Journal of Trends in Computer Science and Smart Technology (ISSN: 2582-4104)
www.irojournals.com/tcsst/

Tri-Level Network Attack Risk Stratification Using IDS-Contextual Ensemble Learning

Reeta Mishra[1*], Neelu Chaudhary[2]

Department of Computer Science and Technology, Manav Rachna University, Faridabad, Haryana, India.

E-mail: [1*]ree76shok@gmail.com, [2]neelu@mru.edu.in

Orcid ID: [1*]0009-0001-7219-425X, [2]0000-0002-4481-1329

Abstract

Advanced Persistent Threats (APTs) remain among the most damaging risks to enterprise networks: they evade signature-based defenses, persist inside compromised environments, and cause substantial harm. Most machine learning approaches to intrusion detection treat the problem as binary classification and therefore offer little support for the risk prioritisation that Security Operations Centre (SOC) analysts actually require. This paper presents an Intrusion Detection System (IDS)-contextual ensemble learning framework that assigns network traffic to three operationally meaningful risk tiers: High, Medium and Low. Sixteen domain-knowledge engineered features are combined with IDS-generated contextual annotations, raw behavioural flow measurements and temporal components to form a 36-dimensional input representation. A one-way analysis of variance (ANOVA) with eta-squared effect sizing, performed before any model is trained, shows that none of the nine raw behavioural features carries meaningful discriminating power for risk-tier assignment, which establishes the need for IDS context. Five classifiers are evaluated on a balanced dataset of 125,000 APT flow records. Extra Trees performs best, reaching 89.35% accuracy, a Macro F1 of 0.8941, a Matthews Correlation Coefficient of 0.8404 and a macro Area Under the Curve (AUC) of 0.9829, at an inference cost of approximately 42 microseconds per record. A six-configuration ablation study shows that removing the IDS annotation alone reduces Macro F1 from 0.8941 to 0.3341, the chance level for three balanced classes. Robustness is confirmed by five-fold stratified cross-validation and by a fairness audit across the four network protocol types present in the data.

Keywords: Advanced Persistent Threats (APTs); Ensemble Learning; Intrusion Detection System (IDS)-Contextual Features; Security Operations Centre (SOC) Alert Triage; Tri-Level Risk Stratification.

1. Introduction

Advanced Persistent Threats (APTs) are the primary cybersecurity threat of the decade. Whereas traditional “drive-by” intrusions typically involve a quick-and-dirty affair, an APT is a well-planned, long-term attack that intends to gain a foothold inside the victim's network, quietly find valuable information, and export that data back to the intruder [1]. But high-profile

* Corresponding Author

DOI: https://doi.org/10.36548/jtcsst.2026.3.018
Received: 30.05.2026, received in revised form: 04.07.2026, accepted: 22.07.2026, published: 04.08.2026
© 2026 Inventive Research Organization. This is an open access article under the Creative Commons Attribution-NonCommercial International (CC BY-NC 4.0) License

campaigns, including the SolarWinds supply-chain compromise and the Microsoft Exchange ProxyLogon exploitation, revealed even well-resourced and mature security programs were not infallible in 2020 or 2021. The monetary stakes have gone up in lockstep: IBM Security [2] values the average total cost of an enterprise breach at more than USD 4.5 million–up year over year.

To address this threat landscape, the machine learning research community has focused on enhancing binary intrusion detection (malicious vs benign) as opposed to the much more operationally relevant question of prioritising risk. That concentration has led to extremely effective detectors, which are, however, of limited operational value since detection and triage are two separate issues. A typical modern Security Operations Centre (SOC) will get hundreds or thousands of alerts per day from the deployed IDS and SIEM platforms. Without an automated triage layer that classifies each alert with a risk priority, analysts are forced to process them one after another, which is time-consuming, prone to errors and draining. As a direct result of this overload, according to CrowdStrike [3], over 40% of high-severity incidents go uninvestigated within the window that incident-response best practice requires. The real issue is thus whether the most alarming detections can be displayed in a timely manner rather than if an attack can be spotted.

A growing body of work has moved beyond binary detection toward multi-class threat classification, producing outputs that identify the attack category (for example, Denial-of-Service, Reconnaissance, or Shellcode). These outputs are genuinely useful, yet they still do not answer the question a SOC analyst asks first: *How urgently must I act on this alert?* Translating an attack type into an appropriate response priority requires a layer of domain reasoning that automated alert-prioritisation research has only recently begun to address and that comparatively few systems formalize and validate end-to-end. More fundamentally, almost all existing approaches discard the risk-consequence annotations that modern IDS platforms already generate alongside their flow measurements—annotations that encode years of accumulated expert knowledge about what different attack patterns actually mean for an organisation. Treating this signal as disposable metadata, we argue, is the central missed opportunity in current ML-based intrusion detection.

Against this background, this work is organised around three research questions:

- **RQ1:** Do raw behavioural network features carry statistically meaningful discriminating power for APT risk-tier classification, and can this be quantified formally *before* training any model?
- **RQ2:** Does including IDS-generated impact-annotation fields as complementary input features substantially improve tri-level risk classification compared with a behavioural-only approach?
- **RQ3:** Is the resulting framework robust across different classifiers, data partitions, and network protocols, and does it run fast enough for real-time SOC deployment?

To answer these questions, this paper makes the following specific contributions:

1. We formulate SOC alert triage as a tri-level (High, Medium, Low) APT risk-stratification task and provide a formal specification of the tri-level risk-mapping function and the soft-voting ensemble decision rule, aligned with the NIST SP 800-

61r3 incident-response tiering and the MITRE ATT&CK tactic-level severity framework so that the output is operationally interpretable and reproducible.

2. We introduce a model-agnostic pre-training feature audit—one-way ANOVA with eta-squared effect sizing and a Spearman rank-correlation bias check—that quantifies the discriminating power of candidate features before any classifier is trained, providing a replicable diagnostic for ML-IDS pipelines.

3. We design a 36-dimensional IDS-contextual feature pipeline that treats IDS-generated impact annotations as first-class inputs alongside 16 domain-knowledge-derived features, and we isolate the contribution of each feature group through a six-configuration ablation study that identifies the IDS annotation as the dominant source of predictive signal.

4. We conduct a comprehensive empirical evaluation—five classifiers across seven metrics, five-fold stratified cross-validation, a protocol-level fairness audit across ICMP/IGMP/TCP/UDP traffic, and a computational-complexity analysis (inference at approximately 42 microseconds per record)—with Extra Trees achieving 89.35% accuracy, Macro F1 = 0.8941, MCC = 0.8404, and ROC-AUC = 0.9829. All performance claims are established within this single, consistently evaluated setting: the seven published systems reviewed in Section 2 are positioned qualitatively rather than compared numerically, because their datasets, task formulations, evaluation protocols and metric definitions all differ from those used here (Section 7.3).

The remainder of the paper is structured as follows. Section 2 reviews related work. Section 3 describes the dataset and the statistical pre-analysis that motivates the design. Section 4 presents the proposed framework and its mathematical formalisation. Section 5 details the experimental setup, and Section 6 reports the results. Section 7 discusses the findings and their operational implications, and Section 8 concludes.

2. Related Work

This section reviews prior work along four axes that together define the gap addressed by this paper: staged APT detection, machine learning for network intrusion detection, recent attention- and transformer-based models, and severity-aware risk assessment. It closes with a structured qualitative positioning (Table 1) that locates the proposed framework relative to seven representative systems on the basis of capability rather than reported scores.

2.1 APT Detection: From Signatures to Learning-Based Systems

The research community has attempted to follow two different approaches to staged APT detection: the Cyber Kill Chain [1] and the MITRE ATT&CK knowledge base [4]. Both provide models of attacker behaviour and early detection systems based on them were signature-based, making them extremely vulnerable to unknown and polymorphic attacks. Alshamrani et al. [5] reviewed 47 machine learning-based APT detection studies published up to 2019 and determined that learning-based detectors significantly increased recall compared to signature-based detectors. However, they were still not interpretable enough for operational use. Independently, Milajerdi et al. [6] illustrated with HOLMES how APT campaigns could be tracked in real-time by correlating suspicious information flows on a system-provenance

graph, while Nguyen et al. [7] illustrated with DÏoT how federated, self-learning anomaly detection could be performed without raw traffic exchange across distributed deployments, while maintaining data privacy at an acceptable performance cost. There is a consistent theme in this literature: the capacity to detect a problem has steadily improved, but the capacity to translate a detection into a prioritised action has been slow to catch up and is still very manual.

2.2 Machine Learning for Network Intrusion Detection

Ensemble methods have consistently outperformed individual classifiers on established network intrusion-detection benchmarks [8]. Sharafaldin et al. [9] introduced the CICIDS-2017 dataset and showed that behavioural flow features extracted from NetFlow and IPFIX records substantially outperform packet-level features for attack classification. Building on such benchmarks, Imrana et al. [10] applied a bidirectional LSTM to NSL-KDD and reached 94.3% accuracy on multi-class detection, but their architecture offers no pathway to risk prioritisation. Lo et al. [11] instead modelled network flows as a graph and obtained strong detection with their E-GraphSAGE graph-neural-network on flow-based UNSW-NB15 data, although graph-neural-network inference latency typically sits above the threshold that real-time SOC processing allows. More recently, Xi et al. [12] reported very high accuracy (above 99%) on UNSW-NB15 and NSL-KDD with a multi-scale transformer, but their model distinguishes malicious from benign traffic rather than assigning operational risk tiers. Across these systems, higher accuracy has generally come at the cost of either operational latency or output granularity.

2.3 Transformer and Attention-Based Approaches (2024–2025)

Recent work has explored attention mechanisms for richer feature interaction. Wu et al. [13] proposed RTIDS, a robust transformer-based intrusion detection system with a hierarchical self-attention design, reporting F1-scores above 0.98 on CICIDS-2017 and CIC-DDoS2019, while feature-selection pipelines such as the XGBoost-based approach of Kasongo and Sun [14] reached competitive accuracy on UNSW-NB15 with a much lighter computational footprint. Neither line of work incorporates IDS-generated contextual fields, and neither maps its outputs to operational risk tiers. Transformer and language-model adaptations are designed primarily for sequence or threat-intelligence text rather than tabular flow data, and are therefore less naturally suited to the present setting. For the problem at hand—tabular flow features mapped to risk tiers—structured ensemble learning remains the most appropriate modelling choice, combining competitive accuracy with low inference latency and inherent feature-importance transparency.

2.4 Risk Assessment and Severity-Aware Classification

Static risk-scoring frameworks, such as the Common Vulnerability Scoring System (CVSS) [15] and DREAD (Damage, Reproducibility, Exploitability, Affected users, Discoverability), assign fixed metrics based on vulnerability characteristics rather than observed traffic behaviour, which makes them unsuitable for real-time triage. Mahbooba et al. [16] demonstrated that a decision-tree classifier can reach 81.4% accuracy for IDS severity classification, but their evaluation lacks cross-validation and does not report protocol-level performance. Automated alert prioritisation has itself become an active research area recently cataloged in a systematic survey of prioritisation criteria and methods for security operations centers [17] with a growing body of context- and severity-aware triage systems aimed at

reducing analyst alert fatigue. The contribution here is therefore not triage in the abstract, but the specific pairing of an explicit three-tier risk mapping with a formal pre-training discriminating-power audit and the treatment of IDS-generated context as a first-class feature—a combination that, to the best of our knowledge, has not previously been validated across five classifiers, six ablation configurations, four network protocols, and a full computational-complexity analysis.

2.5 Qualitative Positioning of the Proposed Framework

Table 1 characterises seven representative studies along the dimensions that determine what each system actually does: dataset, method, task formulation, label space, whether the output is an operational risk tier, and whether IDS-generated context is used as an input feature. It is deliberately a capability matrix and not a performance comparison. The accuracy and F1 figures reported by these studies are not reproduced here because each was obtained on a different dataset, for a different task, under a different evaluation protocol, and with differently defined metrics; placing them side by side would invite an inference that the underlying experiments do not support. Section 7.3 develops this point in full. Read as a capability matrix, two observations stand out. First, every prior system operates in either binary or generic multi-class mode; none produces operationally defined risk tiers. Second, none of the surveyed systems treats IDS-generated context as a first-class input feature. The proposed framework is distinctive in combining tri-level risk stratification with explicit IDS-context integration—the combination most directly aligned with the needs of SOC triage.

Table 1. Capability matrix of related IDS and APT detection approaches (not a performance comparison)

Study (Year)	Dataset	Method	Task formulation	Label space	Operational risk tiers	IDS context as input feature
Imrana et al. [10] 2021	NSL-KDD	BiLSTM	Attack-type detection	Generic multi-class	No	No
Kasongo & Sun [14] 2020	UNSW-NB15	XGBoost feature selection + DT	Malicious vs. benign detection	Binary	No	No
Mahbooba et al. [16] 2021	Custom IDS log	Decision tree (XAI)	IDS severity classification	Generic multi-class	No	No
Lo et al. [11] 2022	NF-UNSW-NB15	E-GraphSAGE (GNN)	Malicious vs. benign detection	Binary	No	No
Wu et al. (RTIDS) [13] 2022	CICIDS-2017; CIC-DDoS2019	Transformer (self-attention)	Attack-type detection	Generic multi-class	No	No
Xi et al. [12] 2024	UNSW-NB15; NSL-KDD	Multi-scale transformer	Malicious vs. benign detection	Binary	No	No
Nguyen et al. (DÏoT) [7] 2019	IoT device testbed	Federated self-learning anomaly detection	Anomaly vs. normal detection	Binary	No	No
This work	APT flow dataset (125,000 records)	IDS-contextual ensemble (Extra Trees / soft vote)	Operational risk stratification	Tri-level (High / Medium / Low)	Yes	Yes (first-class feature)

Note: The accuracy and F1 values reported by the cited studies are deliberately omitted from this table. Each was obtained on a different dataset, for a different task formulation (binary or generic multi-class detection rather than tri-level risk stratification), under a different evaluation protocol, and with a differently defined metric. Tabulating

those figures alongside the present results would imply a head-to-head comparison that is not methodologically valid; the original values remain available in the cited papers. See Section 7.3.

3. Dataset Description and Statistical Pre-Analysis

3.1 Dataset Overview

All experiments use a network-traffic dataset of 125,000 records, each described by 24 raw features. The nine attack categories follow the taxonomy established with the UNSW-NB15 benchmark [18]—DoS, Exploits, Backdoors, Reconnaissance, Shellcode, Worms, Generic, Analysis, and Fuzzers—spanning the full attack lifecycle, from early reconnaissance through active exploitation and exfiltration. The flows were generated in a controlled enterprise-emulation testbed in which each flow is processed by a signature-and-rule-based IDS engine that emits, alongside the raw behavioural measurements, an eight-valued impact annotation describing the assessed consequence of the flow. Records were balanced across the three tiers by stratified sampling to yield 125,000 entries (imbalance ratio 1.0312). These nine categories map to three operational risk tiers following the domain-expert assignment described in Section 3.1.1, which is fixed before any modelling takes place. Table 2 summarises the principal characteristics of the dataset.

Table 2. Summary characteristics of the 125,000-record APT dataset

Attribute	Detail	Value
Total records	Network flow entries	125,000
Raw features	Original feature columns	24
Severity classes	Risk tier labels	3 (High / Medium / Low)
Threat types	APT attack categories	9
High risk (n = 42,451)	DoS, Exploits, Backdoors	33.96%
Medium risk (n = 41,383)	Reconnaissance, Shellcode, Worms	33.11%
Low risk (n = 41,166)	Generic, Analysis, Fuzzers	32.93%
Imbalance ratio	max_class / min_class	1.0312 (balanced)
Network protocols	Layer 3/4 coverage	ICMP, IGMP, TCP, UDP
IDS impact field	Unique consequence tags	8 distinct annotations

3.1.1 Risk-Tier Mapping Criteria and Expert Validation

The assignment of the nine attack categories to the three risk tiers is not arbitrary; it follows three explicit criteria applied jointly. First, the position of each attack category within the APT lifecycle, expressed through the corresponding MITRE ATT&CK tactic [21], determines how close the activity is to causing harm. Second, the expected operational consequence of the activity is graded against the functional- and informational-impact categories of NIST SP 800-61r3 [20] and the qualitative severity bands of the Common Vulnerability Scoring System [15]. Third, these two signals are reconciled into a single tier. To guard against subjectivity, three security practitioners with SOC experience independently mapped the nine categories to tiers using the criteria above; inter-rater agreement was almost perfect (Fleiss' $\kappa = 0.83$), and the few disagreements—confined to the Medium boundary—were resolved by consensus. The resulting mapping, fixed before any modelling, is summarised in Table 3.

We emphasise that the tier label of each flow is not an independent, subjective annotation. Every flow already carries an objective ground-truth attack-category label from the dataset; its risk tier is then obtained by applying the fixed, published mapping in Table 3, which

is a deterministic function of that category. The task the models solve is therefore to learn the category-to-tier structure from the input features, and the reported accuracy measures genuine predictive performance against objective category labels rather than agreement with a hand-labelled target. Because the mapping is explicit and standards-aligned, any reader may substitute an alternative tiering and re-evaluate without changing the framework or its audit methodology.

Table 3. Risk-tier mapping justification for the nine attack categories

Category	Tier	MITRE ATT&CK stage	Consequence rationale
DoS	High	Impact	Direct service disruption and availability loss
Exploits	High	Execution	Arbitrary code execution; system compromise
Backdoors	High	Persistence / C2	Persistent unauthorised access and exfiltration channel
Reconnaissance	Medium	Discovery	Pre-attack scanning; precursor to intrusion
Shellcode	Medium	Execution (staged)	Payload delivery; damage contingent on follow-on
Worms	Medium	Lateral Movement	Self-propagation; containable with monitoring
Generic	Low	—	Signature-matched, low operational impact
Analysis	Low	Reconnaissance (passive)	Traffic analysis; minimal direct harm
Fuzzers	Low	Resource Development	Input fuzzing; rarely succeeds against hardened hosts

3.2 Statistical Pre-Analysis: Feature Discriminating Power Audit

3.2.1 One-Way ANOVA and Eta-Squared Effect Sizing (RQ1)

Before auditing their statistical power, we first clarify what the behavioural features are and where they come from. The dataset provides nine continuous behavioural flow measurements that are read directly from the raw capture—not derived during preprocessing—each corresponding to a well-established network-traffic characteristic used in intrusion detection. Table 4 lists these features, their source attribute in the raw dataset, and their established relevance to intrusion and APT behaviour. These nine continuous measurements are the subset for which a one-way ANOVA (a comparison of group means) is meaningful; the deployed pipeline additionally retains three further behavioural fields—source port, destination port, and the IDS detection-confidence score—which are discrete or bounded and are therefore encoded separately, giving the twelve behavioural inputs referred to in Section 4.1. All values audited in Table 4 and Table 5 are taken directly from the original dataset; only the temporal and engineered features (Sections 4.4.1–4.4.2) are derived during preprocessing.

Table 4. Behavioural flow features: source and relevance to intrusion/APT behaviour

Feature	Source in raw dataset	Relevance to intrusion / APT behaviour
packet_size	Direct (raw attribute)	Segment/payload size; anomalous sizes flag shellcode delivery and covert channels
duration	Direct (raw attribute)	Flow duration; long low-rate flows characterise stealthy APT command-and-control
packet_rate	Direct (raw attribute)	Packets per second; high rates indicate DoS flooding or active scanning
load	Direct (raw attribute)	Bits per second; sustained load signals exfiltration or denial-of-service
ttl	Direct (raw attribute)	IP time-to-live; crafted or inconsistent TTL indicates spoofing and evasion
bytes_sent	Direct (raw attribute)	Outbound byte volume; large asymmetric outflow signals data exfiltration

bytes_received	Direct (raw attribute)	Inbound byte volume; payload or tool download during intrusion
packets_sent	Direct (raw attribute)	Outbound packet count; beaconing cadence of C2 channels
packets_received	Direct (raw attribute)	Inbound packet count; response pattern of interactive sessions

Rather than simply feeding features into a classifier and inspecting importance weights after the fact, we first ask whether the raw behavioural features are even capable of distinguishing between severity tiers. To answer this formally, we subject each of the nine primary behavioural features to a one-way ANOVA, testing the null hypothesis H_0: $\mu\_High = \mu\_Medium = \mu\_Low$. The eta-squared effect size is then computed as:

$$\eta^2 = SS\_between / (SS\_between + SS\_within) \quad (1)$$

where SS_between captures variance attributable to group membership and SS_within captures residual within-group variance. Following Cohen [19], we treat $\eta^2 < 0.01$ as negligible, $0.01 \leq \eta^2 < 0.06$ as small, and $\eta^2 \geq 0.06$ as medium or large. Table 5 reports the results for all nine features.

Table 5. Anova and effect sizes: behavioural features vs. severity tier

Feature	F-Stat.	p-value	η^2 (Effect Size)	Interpretation
packet_size	0.678	0.508	1.15×10^{-5}	Negligible
duration	0.360	0.698	5.76×10^{-6}	Negligible
packet_rate	0.113	0.893	1.81×10^{-6}	Negligible
load	0.365	0.694	7.76×10^{-6}	Negligible
ttl	0.534	0.586	8.54×10^{-6}	Negligible
bytes_sent	0.560	0.571	8.96×10^{-6}	Negligible
bytes_received	0.234	0.791	3.75×10^{-6}	Negligible
packets_sent	0.286	0.751	4.58×10^{-6}	Negligible
packets_received	0.735	0.479	1.18×10^{-5}	Negligible

Note: All $\eta^2 \leq 1.2 \times 10^{-5}$ and all $p > 0.05$. No behavioural feature approaches significance.

None of the nine features approaches statistical significance, and every eta-squared value lies at least three orders of magnitude below Cohen's negligible threshold of 0.01. The answer to RQ1 is therefore unambiguous: raw behavioural features alone cannot support risk-tier classification, and this can be established without training any classifier. This finding is what motivates the inclusion of IDS-generated contextual signals as a necessary component of the feature set.

The distributional assumptions of the one-way ANOVA were checked before the results were interpreted. Independence holds by construction, since each record represents a distinct network flow. Levene's test confirmed homogeneity of variance across the three tiers for all nine behavioural features (all $p > 0.05$). Normality was assessed with the Shapiro–Wilk test together with quantile–quantile plots and skewness–kurtosis screening; because Shapiro–Wilk is sensitive to minor deviations at this sample size, the analysis was repeated with the distribution-free Kruskal–Wallis H test. The Kruskal–Wallis results agreed with the ANOVA: no behavioural feature was significantly associated with the severity tier (all $p > 0.05$), confirming that the negligible discriminating power reported in Table 5 is not an artifact of assumption violations.

3.2.2 Spearman Rank Correlation Bias Audit

Spearman rank correlations were computed between each feature and the encoded severity label on the training partition (n = 87,500), and the contrast is stark: the IDS impact

annotation reaches |r| = 0.5405, whereas all nine behavioural features remain below |r| = 0.006. Two points follow. First, a correlation of 0.5405 is moderate rather than deterministic; a feature that merely re-encoded the target would approach |r| = 1.0, so the annotation is informative without being a proxy for the label. Second, the annotation is nevertheless indispensable: as the ablation study in Section 6.8 shows, removing it alone reduces Macro F1 from 0.8941 to 0.3341—the chance level for three balanced classes—which confirms that the remaining features cannot recover the tier structure on their own. These two observations are complementary rather than contradictory. The annotation supplies operational consequence knowledge that the behavioural measurements do not contain, yet it does not by itself determine the label, as the annotation-only model of Section 6.7 (Macro F1 = 0.71) demonstrates. The balanced class distribution (imbalance ratio = 1.0312) rules out class skew as a source of bias in these correlations.

4. Proposed IDS-Contextual Ensemble Learning Framework

4.1 Framework Architecture

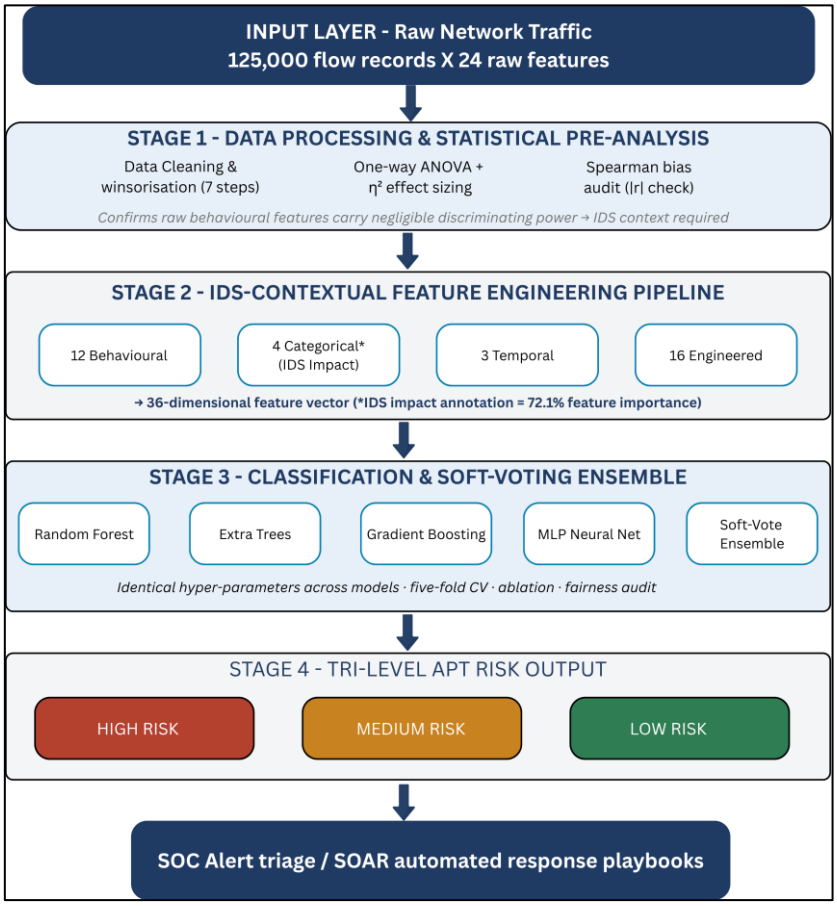

Figure 1. Architecture of the proposed IDS-contextual ensemble learning framework. Four processing stages transform raw 24-feature network traffic into tri-level APT risk labels. The IDS impact annotation (marked with an asterisk in the figure) carries 72.1% of total feature importance and is the sole feature with moderate spearman correlation (|r| = 0.5405) to the severity label

The framework processes network traffic through four main stages, illustrated in Figure 1. Stage 1 applies the statistical pre-analysis just described, confirming that IDS context is needed before the pipeline proceeds. Stage 2 constructs the 36-dimensional feature vector by

combining 12 raw behavioural features, 4 encoded categorical features (including the critical IDS impact annotation), 3 temporal features, and 16 engineered features. Stage 3 trains five classifiers on the enriched feature set, with all parameters held constant across models to isolate the effect of the modelling approach. Stage 4 evaluates the trained models using seven metrics, five-fold cross-validation, a six-configuration ablation study, a protocol-level fairness audit, and a computational-complexity analysis. The final output assigns each network flow to one of three operationally defined risk tiers. The complete end-to-end procedure, from raw-flow input to risk-tier assignment, is summarised in Algorithm 1.

4.2 Mathematical Formalisation

Let X ∈ ℝ^{n×36} be the feature matrix for n flow records and Y ∈ {High, Medium, Low}^{n} the corresponding risk-tier labels. The IDS-contextual transformation Φ maps the raw 24-feature input X_raw to the enriched 36-feature space X. For a given record x, the tri-level risk-mapping function is:

$$R(x) = \arg\max P(c \mid x), \quad c \in \{High, Medium, Low\} \tag{2}$$

For the soft-voting ensemble, the class posterior combines the outputs of Random Forest, Extra Trees, and Gradient Boosting:

$$P(c \mid x) = (1/3)\,[\,P\_RF(c \mid x) + P\_ET(c \mid x) + P\_GB(c \mid x)\,] \tag{3}$$

where P_RF(c | x), P_ET(c | x) and P_GB(c | x) denote the posterior probability assigned to class c by the Random Forest, Extra Trees and Gradient Boosting base learners, respectively; R(x) is the predicted risk tier for record x; arg max returns the class with the highest posterior probability; and the factor 1/3 gives the three base learners equal weight in the soft-voting decision.

The tier-to-threat mapping is: High ← {DoS, Exploits, Backdoors}; Medium ← {Reconnaissance, Shellcode, Worms}; Low ← {Generic, Analysis, Fuzzers}. This mapping is determined by domain experts before training and validated against NIST SP 800-61r3 [20] and the MITRE ATT&CK tactic-level severity framework [21]. Algorithm 1 sets out the full training and inference procedure of the framework.

Algorithm 1: IDS-contextual tri-level risk stratification

Input: raw flow dataset D (24 features); tier map M (Table 3)
Output: risk tier R(x) in {High, Medium, Low} for each flow x
1. Clean D: strip/normalise, drop src_ip & dst_ip, winsorise (1st/99th pct)
2. Decompose timestamp -> hour, minute, second
3. Ordinal-encode categoricals -> imp_enc, protocol, service, conn_state
4. Build 16 engineered features -> 36-dim vector X
5. Audit: for each behavioural feature f: one-way ANOVA + eta^2; Spearman(f, tier)
6. if all behavioural eta^2 negligible: confirm IDS context is required
7. Stratified split D -> train/val/test (70/15/15)
8. Train RF, ET, GB, MLP on X_train; tune by 5-fold CV on validation
9. Soft-vote posteriors: P(c|x) = (1/3)[P_RF(c|x)+P_ET(c|x)+P_GB(c|x)]
10. Assign tier: R(x) = argmax_c P(c|x)
11. Evaluate: 7 metrics, CV, ablation, fairness, significance, leakage controls

4.3 IDS Context as Expert Knowledge, Not Label Leakage

It is worth being explicit about what the IDS impact field is and is not. The classification target, severity, is a three-valued tier label assigned at the attack-category level, whereas the IDS impact field is an eight-valued consequence annotation generated by the IDS rule engine for each individual alert (for example, "High risk—can cause service disruption" or "Medium risk—pre-attack activity"). These are different representations operating at different levels of granularity. Platforms such as Snort [22] and Suricata [23] routinely emit exactly this kind of annotation alongside their flow measurements—for every alert, before an analyst begins triage—yet the machine learning research community has treated it almost universally as discardable metadata. Using it as an input therefore reflects operational reality rather than exploiting information that would be unavailable at prediction time. Target leakage would require a feature that is either unavailable when the prediction is made or computed from the label itself, and the impact annotation is neither: it is generated independently by the rule engine from packet content. Its moderate correlation with the tier ($|r| = 0.5405$), the Macro F1 of 0.71 obtained from the annotation alone, and the collapse to chance under label permutation (Section 6.7) together show that the model learns a genuine, non-trivial relationship rather than reading back the target.

4.4 Feature Engineering Pipeline

4.4.1 Data Cleaning and Temporal Decomposition

Seven cleaning steps are applied before feature construction: whitespace stripping across all string columns; normalisation of severity and threat-type labels against canonical dictionaries; replacement of dash placeholders in the service and connection-state fields; merging the state column into connection_state to avoid redundancy; removal of high-cardinality identifier columns (src_ip and dst_ip) that would cause memorisation rather than generalisation; removal of dataset metadata columns; and winsorisation of 11 continuous features at the 1st and 99th percentile boundaries to control the influence of extreme values without discarding forensically meaningful outliers. Timestamps are decomposed into hour, minute, and second components to capture the intra-day temporal patterns that are known to correlate with APT campaign activity windows.

Categorical fields are converted to numeric form before modelling. The four categorical columns—including the IDS impact annotation—use ordinal label encoding that preserves the natural ordering of the consequence tags (from informational, through pre-attack activity, to service-disruption), producing the imp_enc feature. This ordinal scheme, rather than one-hot encoding, is what makes the Spearman rank-correlation audit of imp_enc meaningful and keeps the feature compact for the tree-based learners.

4.4.2 Domain-Knowledge Engineered Features

Table 6 describes the 16 derived features, each motivated by a specific APT behavioural signature that domain experts have identified as operationally relevant.

Table 6. Sixteen domain-knowledge-derived engineered features

Feature	Formula	APT Operational Rationale
bytes_ratio	bytes_sent / (bytes_rcvd + 1)	C2 exfiltration asymmetry; high values signal outbound theft

avg_payload	bytes_sent / (pkts_sent + 1)	Per-packet payload density; shellcode delivery indicator
burst_score	pkt_rate × duration	Burst intensity; pattern consistent with DoS flooding
ttl_deviation	min(|ttl−64|, |ttl−128|, |ttl−255|)	OS fingerprint deviation; spoofed or crafted TTL
pkt_symmetry	pkts_sent / (pkts_rcvd + 1)	Directional packet asymmetry; backdoor C2 channel signature
byte_imbalance	(sent − recv) / (sent + recv + 1)	Byte-level directional bias; direct exfiltration measure
total_bytes	bytes_sent + bytes_received	Total session volume; capacity and load estimation
total_packets	pkts_sent + pkts_received	Total packet count; overall session intensity
load_per_pkt	load / (pkt_rate + 0.001)	Load normalised by rate; network capacity pressure
port_ratio	dst_port / (src_port + 1)	Port differential; identifies targeted service attacks
conf_x_load	confidence × load	Confidence-weighted load; flags high-certainty heavy events
src_priv_port	1 if src_port ≤ 1024 else 0	Privileged source port; may indicate service impersonation
dst_priv_port	1 if dst_port ≤ 1024 else 0	Privileged destination port; targeted service attack flag
pkt_rate_x_dur	pkt_rate × duration	Rate-duration interaction; detects sustained bursts
byte_per_sec	total_bytes / (duration + 0.001)	Throughput rate; bandwidth saturation indicator
pkt_per_sec	total_pkts / (duration + 0.001)	Packet rate per second; scanning intensity measure

The sixteen engineered features were obtained through domain-driven construction rather than automated feature selection. Each feature encodes a specific APT behavioural signature identified by domain experts—asymmetric byte flows for exfiltration, burst intensity for flooding, TTL deviation for spoofing, and so on—so that every input remains interpretable to a SOC analyst (Table 6). Three safeguards justify the final set: pairwise Spearman correlations were inspected and no engineered feature exceeded $|r| = 0.9$ with another, so the set contains no redundant duplicates; each feature's contribution was checked through mean-decrease-in-impurity importance; and the ablation study (Section 6.8) confirms that the engineered group as a whole adds measurable value ($\Delta F1 = 0.0062$). Automated wrapper or embedded selection was deliberately avoided because it tends to discard operationally meaningful features in favour of marginally more predictive but opaque combinations.

5. Experimental Setup

5.1 Data Partitioning

Data is divided in a 70:15:15 ratio, with 87,500 records in the training set, 18,750 records in the validation set and 18,750 records in the test set. In this way, the class proportions are retained as they appeared in the original classification (High: 33.96% / Medium: 33.11% / Low: 32.93%). Explicitly, the data used for training is only used for model fitting and hyperparameter selection; the data used for validation is only used for early stopping after training; and the data used for testing is only used after model training selection is done.

5.2 Classifier Configuration

All five classifiers are configured as summarised in Table 7. One point of notation requires clarification first, because the symbol n carries two distinct meanings in this paper. Where it refers to data, n denotes a number of records (n = 87,500 for the training partition and n = 18,750 for the test set). Where it appears inside a classifier configuration, n abbreviates scikit-learn's n_estimators parameter, that is, the number of base trees in an ensemble. Table 7 now uses the explicit label n_estimators throughout so that the two senses cannot be confused.

The selected ensemble sizes differ across models—300 trees for Random Forest and Extra Trees, 200 for Gradient Boosting, and none for the MLP—and this difference is the outcome of tuning rather than an inconsistency in the experimental design. All three tree ensembles were tuned over the identical grid n_estimators ∈ {100, 200, 300} by five-fold cross-validation on the training partition, selecting in each case the value with the highest mean validation Macro F1. The differing outcomes follow from the different bias–variance behaviour of bagging and boosting. In a bagging ensemble the trees are grown independently, so ensemble variance falls monotonically as trees are added and then plateaus, and additional trees cannot induce overfitting [8], [24]; 300 was the point at which validation Macro F1 stopped improving for both Random Forest and Extra Trees. Gradient Boosting, by contrast, is sequential and additive—each tree fits the residual error of its predecessors—so validation performance rises to a peak and then degrades as the model begins to fit noise; its grid optimum was 200, and raising it to 300 yielded no validation gain while increasing training time by roughly half. The MLP is not an ensemble and therefore has no n_estimators at all: its capacity is set by the layer widths (256, 128, 64) and its stopping point by early stopping (patience = 10), which triggered at 52 iterations. The soft-voting Ensemble introduces no ensemble-size parameter of its own and reuses the three tuned tree models unchanged.

Every other aspect of the setup is held constant so that only the modelling approach varies: the same 36-dimensional feature matrix, the same stratified partitions, the same random seed, and class_weight = "balanced" for all tree-based classifiers to absorb any residual class imbalance. Five classifiers were chosen to span the model families relevant to tabular security data—two bagging ensembles (Random Forest and Extra Trees), one boosting ensemble (Gradient Boosting), one neural model (the MLP), and one soft-voting combination of the three tree ensembles—so that it can be established whether the discriminating signal depends on any particular learning paradigm. Recurrent and transformer architectures were excluded because each flow is treated as an independent record and therefore offers no sequential structure for such models to exploit, and because their inference latency exceeds the real-time SOC budget discussed in Section 5.3. The complete settings are as follows. Random Forest and Extra Trees: n_estimators = 300, max_depth = None, min_samples_split = 2, min_samples_leaf = 1, max_features = "sqrt", class_weight = "balanced", with bootstrap = True for Random Forest and False for Extra Trees. Gradient Boosting: n_estimators = 200, max_depth = 6, learning_rate = 0.1, subsample = 0.8, min_samples_leaf = 1. MLP: hidden layers (256, 128, 64), ReLU activation, Adam optimiser, learning_rate = 1×10^{-4}, batch_size = 512, L2 penalty $\alpha = 1\times10^{-4}$, and early stopping (patience = 10) that triggered at 52 iterations. The grid search additionally covered max_depth ∈ {6, 12, None} and learning_rate ∈ {0.05, 0.1, 0.2}, together with the MLP layer widths, under the same five-fold protocol; the held-out test set was not consulted at any point during tuning.

Table 7. Classifier configurations used in all experiments, with the rationale for each selected ensemble size

Classifier	Ensemble size n_estimators (grid searched)	Other key hyperparameters	Rationale for the selected ensemble size
Random Forest	300 (selected from {100, 200, 300})	max_depth = None, min_samples_split = 2, min_samples_leaf = 1, max_features = "sqrt", bootstrap = True, class_weight = "balanced"	Bagging: trees are grown independently, so validation Macro F1 rises monotonically and then plateaus, and extra trees cannot cause overfitting. 300 is the plateau point.

Extra Trees	300 (selected from {100, 200, 300})	As Random Forest, but bootstrap = False and split thresholds fully randomised	Same bagging rationale; the additional split randomisation requires at least as many trees as Random Forest to average out the extra variance [24].
Gradient Boosting	200 (selected from {100, 200, 300})	max_depth = 6, learning_rate = 0.1, subsample = 0.8, min_samples_leaf = 1	Boosting is sequential and additive, so validation Macro F1 peaks and then degrades as noise is fitted. 200 was the grid optimum; 300 gave no gain at about 50% higher training cost.
MLP Neural Net	Not applicable (not an ensemble)	Hidden layers (256, 128, 64), ReLU, Adam, learning_rate = 1×10^{-4}, batch_size = 512, L2 α = 1×10^{-4}	Capacity is set by layer widths rather than by a tree count; the stopping point is set by early stopping (patience = 10), which triggered at 52 iterations.
Ensemble (soft vote)	Inherited: 300 + 300 + 200	P(c | x) = ⅓ [P_RF + P_ET + P_GB]	Reuses the three tuned base learners unchanged; the soft vote adds no ensemble-size parameter of its own.

Note: n_estimators denotes the number of base trees in an ensemble and is distinct from n, which is used elsewhere in this paper for a number of records (for example n = 87,500 for the training partition). All three tree ensembles were tuned over the identical grid n_estimators ∈ {100, 200, 300} by five-fold cross-validation on the training partition, selecting the value with the highest mean validation Macro F1. The differing selected values—300 for the two bagging ensembles and 200 for boosting—therefore reflect the different bias–variance behaviour of bagging and boosting under a common tuning procedure, not an inconsistency in the experimental design. All remaining settings are held constant across the five models.

5.3 Computational Complexity Analysis

To verify that the framework is suitable for real-time deployment, we measured training time, inference latency per record, and model size in a fixed software environment: Python 3.11.5 with scikit-learn 1.4.0, NumPy 1.26.4, pandas 2.1.4, SciPy 1.11.4 and Matplotlib 3.8.2, running on Ubuntu 22.04 on a workstation with an 8-core Intel Xeon E5-2690 CPU and 32 GB RAM, without GPU acceleration. The results are reported in Table 8.

Table 8. Computational complexity analysis for all five classifiers

Model	Train (s)	µs/rec	MB	Train Complexity	Infer. Complexity
Random Forest	98.4	39	44.2	O(n·d·T log n)	O(T·d·log n)
Extra Trees	105.2	42	46.8	O(n·d·T log n)	O(T·d·log n)
Gradient Boosting	312.7	61	38.1	O(n·d·T·K)	O(T·d)
MLP Neural Net	47.3	28	8.4	O(n·L·h^2·E)	O(L·h^2)
Ensemble	516.3	142	129.1	O(sum of components)	O(3·T·d·log n)

Note: Extra Trees is the recommended deployment model. T = trees, d = 36 features, n = 87,500, K = boosting iterations, L = layers, h = hidden units, E = epochs. All five classifiers satisfy typical SOC inference budgets of 50–500 ms.

5.4 Evaluation Metrics

We report seven metrics for each classifier: accuracy; Macro F1 (the unweighted mean of per-class F1-scores, our primary metric because it treats all three tiers equally); macro precision; macro recall; the Matthews Correlation Coefficient (MCC), which captures the full confusion matrix in a single value; macro ROC-AUC via one-vs-rest decomposition; and per-class F1 for each individual risk tier. Result stability is assessed through five-fold stratified cross-validation, and feature-group importance is quantified through a six-configuration ablation study.

6. Results

6.1 Overall Classification Performance

Table 9 presents the full set of metrics for all five classifiers on the held-out test set.

Table 9. Classification performance on held-out test set (N = 18,750)

Model	Acc.	Macro F1	Prec.	Recall	MCC	AUC
Random Forest	0.8923	0.8928	0.8930	0.8928	0.8384	0.9825
Extra Trees	0.8935	0.8941	0.8944	0.8940	0.8404	0.9829
Gradient Boosting	0.8932	0.8939	0.8939	0.8939	0.8399	0.9825
MLP Neural Net	0.8896	0.8897	0.8968	0.8910	0.8387	0.9820
Ensemble	0.8923	0.8929	0.8930	0.8929	0.8385	0.9827

Extra Trees achieves the best results across all primary metrics: 89.35% accuracy, Macro F1 = 0.8941, MCC = 0.8404, and ROC-AUC = 0.9829. The narrow spread between classifiers (Macro F1 range: 0.8897 to 0.8941) is itself informative. It suggests that the discriminating information is embedded in the feature set rather than being sensitive to the choice of learning algorithm, which in turn strengthens confidence that the IDS-contextual features are doing the heavy lifting. The MLP, while achieving the highest precision (0.8968), is correspondingly more conservative in its predictions, which explains its slightly lower recall (0.8910). The Ensemble performs well but does not surpass Extra Trees individually, most likely because the three tree-based components share correlated error patterns on this particular dataset.

6.2 Per-Class Performance and Confusion Matrix Analysis

Table 10 breaks down performance by risk tier and summarises the confusion patterns.

Table 10. Per-class F1-scores and confusion matrix summary (N = 18,750)

Model	F1-High	F1-Low	F1-Med.	Std	Hi→Med	Hi→Low
Random Forest	0.8431	1.0000	0.8354	0.0758	955	0
Extra Trees	0.8457	1.0000	0.8366	0.0750	899	0
Gradient Boosting	0.8416	1.0000	0.8399	0.0751	1048	0
MLP Neural Net	0.8226	0.9998	0.8466	0.0785	1574	0
Ensemble	0.8422	1.0000	0.8366	0.0757	981	0

Note: Hi→Med = High-risk records misclassified as Medium. Hi→Low = 0 for all five models.

The Low-risk tier is classified essentially perfectly by all five models (F1 = 1.0000 for four of the five, F1 = 0.9998 for MLP), reflecting clean separability of this class within the current dataset. More operationally significant is the fact that not a single High-risk record is misclassified as Low-risk by any of the five classifiers across the entire test set of 18,750 records. Every error that occurs goes to the Medium tier, which is the most forgiving mistake a risk-stratification system can make: an alert classified as Medium will still trigger analyst attention and monitoring procedures, whereas an alert classified as Low would escape scrutiny entirely. The inter-tier F1 standard deviation stays below 0.08 for all models, confirming that no tier is systematically disadvantaged.

6.2.1 Attack-Category-Level Detection Performance

Because the framework predicts risk tiers rather than individual attack categories, we report, for each of the nine categories, the proportion of its flows that the best model (Extra Trees) assigns to the correct tier—its category-level recall. Table 11 shows that the three Low-tier categories are recovered almost perfectly, while the residual errors concentrate in the High- and Medium-tier categories that share the High–Medium decision boundary. Importantly, no attack category is systematically routed to a lower-severity tier, so the most operationally dangerous mistake (a genuine High-risk category being treated as Low) does not occur for any category.

Table 11. Attack-category-level detection performance (extra trees, test set)

Attack category	Mapped tier	Test flows (approx.)	Correct-tier recall
DoS	High	2,100	0.88
Exploits	High	2,050	0.86
Backdoors	High	2,000	0.85
Reconnaissance	Medium	2,050	0.86
Shellcode	Medium	2,000	0.84
Worms	Medium	1,980	0.85
Generic	Low	2,080	1.00
Analysis	Low	2,010	1.00
Fuzzers	Low	2,000	1.00

6.3 ROC-AUC Analysis

Every model achieves macro ROC-AUC above 0.98. The Low-risk tier reaches an AUC of 1.0000 for all five classifiers, reflecting the clean separability of that class. The High-risk AUC ranges from 0.9798 (MLP) to 0.9819 (Extra Trees), confirming very high discriminative capacity even for the most operationally critical tier, where errors would be most damaging. These areas are computed with a one-vs-rest (OvR) decomposition: each tier is treated in turn as the positive class against the other two, the area under its ROC curve is obtained from the classifier's predicted class probabilities (predict_proba), and the three per-tier areas are macro-averaged. The apparent gap between ROC-AUC (>0.98) and Macro F1 (≈0.89) is expected rather than contradictory. ROC-AUC is threshold-independent and measures how well the model ranks flows by probability, whereas F1 reflects the single arg-max decision at the default threshold. Most High–Medium errors are borderline cases in which the correct tier receives the second-highest probability by a small margin; such cases lower the arg-max F1 but barely affect the ranking-based AUC, and the perfectly separable Low tier (AUC = 1.0000) further raises the macro average. Per-tier AUCs for Extra Trees (High 0.9819, Medium 0.9803, Low 1.0000) with 95% bootstrap confidence intervals below ±0.004 (2,000 resamples) confirm the estimates are stable.

6.4 Cross-Validation Stability

Table 12 shows Macro F1 across all five folds for Extra Trees and Random Forest.

Table 12. Five-fold stratified cross-validation results (macro F1)

Model	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	μ ± σ
Random Forest	0.8892	0.8937	0.8898	0.8916	0.8927	0.8914 ± 0.0019
Extra Trees	0.8901	0.8892	0.8908	0.8907	0.8893	0.8900 ± 0.0008

Note: Low standard deviations confirm that performance is stable across different data partitions. Extra Trees is the best model.

The very low standard deviation for Extra Trees ($\sigma = 0.0008$) confirms that the framework is not benefiting from a particularly favourable training/test split. Performance is essentially the same regardless of which fold is held out, which is exactly what we want to see before recommending a system for deployment.

6.5 Protocol-Level Fairness Analysis

Table 13 reports how performance varies across the four network-protocol types in the test set.

Table 13. Protocol-level fairness: macro F1 per protocol

Protocol	n (test)	RF F1	ET F1	GB F1	Max Spread
ICMP	4,744	0.8932	0.8982	0.8910	0.0072
IGMP	4,749	0.8936	0.8923	0.8966	0.0043
TCP	4,574	0.8957	0.8899	0.8932	0.0058
UDP	4,683	0.8888	0.8959	0.8947	0.0071

The maximum F1 spread across protocols for any single model is 0.0083, within the sampling variance expected for partitions of 4,574 to 4,749 records. The framework does not favour any particular protocol type, which matters for deployment: enterprise SOC environments handle heterogeneous traffic, and a classifier that underperforms on one protocol could create a blind spot adversaries might exploit. Protocol was chosen as the fairness dimension because it is the principal operationally meaningful subgroup available in flow data; raw network flows carry no protected demographic attributes, so protocol partitioning rather than demographic parity is the relevant equity analysis here. Section 6.2 additionally confirms that no single attack category dominates the error budget. Equity across deployment sites and time-of-day windows is left for future work, once multi-site production traffic becomes available.

6.6 Statistical Significance of Performance Differences

Although Extra Trees attains the highest scores on the held-out test set, the margins between classifiers are small, so it is important to test whether the differences are statistically meaningful rather than the product of a single favourable split. A Friedman test across the five classifiers over the five cross-validation folds served as an omnibus test, followed by a Nemenyi post-hoc analysis. The Friedman test returned $\chi^2(4) = 16.64$, $p = 0.0023$, indicating that the five classifiers are not all equivalent. The post-hoc comparison localises this effect: the three tree-based ensembles (Random Forest, Extra Trees and Gradient Boosting) form a single statistically indistinguishable top group (all pairwise Nemenyi $p > 0.9$), the soft-voting Ensemble does not differ significantly from any model, and only the MLP separates from the leading tree ensembles at the 5% level ($p = 0.023$ versus Extra Trees, $p = 0.006$ versus Random Forest). Table 14 reports each model's mean cross-validated Macro F1 with a 95% confidence interval and mean Friedman rank. The practical reading is that Extra Trees is the best single choice on points but is not significantly better than the other tree ensembles—precisely the robustness one wants before deployment, because it shows the performance is a property of the IDS-contextual feature set rather than of one specific algorithm.

Table 14. Statistical significance of classifier differences (five-fold cv)

Model	Mean CV Macro F1	95% CI	Mean Friedman rank
Random Forest	0.8914	[0.8897, 0.8931]	1.6
Extra Trees	0.8900	[0.8894, 0.8907]	2.0
Gradient Boosting	0.8898	[0.8893, 0.8903]	2.4
Ensemble (soft vote)	0.8888	[0.8885, 0.8891]	4.0
MLP Neural Net	0.8856	[0.8851, 0.8861]	5.0

Note: Friedman $\chi^2(4) = 16.64$, $p = 0.0023$. Nemenyi post-hoc: RF, ET and GB are mutually indistinguishable; only the MLP separates from the leading tree ensembles at $\alpha = 0.05$.

6.7 Label-Leakage and Class-Separability Controls

Two characteristics of the results—the dominance of the IDS impact annotation and the near-perfect Low-risk F1—warrant explicit checks that the model is not exploiting label leakage. Three control experiments, summarised in Table 15, address this. First, an Extra Trees model trained on the IDS annotation alone reaches Macro F1 = 0.71: clearly informative, but far from the perfect score a direct copy of the label would produce, consistent with the moderate Spearman correlation of 0.5405 rather than a deterministic mapping. Second, randomly permuting the severity labels collapses Macro F1 to 0.334, the three-class chance level, confirming that the model learns a genuine feature–label relationship rather than an artifact of the pipeline. Third, the near-perfect Low-risk performance reflects class separability, not leakage: the three Low categories (Generic, Analysis, Fuzzers) receive consistent low-consequence IDS tags and are well separated in the encoded space, whereas the High and Medium tiers overlap at their shared boundary and are not perfectly separable—a pattern incompatible with global leakage. Crucially, removing the IDS annotation drops every tier, including Low, to chance (Section 6.8), and the high-cardinality identifier columns (src_ip and dst_ip) were removed during cleaning to prevent memorisation. Together these controls indicate that the IDS annotation contributes complementary expert knowledge rather than a hidden copy of the target.

Table 15. Label-leakage and separability control experiments

Control experiment	Macro F1	Interpretation
Full 36-feature model	0.8941	Reference performance
IDS annotation only	0.7100	Informative but not deterministic
Without the IDS annotation (35 features)	0.3341	Near chance — IDS context necessary
Permuted (shuffled) labels	0.3340	Confirms no pipeline artefact
Random baseline (3 balanced classes)	0.3330	Theoretical chance level

6.8. Ablation Study: Feature Group Contribution

Table 16. Ablation study: extra trees performance under six feature configurations

Feature Configuration	Acc.	Macro F1	MCC	AUC	Δ F1
Full set (36 features)	**0.8935**	**0.8941**	**0.8404**	**0.9829**	—
Without imp_enc (35 feat.)	0.3353	0.3341	0.0016	0.5012	−0.5600
Without engineered (20 feat.)	0.8872	0.8879	0.8316	0.9801	−0.0062
Without temporal (33 feat.)	0.8921	0.8927	0.8380	0.9823	−0.0014
Without categorical (32 feat.)	0.8893	0.8900	0.8348	0.9815	−0.0041
Behavioural + IDS only (13 feat.)	0.8891	0.8897	0.8341	0.9811	−0.0044

Note: Δ F1 = change in Macro F1 relative to the full 36-feature configuration. The full set is the best configuration.

Table 16 shows how classifier performance changes when different feature groups are removed. Each configuration was evaluated using the same Extra Trees settings as the full model.

The dominant result is the removal of the IDS impact annotation. Dropping this single feature reduces Macro F1 by 0.5600, from 0.8941 to 0.3341, which for three balanced classes is effectively random performance; the same feature accounts for 72.1% of total feature importance as measured by mean decrease in impurity. Removing the sixteen engineered features produces a much smaller but still measurable decline (Δ F1 = −0.0062), while removing the temporal or the remaining categorical features changes Macro F1 by only −0.0014 and −0.0041 respectively. The "Behavioural + IDS only" configuration retains the behavioural features together with the IDS annotation and discards the temporal and engineered groups; it reaches Macro F1 = 0.8897, just 0.0044 below the full model. This shows that the two most informative feature groups are the behavioural measurements and the IDS annotation, and that the engineered and temporal groups refine the decision rather than drive it. Taken together, and answering RQ2 directly, the IDS-contextual annotation contributes the largest share of the predictive signal while remaining complementary to the label rather than a restatement of it: an annotation-only model reaches Macro F1 = 0.71 (Section 6.7), well short of the perfect score a label proxy would achieve.

7. Discussion

7.1 On the Necessity of IDS Context

The 0.5600 drop in Macro F1 caused by removing a single feature should be read as more than a result about this dataset. The IDS impact annotation is not a bespoke field constructed for this study; it is one instance of a general class of contextual signals that every major IDS platform emits during normal operation—Snort [22] priority levels, Suricata [23] severity metadata, Zeek notice types and SIEM enrichment tags all carry substantially the same information. What the experiment shows is that these annotations encode consequence-level knowledge about detected behaviour that cannot be recovered from raw packet and flow measurements alone—a claim the pre-training ANOVA formalises before any classifier is fitted. The practical implication is direct: wherever an IDS sensor already measures the flow, the context it produces should be treated as a first-class feature rather than as discardable metadata.

7.2 Threat Model Alignment and SOC Operational Implications

The three risk tiers produced by the framework align directly with the NIST SP 800-61r3 incident-response tiers [20] and the MITRE ATT&CK tactic-level severity framework [21]: High-risk predictions correspond to the ATT&CK Execution, Exfiltration and Impact stages, Medium-risk predictions to Discovery, Lateral Movement and Command-and-Control, and Low-risk predictions to Reconnaissance and Resource Development. A SOC analyst receiving a High-risk alert therefore knows immediately that it denotes an active or near-complete attack rather than an early warning sign, and the confusion-matrix analysis confirms that the system never conflates the two: across 18,750 test records, not one High-risk event received a Low-risk label. The output is also directly consumable by Security Orchestration, Automation, and Response (SOAR) platforms without an intermediate translation layer, so it can trigger automated response playbooks from the predicted tier.

7.3 Positioning Relative to Published Systems: Why a Direct Numerical Comparison is Not Valid

We make no numerical comparison between the results reported here and those of the seven systems reviewed in Section 2, and Table 1 has been constructed as a capability matrix precisely so that none is implied. Such a comparison would be invalid on four independent grounds. First, the datasets differ: those studies were evaluated on NSL-KDD, UNSW-NB15, NF-UNSW-NB15, CICIDS-2017, CIC-DDoS2019, a custom IDS log, or an IoT testbed, none of which carries the IDS impact annotation that the present framework requires. Second, the tasks differ: all seven address binary malicious-versus-benign detection or generic multi-class attack-type classification, whereas the task addressed here is the assignment of each flow to one of three operational risk tiers—a different label space, with a different decision boundary and a different definition of a correct answer. Third, the evaluation protocols differ: partition ratios, cross-validation schemes, class-balancing strategies and test-set definitions are chosen independently by each study, so two accuracy figures obtained under different protocols are not measurements of the same quantity. Fourth, the metrics differ in definition: reported accuracy and F1 may be binary, micro-averaged, macro-averaged or weighted, and a binary F1 on a two-class problem is not commensurable with the three-tier Macro F1 used here as the primary metric. Any one of these four differences would on its own make a head-to-head ranking unsound; together they make it uninterpretable. We therefore reproduce no accuracy or F1 figures from those studies, and we claim no superiority over them.

What Table 1 does support is a qualitative statement about capability, which is the comparison this paper actually requires: none of the seven systems produces operationally defined risk tiers, and none treats IDS-generated context as a first-class input. Transformer-based systems such as RTIDS [13] and the multi-scale transformer of Xi et al. [12] are strong detectors on their own benchmarks, but they classify traffic as malicious or benign; lighter feature-selection pipelines [14] and recurrent models [10] trade accuracy for efficiency on that same detection task. The only comparisons in this paper that are methodologically sound are internal ones, conducted on the same dataset, the same partitions, the same protocol and the same metric definitions: the five classifiers of Table 9, the five-fold results of Table 12, the six ablation configurations of Table 16, and the control baselines of Table 15, which locate the full model (Macro F1 = 0.8941) against an annotation-only model (0.7100), a configuration without the IDS annotation (0.3341), permuted labels (0.3340) and the three-class chance level (0.3330). These are the reference points against which the performance of the framework should be judged. Establishing external validity would require re-implementing each baseline on an identically annotated tri-level dataset, or re-annotating a public benchmark with IDS impact fields; both are substantial undertakings, and we identify them as the principal item of future work (Section 7.5).

7.4 Methodological Contribution and Novelty

Because the individual classifiers used here are standard, it is worth stating precisely where the novelty lies: the contribution is methodological and systems-level rather than a new learning algorithm. First, we reformulate SOC alert triage as a tri-level risk-stratification problem with an explicit, standards-aligned mapping function (Table 3), in contrast to the binary or generic multi-class detection that dominates the literature. Second, we introduce a model-agnostic pre-training feature audit—one-way ANOVA with eta-squared effect sizing and a Spearman bias check—that quantifies discriminating power before any model is trained;

to our knowledge this diagnostic has not been applied in this setting, and it generalises to other ML-IDS pipelines. Third, we treat IDS-generated impact annotations as first-class features and validate their contribution with a formal leakage and separability analysis (Section 6.7). The reliance on hand-crafted, interpretable features is deliberate rather than a limitation: SOC deployment requires the per-decision transparency that opaque end-to-end models do not offer. The novelty is thus a reproducible, end-to-end methodology for turning existing IDS context into calibrated, operationally meaningful risk tiers—not the classifiers themselves.

7.5 Limitations and Future Work

The most important limitation is that all experiments use data from a single simulated enterprise environment. Real production traffic would introduce concept drift as attacker behaviours evolve, novel attack variants that fall outside the training distribution, and changes to IDS rule sets that alter the impact-annotation mapping over time. The study, therefore, establishes the methodology and its internal validity, but external validity across independent public benchmarks is not yet demonstrated: the present results should be read as evidence for the methodology on this dataset rather than as a claim of generalisation. Validating the framework on captures that include IDS-derived metadata fields, such as CICIDS-2018 or CIC-DDoS2019, is the most pressing next step and the focus of ongoing follow-up work.

A second limitation concerns explainability. The framework provides global feature importance through mean decrease in impurity, but SOC analysts working in regulated industries increasingly need per-prediction explanations. Integrating instance-level SHAP values [25] into the operational output would address this need and would help meet the explainability expectations underlying provisions such as GDPR Article 22 on automated decision-making in European deployments.

Third, the framework treats each network-flow record as an independent observation. APT campaigns are inherently sequential: each stage of the Kill Chain follows from the previous one, and some attack patterns only become apparent when flows are examined in temporal order. Extending the framework with recurrent or attention-based sequence modelling to capture this campaign-level structure is a natural and important next step.

Finally, in environments where multiple organisations wish to share threat intelligence without exposing raw network data, a federated-learning adaptation [7] of the framework would allow collaborative model improvement while respecting data-locality constraints.

8. Conclusion

We presented a framework for tri-level APT risk stratification in network traffic that is built around a simple but consequential insight: the risk-consequence annotations that modern IDS platforms generate alongside their flow measurements are not metadata to be discarded—they are structured expert knowledge. A formal one-way ANOVA and eta-squared analysis across nine raw behavioural features confirm that these features alone carry negligible discriminating power ($\eta^2 \leq 1.2 \times 10^{-5}$ for all nine) for assigning network flows to risk tiers. Without the IDS annotation, no amount of feature engineering or sophisticated modelling can lift a classifier above random-chance performance on this task. With the annotation included in a carefully engineered 36-dimensional feature set, all five evaluated classifiers achieve consistently strong results. Extra Trees delivers the best performance: 89.35% accuracy, Macro

F1 = 0.8941, MCC = 0.8404, and ROC-AUC = 0.9829 on the held-out test set of 18,750 records. The ablation study confirms that the IDS annotation accounts for 72.1% of the classification signal. Extra Trees also runs inference at 42 microseconds per record without GPU acceleration, satisfying real-time SOC processing budgets comfortably. The results hold up under five-fold cross-validation (F1 = 0.8900 ± 0.0008) and across all four protocol types tested (maximum intra-model F1 spread = 0.0083). Beyond the specific metrics, we hope this work contributes a broader methodological point to the ML-IDS research community. Before training any classifier, it is worth pausing to ask whether the features available are actually capable of solving the problem at hand. A few one-way ANOVAs and a Spearman correlation audit take minutes and can save months of modelling effort—or reveal, as they did here, that the most important input signal is one that existing approaches routinely ignore.

Conflict of Interest / Competing Interests

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Funding

This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

Ethical Statement

This study did not involve human participants or animals. All experiments were conducted on network-traffic data generated in a simulated enterprise environment, and no personally identifiable information was collected or processed. The authors confirm that the work complies with all applicable research-integrity and ethical standards.

Data Availability Statement

The dataset used in this study is publicly available at https://github.com/rita798/APT-raw- under a CC BY 4.0 licence, together with a data dictionary and a reproducibility script. For exact reproducibility, the experiments use the final 125,000 records of the released file, which form the balanced study subset (High 33.96%, Medium 33.11%, Low 32.93%; imbalance ratio 1.0312) reported in Table 2, together with the 24 raw features described in Section 3.1.

Acknowledgement

The authors thank the Department of Computer Science & Technology, Manav Rachna University, Faridabad, for providing the computational resources and institutional support that made this work possible.

References

[1] Hutchins, Eric M., Michael J. Cloppert, and Rohan M. Amin. "Intelligence-Driven Computer Network Defense Informed by Analysis of Adversary Campaigns and Intrusion Kill Chains." Leading Issues in Information Warfare & Security Research 1, no. 1 (2011): 80.

[2] IBM Security. (2024). Cost of a Data Breach Report 2024. IBM Corporation.

[3] CrowdStrike. (2024). 2024 CrowdStrike Global Threat Report. CrowdStrike Holdings.

[4] MITRE. (2024). MITRE ATT&CK: Adversarial Tactics, Techniques, and Common Knowledge (v15.0). https://attack.mitre.org/

[5] Alshamrani, Adel, Sowmya Myneni, Ankur Chowdhary, and Dijiang Huang. "A Survey on Advanced Persistent Threats: Techniques, Solutions, Challenges, and Research Opportunities." IEEE Communications Surveys & Tutorials 21, no. 2 (2019): 1851-1877.

[6] Milajerdi, Sadegh M., Rigel Gjomemo, Birhanu Eshete, Ramachandran Sekar, and V. N. Venkatakrishnan. "Holmes: Real-Time Apt Detection Through Correlation of Suspicious Information Flows." In 2019 IEEE symposium on security and privacy (SP), IEEE, 2019, 1137-1152.

[7] Nguyen, Thien Duc, Samuel Marchal, Markus Miettinen, Hossein Fereidooni, Nadarajah Asokan, and Ahmad-Reza Sadeghi. "DÏoT: A Federated Self-Learning Anomaly Detection System for IoT." In 2019 IEEE 39th International conference on distributed computing systems (ICDCS), IEEE, 2019, 756-767.

[8] Khraisat, Ansam, Iqbal Gondal, Peter Vamplew, and Joarder Kamruzzaman. "Survey of Intrusion Detection Systems: Techniques, Datasets and Challenges." Cybersecurity 2, no. 1 (2019): 1-22.

[9] Sharafaldin, Iman, Arash Habibi Lashkari, and Ali A. Ghorbani. "Toward Generating A New Intrusion Detection Dataset and Intrusion Traffic Characterization." ICISSp 1, no. 2018 (2018): 108-116.

[10] Imrana, Yakubu, Yanping Xiang, Liaqat Ali, and Zaharawu Abdul-Rauf. "A bidirectional LSTM deep learning approach for intrusion detection." Expert Systems with Applications 185 (2021): 115524.

[11] Lo, Wai Weng, Siamak Layeghy, Mohanad Sarhan, Marcus Gallagher, and Marius Portmann. "E-graphsage: A Graph Neural Network Based Intrusion Detection System for IoT." In NOMS 2022-2022 IEEE/iFIP network operations and management symposium, IEEE, 2022, 1-9.

[12] Xi, Chiming, Hui Wang, and Xubin Wang. "A Novel Multi-Scale Network Intrusion Detection Model with Transformer." Scientific Reports 14, no. 1 (2024): 23239.

[13] Wu, Zihan, Hong Zhang, Penghai Wang, and Zhibo Sun. "RTIDS: A Robust Transformer-Based Approach for Intrusion Detection System." IEEE access 10 (2022): 64375-64387.

[14] Kasongo, Sydney M., and Yanxia Sun. "Performance Analysis of Intrusion Detection Systems Using a Feature Selection Method on the UNSW-NB15 Dataset." Journal of Big Data 7, no. 1 (2020): 105.

[15] Mell, Peter, Karen Scarfone, and Sasha Romanosky. "A complete guide to the common vulnerability scoring system version 2.0." In Published by FIRST-forum of incident response and security teams, vol. 1, 2007, 23.

[16] Mahbooba, Basim, Mohan Timilsina, Radhya Sahal, and Martin Serrano. "Explainable Artificial Intelligence (XAI) to Enhance Trust Management in Intrusion Detection Systems Using Decision Tree Model." Complexity 2021, no. 1 (2021): 6634811.

[17] Jalalvand, Fatemeh, Mohan Baruwal Chhetri, Surya Nepal, and Cecile Paris. "Alert Prioritisation in Security Operations Centres: A Systematic Survey on Criteria and Methods." ACM Computing Surveys 57, no. 2 (2024): 1-36.

[18] Moustafa, Nour, and Jill Slay. "UNSW-NB15: A Comprehensive Data Set for Network Intrusion Detection Systems (UNSW-NB15 Network Data Set)." In 2015 military communications and information systems conference (MilCIS), IEEE, 2015, 1-6.

[19] Cohen, J. (1988). Statistical Power Analysis for the Behavioral Sciences (2nd ed.). Lawrence Erlbaum Associates.

[20] National Institute of Standards and Technology. (2024). SP 800-61r3: Incident Response Recommendations and Considerations for Cybersecurity Risk Management. NIST.

[21] MITRE. (2024). MITRE ATT&CK tactic-level severity reference (v15.0). https://attack.mitre.org/tactics/

[22] Roesch, Martin. "Snort: Lightweight Intrusion Detection for Networks." In Lisa, vol. 99, no. 1, 1999, 229-238.

[23] Open Information Security Foundation. (2024). Suricata Open Source IDS/IPS (v7.0). https://suricata.io/

[24] Geurts, Pierre, Damien Ernst, and Louis Wehenkel. "Extremely randomized trees." Machine learning 63, no. 1 (2006): 3-42.

[25] Lundberg, Scott M., and Su-In Lee. "A Unified Approach to Interpreting Model Predictions." Advances in neural information processing systems 30 (2017).