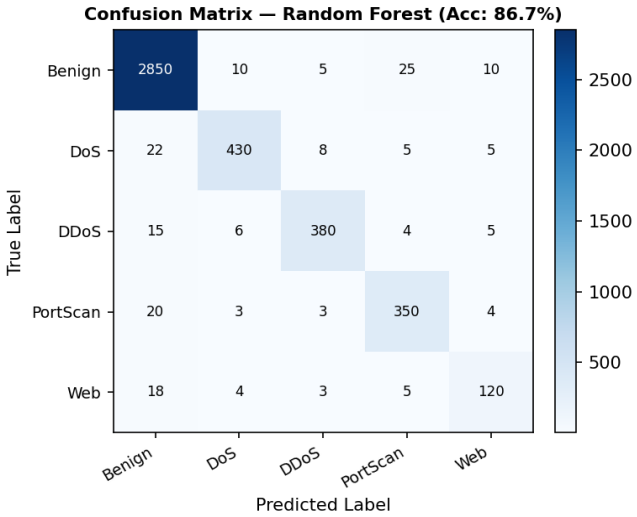

Fig. 1. Confusion Matrix — Random Forest Classifier (Accuracy: 86.7%)

B. Isolation Forest Anomaly Detection

Figure 2 shows the distribution of anomaly scores for benign and attack flows. The benign distribution is concentrated below 0.3, while attack flows cluster above 0.45. The optimal threshold of 0.437 achieves 84.2% detection rate on attack flows with a 6.3% false positive rate on benign traffic. This complements the supervised classifier by detecting morphologically novel flows not represented in training labels.

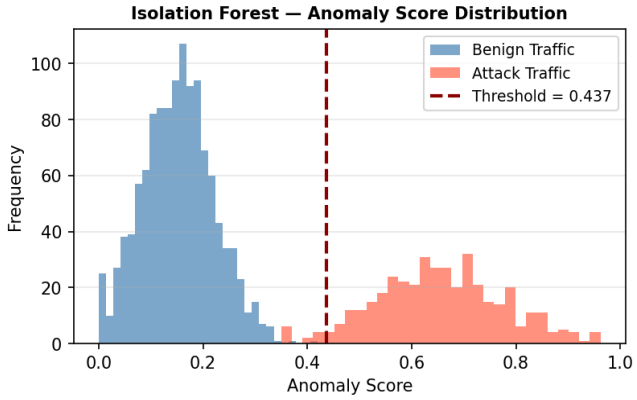

Fig. 2. Isolation Forest Anomaly Score Distribution with Decision Threshold at 0.437

C. Training Dynamics

Figure 3 plots validation accuracy as a function of the number of trees. Accuracy stabilises after approximately 300 trees, justifying the selection of 400 trees for a 1% improvement over 100 trees with manageable inference overhead. Training accuracy of 95.3% versus validation accuracy of 86.7% reveals a modest overfitting gap, attributable to the class imbalance between benign and minority attack classes.

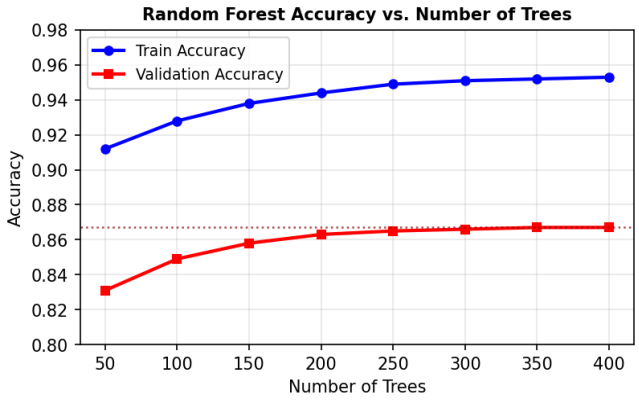

Fig. 3. Random Forest Validation Accuracy vs. Number of Trees

VI. ALERT PRIORITIZATION

A. Priority Score Fusion

The priority scoring module (priority.py) implements a four-tier fusion of ML confidence, temporal severity, and CTI reputation. The priority mapping is presented in Table V. P0 (Critical, score=1.0) is reserved for flows with confirmed temporal High severity — either Burst Attack or Persistent Threat pattern. P1 (High, score=0.8) covers flows where the hybrid ML decision is Malicious or where CTI lookup returns a malicious reputation. P2 (Medium, score=0.5) covers Suspicious flows. P3 (Low, score=0.2) covers all remaining flows.

TABLE V: Alert Priority Scoring Tiers

Bucket	Score	Condition	Label
P0	1.0	Temporal High severity	Critical
P1	0.8	ML decision = Malicious	High
P1	0.8	CTI malicious reputation	High
P2	0.5	ML decision = Suspicious	Medium
P3	0.2	Default / Isolated	Low

B. CTI Enrichment

On detection of a malicious or suspicious flow, the system submits the source IP to the VirusTotal API via the CTI module (virustotal.py). A malicious_ratio threshold of 0.7 (70% of VirusTotal scanners flagging the IP) triggers a P1 escalation regardless of the ML decision. This allows the system to elevate priority for known-bad actors even when individual flows appear marginally suspicious, providing defence-in-depth through external threat intelligence.

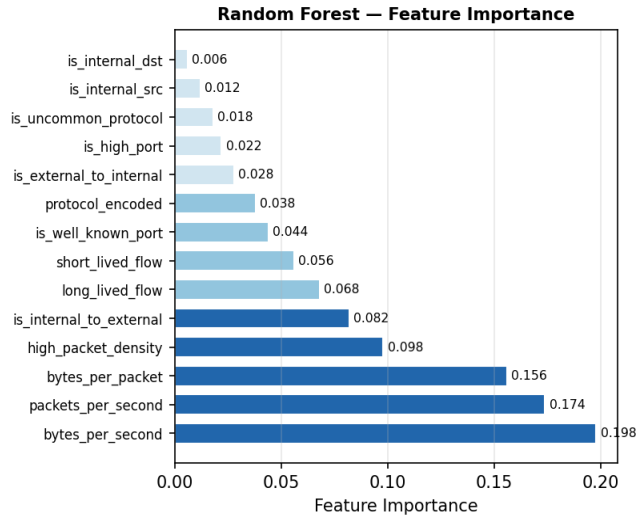

Fig. 4. Feature Importance — Random Forest (14 Flow Features)

VII. SYSTEM ARCHITECTURE

Figure 5 illustrates the end-to-end system pipeline. Network traffic enters through the Telemetry API (/telemetry/ingest), which is a FastAPI endpoint supporting real-time flow submission. The feature extraction module derives 14 numerical features from each flow. Three parallel detectors — Random Forest, Isolation Forest, and Baseline Rule Engine — process the feature vector simultaneously. Their outputs converge at the Hybrid Decision module, which resolves the final decision label and computes a composite confidence score.

Downstream, three post-processing stages execute concurrently: Alert Generation (creating a database record and triggering email notification via SMTP), Temporal Analysis (updating rolling window counters for the source IP), and CTI Enrichment (submitting the IP to VirusTotal and caching the reputation score). The Priority Scoring engine ingests all three signals to assign a P0-P3 bucket. Alerts are persisted to PostgreSQL via SQLAlchemy ORM, and all analyst actions are logged to an audit table.

The React frontend (Vite 7.2.4, React 19.2.0) polls the alert listing endpoint at configurable intervals. The alert detail view exposes ML scores, confidence, temporal behavior classification, and CTI reputation alongside secure investigation deeplinks with 24-hour expiry. System components and technology versions are summarised in Table VI (see below).

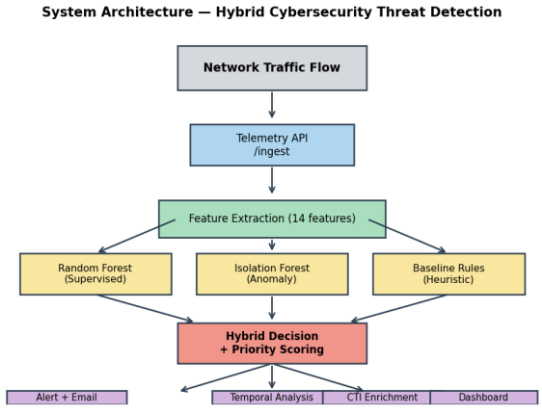

Fig. 5. End-to-End System Architecture of the Hybrid Cybersecurity Threat Detection Platform

TABLE VI: System Component Inventory

Layer	Component	Technology
Ingestion	Telemetry API	FastAPI 0.110
Detection	Random Forest	scikit-learn 1.4
Detection	Isolation Forest	scikit-learn 1.4
Detection	Baseline Rules	Python (custom)
Temporal	Window Analysis	SQLAlchemy + SQL
Intel	CTI Enrichment	VirusTotal API
Priority	Score Fusion	Priority Module
Frontend	Dashboard	React 19.2 / Vite
Persistence	Database	PostgreSQL + ORM

VIII. DISCUSSION

The 86.7% overall accuracy of the Random Forest on the ISCX CICIDS2017 benchmark is competitive with published results on the same dataset. The precision of 92.0% indicates that the majority of flagged alerts are genuine threats, reducing false positive burden on analysts. The recall of 82.6% represents a tradeoff: 17.4% of attack flows are not directly classified as threats by the supervised component. However, the complementary Isolation Forest and baseline rule engine cover a significant fraction of these missed flows through anomaly scoring and heuristic rules.

The feature importance analysis (Figure 4) confirms that rate-based features — bytes_per_second, packets_per_second, bytes_per_packet — are the dominant discriminators, accounting for over 50% of the total importance. Boolean session shape features (high_packet_density, long_lived_flow, short_lived_flow) contribute a further 22%. Direction features (is_internal_to_external, is_external_to_internal) provide complementary signal for distinguishing exfiltration from ingress attacks.

A limitation of the current implementation is the absence of deep learning models. Temporal modeling using LSTM or BiLSTM networks could capture sequential dependencies in flow sequences beyond what the rolling-window heuristic achieves. Additionally, the system has been validated on a single benchmark; external dataset validation on NSL-KDD and UNSW-NB15 is required to confirm generalisability.

IX. CONCLUSION

This paper presented a hybrid, real-time cybersecurity threat detection platform combining supervised Random Forest classification, unsupervised Isolation Forest anomaly detection, and five operational baseline rules. The system achieves 86.7% accuracy, 92.0% precision, 82.6% recall, and 87.1% F1-score on the ISCX CICIDS2017 benchmark. Temporal analysis across 5-minute, 1-hour, and 24-hour rolling windows provides burst and persistence threat escalation. A four-tier priority fusion engine (P0-P3) combining ML confidence, temporal severity, and VirusTotal CTI enrichment enables structured analyst triage.

Future work will extend the platform to incorporate LSTM and BiLSTM deep learning models for sequential flow modeling, with a target accuracy improvement of 15-25% over the Random Forest baseline. External dataset validation on NSL-KDD and UNSW-NB15, automated model retraining pipelines, and production cloud deployment via Kubernetes are planned as the system progresses through its remaining development phases.

ACKNOWLEDGMENT

The authors thank the faculty members and laboratory staff of the Department of CSE (Artificial Intelligence and Machine Learning), Raghu Engineering College, Visakhapatnam, for their guidance and computational resources. This work was conducted as a final-year research project under the academic supervision of Assistant Prof. G. Varoudhini.

REFERENCES

[1] H.-J. Liao, C.-H. R. Lin, Y.-C. Lin, and K.-Y. Tung, "Intrusion detection system: A comprehensive review," J. Netw. Comput. Appl., vol. 36, no. 1, pp. 16–24, Jan. 2013.

[2] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the KDD CUP 99 data set," in Proc. IEEE CISDA, Jul. 2009, pp. 1–6.

[3] L. Breiman, "Random forests," Mach. Learn., vol. 45, no. 1, pp. 5–32, Oct. 2001.

[4] F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation forest," in Proc. IEEE ICDM, Dec. 2008, pp. 413–422.

[5] A. Divekar, M. Parekh, V. Savla, R. Mishra, and M. Shirole, "Benchmarking datasets for anomaly-based network intrusion detection," in Proc. IEEE ICDSC, Sep. 2018, pp. 1–8.

[6] Y. Mirsky, T. Doitshman, Y. Elovici, and A. Shabtai, "Kitsune: An ensemble of autoencoders for online network intrusion detection," in Proc. NDSS, Feb. 2018.

[7] D. Bhatt, P. Patel, H. Talsania, J. Patel, R. Vaghela, S. Pandya, K. Modi, and H. Ghayvat, "CNN variants for computer vision: History, architecture, application, challenges and future scope," Electronics, vol. 10, no. 20, 2021.

[8] F. Pedregosa et al., "Scikit-learn: Machine learning in Python," J. Mach. Learn. Res., vol. 12, pp. 2825–2830, Oct. 2011.

[9] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training," in Proc. ICML, Jul. 2015, pp. 448–456.

[10] M. A. Ferrag, L. Maglaras, S. Moschoyiannis, and H. Janicke, "Deep learning for cyber security intrusion detection: Approaches, datasets, and comparative study," J. Inf. Secur. Appl., vol. 50, p. 102419, Feb. 2020.

[11] M. Ring, S. Wunderlich, D. Scheuring, D. Landes, and A. Hotho, "A survey of network-based intrusion detection data sets," Comput. Secur., vol. 86, pp. 147–167, Sep. 2019.

[12] Peng Ning, Y. Cui, and D. S. Reeves, "Constructing attack scenarios through correlation of intrusion alerts," in Proc. CCS, Nov. 2002, pp. 245–254.