AI Based Threat Detection System In Network

Saquib Azam Khan
Dept. of Information Science & Engg.
CMR Institute of Technology
Bengaluru, India
saquib.ise24le@cmrit.ac.in

Shahid Farooq Malik
Dept. of Information Science & Engg.
CMR Institute of Technology
Bengaluru, India
shma23ise@cmrit.ac.in

Shaikh Aslam Sami
Dept. of Information Science & Engg.
CMR Institute of Technology
Bengaluru, India
shas23ise@cmrit.ac.in

Dr. Jagadishwari V
Dept. of Information Science & Engg.
CMR Institute of Technology
Bengaluru, India
jagadishwari.v@cmrit.ac.in

Abstract—Network intrusion detection must adapt to changing traffic while preserving recognition of previously observed attacks. This paper studies the trade-off between retraining frequency, labelled sample size, and detection performance, and designs an evidence-guided security operations centre (SOC) workflow around the detector. A 100-tree Random Forest is evaluated on six constructed traffic windows with increasing proportions of DDoS and PortScan attacks. Across five paired random seeds, an output-based adaptive policy achieves 92.008% pooled incoming attack recall, compared with 92.302% for retraining every two windows, while reducing mean retraining counts from 3.0 to 1.8. No drift alarms occur in the tested stable controls. However, exact-feature grouped evaluation exposes a concentrated retention failure: one seed loses detection of 883 previously recognised old attack rows after an update. A separate validation-gate experiment accepts every candidate and therefore does not establish a protective benefit. A small FTP capture further shows that successful feature conversion does not guarantee attack recognition: all 30 flows are classified benign despite evidence of repeated failed logins. The proposed architecture connects detection with packet investigation, indicator extraction, timelines, evidence-grounded analysis, ATT&CK mapping, and analyst-approved response. Results support a limited offline cost–recall trade-off, while identifying retention, labelling realism, and extractor compatibility as unresolved requirements.

Index Terms—Network intrusion detection; adaptive retraining; distribution shift; Random Forest; security operations centre; network forensics.

I. INTRODUCTION

A useful network intrusion detection system must do more than classify a static test set. It must recognise attacks as traffic changes, decide when an update is worthwhile, and provide evidence that an analyst can investigate. These requirements create competing objectives. Frequent retraining may improve recognition of newly introduced traffic, but consumes computation and labelled examples. Infrequent updates reduce that cost while leaving a detector exposed to unfamiliar patterns. An update can also alter decisions on familiar attacks, even when the training set retains old examples.

Prior research provides substantial foundations for these problems. Deep ResNIDS combines classification, anomaly detection, novelty detection, and transfer learning, while identifying retraining timing and sample size as further research questions [1]. SOC research proposes architectures that integrate detection, intelligence, and response [2]. Recent incident-analysis systems already combine log retrieval, language models, and ATT&CK information [3], [4]. The research opportunity is therefore not the invention of adaptive detection or an integrated SOC. It is the careful evaluation of a specific, reproducible retraining policy and its requirements within an evidence-preserving operational workflow.

This study asks four questions. RQ1: How much incoming attack recall is retained when an adaptive trigger replaces fixed retraining schedules? RQ2: How does the amount of newly labelled attack data affect recall and update frequency? RQ3: Do stable controls, duplicate-aware splits, and a candidate acceptance check support the reliability of the observed behaviour? RQ4: What additional engineering and evidence are needed to connect the offline detector to packet-based incident analysis?

The work contributes an offline comparison of five update policies across five paired seeds; a distinction between fixed-schedule and freely adaptive sample-size experiments; a duplicate audit and grouped retention diagnosis; and a full SOC architecture with explicit implementation boundaries. A small PCAP case study supplies an additional engineering observation, rather than an accuracy benchmark. The central contribution is the combined account of the cost–recall trade-off and its failure conditions, not a claim of universally optimal retraining.

Scope is deliberately limited. The traffic sequence is a constructed mixture of benchmark records, not a timestamp-preserving live stream. The monitoring decision does not use attack labels, but retraining uses immediately available, correct labels and samples known attack rows. The broader investigation and response architecture is proposed; it has not been evaluated as an integrated deployed SOC. These distinctions are necessary to interpret both the positive results and the remaining work.

II. RELATED WORK AND RESEARCH POSITION

A. Adaptive detection and model maintenance

Hore et al. [1] develop a sequential deep learning framework with a malicious-packet classifier and two autoencoders. Their one-shot transfer-learning experiments address adaptation to several types of unfamiliar traffic. Their conclusion explicitly leaves the timing and sample size of retraining open to further investigation. Our study examines those policy choices with a simpler flow-based Random Forest. It does not reproduce their representation, datasets, or neural architecture, and therefore does not claim superior detection performance. Its comparison is between policies applied to the same local experimental setup.

Pingili et al. [5] discuss integrating machine learning with Suricata and an ELK dashboard. Such work motivates the practical connection between detection and analyst-facing monitoring. However, a dashboard integration and a controlled comparison of retraining schedules answer different questions. We focus on the latter while specifying the evidence interfaces required for the former. Retraining

count, incoming recall, and old-attack retention are treated as separate outcomes, rather than reducing the comparison to classification accuracy.

B. SOC integration and forensic investigation

Khayat et al. [2] review AI and ML applications in SOCs and propose a reference architecture. Chourasiya et al. [6] describe log analysis using LSTM and transformer methods with forensic correlation. These studies show that detection, correlation, and incident reporting have established research foundations. They motivate our modular workflow, but their results cannot validate modules that are only designed in the present project. In particular, a flow classifier cannot establish the sequence of attacker actions without additional packet or log evidence.

Ferazza et al. [4] combine event reconstruction, summarisation, clustering, and MITRE-enriched meta-alerts. Their discussion also identifies dependence on summarisation quality and the need for analyst oversight. Cadet et al. [3] evaluate retrieval-augmented incident analysis using targeted queries over security logs and forensic questions. These are direct precedents for evidence-guided analysis and attack reconstruction. Consequently, our proposed timeline and language-model stages are integration objectives, not novel capabilities already demonstrated by this work.

C. Indicators, threat intelligence, and ATT&CK

Liao et al. [7] introduce iACE for extracting indicators and their context from technical threat reports. The distinction between an observable token and a contextualised indicator is especially relevant: an IP address in a flow is not sufficient evidence that the address is malicious. Mishra et al. [8] combine BERT, spaCy, and other components for threat-intelligence extraction and prediction. The supplied version is an article in press; it is used here for architectural context, not as a directly comparable performance baseline.

Orbinato et al. [9] study mapping unstructured threat intelligence to ATT&CK and show that evaluation on full documents differs from classification of short descriptions. Kuppa et al. [10] map vulnerabilities to ATT&CK using joint embeddings, while Abdeen et al. [11] use semantic relationships in CVE descriptions. These tasks require different evidence from network-flow classification. A binary attack prediction alone cannot justify a specific ATT&CK technique. Jiang et al. [12] survey applications of ATT&CK, supporting its role as a shared vocabulary rather than a substitute for incident evidence.

Tellache et al. [13] combine retrieval over threat-intelligence documents with external intelligence searches for incident response. Their approach motivates keeping retrieved knowledge and local observations linked but distinct. A report may explain how a technique usually operates without proving that it occurred in the incident under investigation. Our design therefore requires source references, explicit uncertainty, and analyst approval before operational action.

TABLE I. POSITION RELATIVE TO PRIOR WORK

Research strand	Established contribution	Focus of this study
Adaptive NIDS [1]	Sequential detection and transfer learning	Update timing and sample-size trade-offs
SOC integration [2], [5], [6]	Reference architecture, monitoring, log analysis	Evidence interfaces and implementation boundaries
Incident analysis [3], [4], [13]	Retrieval, correlation and reconstruction	Proposed connection to adaptive detection
IOC / ATT&CK [7] –[12]	Contextual extraction and semantic mapping	Reviewed, evidence-linked outputs; no new result claimed

D. Bounded research gap

The gap addressed empirically is a local policy question: whether a label-independent output trigger can approach a fixed update schedule with fewer refits under a specified attack-mixture shift. The experiments also investigate how that trade-off changes with sample size and duplicate-aware evaluation. They do not establish a general solution to concept drift, autonomous incident response, or zero-day detection. The full architecture connects these experiments to existing investigation methods, with unimplemented components stated explicitly.

III. SYSTEM ARCHITECTURE

A. Design principle and evidence flow

NIDS-AI-SOC is organised around two connected paths. The detection path converts traffic into features, produces model outputs, monitors changes, and evaluates model updates. The investigation path retains the underlying evidence and enriches alerts into an analyst-reviewable incident record. Figure 1 distinguishes work exercised in the project from proposed integration. Batch PCAP processing and the offline adaptation experiments are separate implementations; the diagram does not imply that they already form a continuously operating service.

The architecture uses a common event envelope. Each record should contain an event identifier, original timestamp and timezone information, source and destination endpoints, protocol, capture reference, feature-schema version, detector version, and evidence provenance. Predictions, rule matches, external intelligence, and analyst decisions should be separate fields. This prevents a model label from being silently promoted into an observed fact and permits a report to trace a statement back to the relevant packet or log record.

B. Capture, conversion, and detection

The capture layer accepts an authorised PCAP or, in a future implementation, a live interface. Raw evidence should remain immutable with a digest and capture metadata. A feature adapter aligns extractor output with the classifier schema while preserving the original flow metadata outside the model input. It must check names, order, units, missing values, and finite numeric values. Matching the number of

columns is necessary but insufficient: two extractors can assign different meanings to similarly named fields.

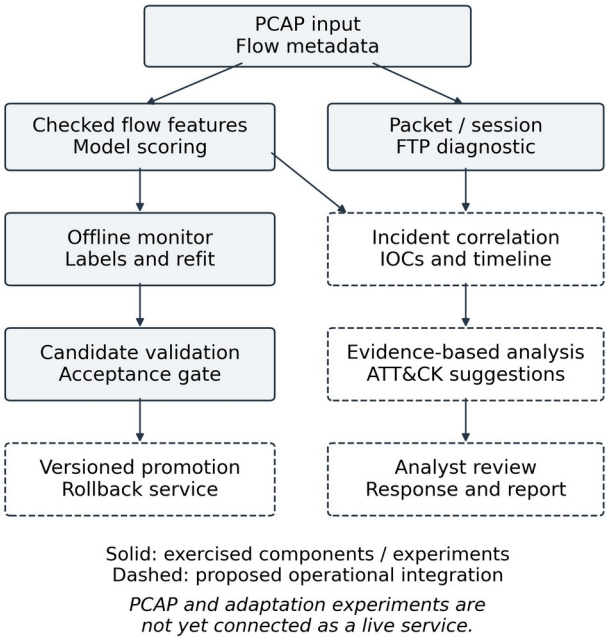

Fig. 1. Architecture and implementation boundaries. Arrows show intended data relationships; solid blocks are not a claim of end-to-end live integration.

The detection layer produces a binary prediction and a model attack score. A protocol or behavioural rule can independently produce an alert, retaining its own rule identifier and supporting events. This separation is important when a model misses a behaviour that is visible in protocol evidence. An alert identifies something to investigate; it does not prove compromise. Scores should not be presented as calibrated probabilities unless calibration has been measured on appropriate held-out data.

C. Monitoring and model lifecycle

The monitoring layer aggregates the active model's outputs over a traffic window and compares them with outputs on a reference sample. A trigger requests a retraining attempt; it does not directly establish a malicious traffic change. In the offline experiments, acquired labelled attacks are accumulated with old training data and a new Random Forest is fitted. For deployment, the proposed lifecycle adds versioned training manifests, a validation gate, an auditable promotion decision, and the ability to restore an earlier model.

Candidate evaluation and final testing have distinct roles. A validation set may guide acceptance, whereas a final test set should assess performance without influencing promotion. Our acceptance experiment implements this separation within its simulation, but it does not demonstrate that the validation set covers every old attack mode. A deployed system would also need label-quality checks, delayed-feedback handling, monitoring after promotion, and an operational definition of when rollback is justified.

D. Investigation, indicators, and timeline

Investigation should resolve an alert into its associated bidirectional flows, packet evidence, and relevant host or authentication logs. The prototype performs a narrow version of this step by reconstructing FTP control sessions and counting commands and replies without printing credentials. A larger system would keep the identifiers of the supporting events, report capture gaps, and distinguish direct observation from inferred relationships.

Indicator extraction would identify candidate IP addresses, domains, URLs, file hashes, and other observable values. Context must explain where a value was found and why it matters. Private addresses from a local laboratory should remain incident observables rather than being treated as globally malicious indicators. Enrichment may add reputation or threat-report references, but unavailable intelligence is not evidence that an observable is benign.

The timeline stage would order events using their recorded times while retaining timezone, clock-quality, and source information. Correlated events could be grouped into a session or incident, but temporal proximity alone would not establish causation. A useful timeline should distinguish a directly observed failed login, a detector alert, and an analyst hypothesis. This separation is required before evaluating higher-level attack reconstruction.

E. Analysis, ATT&CK, response, and reporting

The proposed AI analysis stage retrieves relevant local evidence and curated threat knowledge to produce a concise incident explanation. Every factual claim should link to evidence identifiers. Logs and retrieved documents are untrusted input, so embedded instructions must not control the analysis or response system. Unsupported claims should be omitted or marked uncertain, and the system should be able to report that the available evidence is insufficient.

ATT&CK mapping would suggest techniques from observed behaviours and supporting context, using a recorded knowledge-base version. The output should include the proposed technique, evidence, and uncertainty for analyst review. A flow labelled ATTACK cannot by itself justify a claim about credential theft, lateral movement, or an adversary's objective. Response is therefore specified as a recommendation and approval stage, not automatic blocking. The report should preserve observations, predictions, inferences, review decisions, and actions as separate categories.

TABLE II. IMPLEMENTATION AND EVALUATION STATUS

Component	Evidence in this project
Adaptive ML experiments	Implemented; five-seed offline comparisons
Retention and gate	Grouped diagnosis; all gate candidates accepted
PCAP feature adapter	30 flows checked; semantics not fully validated
FTP investigation	606 packets and 30 sessions inspected
Behavioural login rule	Prepared; no completed evaluation reported
IOC, timeline, AI, ATT&CK	Proposed integrated stages
Response and reporting service	Proposed; analyst approval required
Continuous live integration	Not demonstrated

IV. EXPERIMENTAL METHODOLOGY

A. Data and construction of the stream

Experiments use the MachineLearningCVE flow CSV files associated with CICIDS2017 [14]. Column names and textual labels are normalised; BENIGN is mapped to zero and other labels to one. Infinite values are converted to missing values and incomplete rows are removed. The resulting inputs contain 78 numeric features. Tuesday, Wednesday, and Thursday-morning WebAttacks files provide the old-data pool of 1,307,282 cleaned rows. Friday DDoS and PortScan files contain 225,711 and 286,096 cleaned rows respectively; attack-labelled rows from these sources form the introduced attack pool.

For the original row-based experiment, 100,000 old rows are reserved for training, 20,000 for old testing, and 10,000 separate rows for the monitoring reference. Six windows contain 10,000 rows each. Introduced attack shares are 5%, 10%, 20%, 30%, 40%, and 50%, giving 15,500 attack rows over the sequence. The remaining 44,500 rows are genuine benign records sampled from unused old data. Source-row partitions are disjoint and stream rows are used once, although identical feature vectors can still occur in different partitions.

These windows deliberately change the mixture of known dataset attack families. The available CSVs lack usable timestamp columns for reconstructing a chronological sequence in this setup. Consequently, the experiment represents a controlled distribution shift, not verified natural temporal drift. It cannot distinguish changes in the feature distribution from changes in the conditional relationship between features and labels, and it does not demonstrate recognition of genuinely novel attack mechanisms.

TABLE III. CORE EXPERIMENTAL SETTINGS

Setting	Value
Old train / test / reference	100,000 / 20,000 / 10,000 rows
Windows	6 × 10,000 rows
Introduced attack shares	5, 10, 20, 30, 40, 50%
Total introduced attacks	15,500
Model	100-tree balanced Random Forest
Seeds	42, 43, 44, 45, 46
Default labelled sample	5% of true attacks at each update
Refit memory	Old training + accumulated labels
Primary metric	Pooled recall before each update

B. Classifier and policy comparison

The experimental classifier is a Random Forest with 100 trees, balanced class weights, a fixed seed for each paired run, and parallel fitting. The same initial model, partitions, windows, and potential label samples are used for the compared policies within a seed. The policies are no retraining, retraining every one, two, or three windows, and the adaptive trigger. Five seeds, 42 through 46, vary sampling and model randomisation. Their outcomes are paired experimental replications, not five independent operational environments.

When a policy retrains, it acquires a specified fraction of the true attack rows in that window. The default is 5%. These labelled rows are appended to all original training rows and the accumulated labelled attack memory. A fresh forest is fitted; this is full retraining, not incremental tree updating or partial_fit. Old data are retained to support recognition of familiar traffic, but retention of training rows does not guarantee retention of every test decision.

The attack-only acquisition procedure is an oracle assumption. It avoids confusing label acquisition with label uncertainty, but presupposes that the relevant attacks can be identified for sampling. A real analyst would inspect a mixture of benign and malicious candidates, incur review cost, and sometimes deliver labels after a delay. Our labelled-row counts therefore quantify the simulation's acquired attack examples, not total operational annotation effort.

C. Output-based monitoring signals

Let p(x) be the active model's attack score. Three signals compare scores on the current window with scores on the fixed reference rows. The first is a population stability index (PSI) computed using reference-quantile bins. For reference and current bin fractions r_j and c_j, respectively, the implemented form is:

$$PSI = \Sigma_j (c_j - r_j) \ln(c_j / r_j). \quad (1)$$

Zero bin fractions are replaced by a small value, 10^{-6}, and outer bin boundaries extend to infinity. Repeated quantiles reduce the number of distinct edges; when fewer than three edges remain, the implementation returns zero. This fallback matters when scores concentrate at a few values. A zero PSI

can reflect an uninformative bin construction rather than proof that traffic is unchanged.

The second signal is the L1 distance between normalised score histograms with 20 equally spaced bins between zero and one:

$$D = \Sigma_j |c_j - r_j|. \quad (2)$$

This measures a change in the distribution of scores, not simply a difference in the fraction of records classified as attacks. The third signal compares mean prediction confidence, defined here as $\max(p, 1 - p)$:

$$C = |\text{mean}_{\text{window}} \max(p, 1 - p) - \text{mean}_{\text{reference}} \max(p, 1 - p)|. \quad (3)$$

Signal thresholds are PSI ≥ 0.10, D ≥ 0.10, and C ≥ 0.015. The adaptive policy requests retraining when at least two signals occur in two consecutive windows, or when D ≥ 0.14 and C ≥ 0.02 occur together. The latter condition is called a safety trigger in the scripts; it is a stronger monitoring rule, not a demonstrated safety guarantee. All thresholds are exploratory and are not claimed to be optimal.

After an adaptive update, the same reference rows are rescored with the new model and the consecutive counter is reset. This prevents direct comparison of outputs produced by different active models. The fixed policies retain their scheduled update decisions independently of the signals. Thus, monitoring values may differ between policies without changing the fixed schedules. Labels are excluded from the trigger calculation, but remain essential to the simulated retraining step.

D. Metrics and evaluation order

The primary outcome is pooled attack recall on arriving traffic, measured before each window's possible retraining. If TP_t is the number of detected attacks and A_t is the number of attacks in window t, then:

$$R_{in} = 100 \times (\Sigma_t TP_t) / (\Sigma_t A_t). \quad (4)$$

Pooling gives more weight to windows containing more attacks. It differs from an unweighted average of six window recalls, so the two summaries must not be interchanged. For example, the original seed-42 adaptive run has a mean window recall of 75.198% but pooled recall of 92.058%. The larger later windows explain this difference; neither number should be substituted for the other without identifying the metric.

Old-attack recall and benign false-positive rate (FPR) measure retention and false alerts on old test data. FPR is the percentage of benign rows predicted as attacks. A drift-event rate instead measures how often the monitoring policy raises a trigger. Retraining count, elapsed fitting time, and acquired attack labels describe different costs. Timings are descriptive measurements from the development machine; the record does not establish comparable throughput or hardware-normalised performance.

After-retraining scores use rows withheld from the current label sample. However, exact-feature duplicates may remain, and changing sample shares changes the held-out subset. These scores are secondary diagnostics. Incoming recall, recorded before access to current labels, is the more direct measure of detection exposure under the simulated policy. All reported means and ranges are descriptive; no statistical significance or confidence-interval claim is made from five seeds.

E. Controls and sensitivity experiments

A stable control uses six windows with a fixed mixture of familiar attacks and benign traffic, maintaining approximately 20.48% attacks and source/label composition. Very rare groups rounded to zero at this window size are excluded. A frozen monitor and the adaptive controller use unchanged thresholds. This tests unwanted triggering in a narrow stationary construction, not all realistic benign changes such as software upgrades, working-hour effects, or new services.

A duplicate audit compares feature fingerprints across train and test partitions and within the introduced stream. A follow-up assigns exact-feature groups together when splitting the selected old-data pool. This removes measured exact-feature overlap between old training and old testing, but does not enforce host, session, near-duplicate, or temporal independence. Fingerprint matching is also not a mathematical guarantee against hash collisions. The grouped split changes row counts and class composition and is interpreted as a sensitivity analysis.

Finally, a candidate acceptance experiment reserves 10% of grouped training fingerprints for validation and retains the grouped old test partition for reporting. A candidate must keep validation recall within 0.5 percentage points of both the original anchor and the incumbent, and validation FPR within 0.05 points of both. Frozen, ungated, and gated policies share the reduced training partition. Every attempted refit and acquired label is counted, including a rejected candidate's cost. The old test arrays are excluded from the policy simulation and scored afterwards.

V. RESULTS

A. Adaptive versus fixed retraining

TABLE IV. FIVE-SEED POLICY COMPARISON

Policy	Recall mean (%)	Min–max (%)	Refits mean
No retraining	28.298	27.690–28.800	0.0
Every 1 window	96.826	96.316–97.271	6.0
Every 2 windows	92.302	92.000–92.794	3.0
Every 3 windows	83.124	82.406–83.542	2.0
Adaptive	92.008	91.490–92.652	1.8

Table IV answers RQ1 for the original row-based setup. Without retraining, introduced-attack recall remains near 28.3%. Updating every window obtains the highest mean incoming recall, 96.826%, but performs six refits. The adaptive policy obtains 92.008% with 1.8 mean refits, compared with 92.302% and three refits for the every-two-window policy. The paired recall differences are negative in all five seeds, ranging from −0.142 to −0.510 percentage points.

Using the reported means, adaptive retraining reduces update count by 40% relative to every two windows, with a recall difference of −0.294 points. Relative to every window, it reduces update count by 70% while losing 4.818 points of incoming recall. These are explicit trade-offs, not evidence that the adaptive policy dominates both recall and cost. Retraining every three windows has similar mean update count but substantially lower incoming recall, 83.124%, in this constructed sequence.

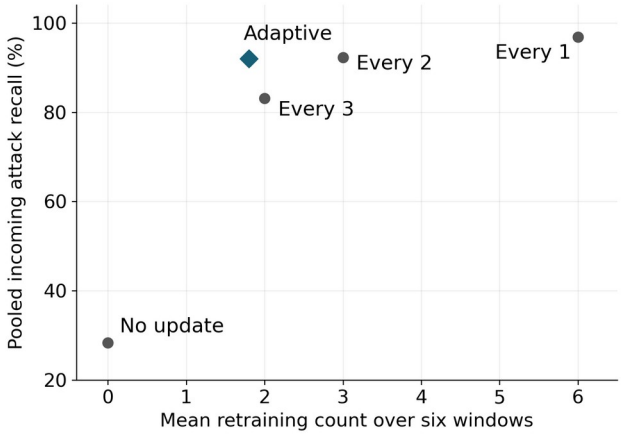

Fig. 2. Incoming recall versus update count, averaged over five seeds in the original row-based experiment. The plot shows a cost–recall trade-off, not statistical significance.

The stable controls report zero drift events and zero retraining operations for both policies over five seeds and six windows per seed. This supports the claim that the chosen thresholds did not trigger on those sampled stable mixtures. It does not imply a zero operational false-alarm rate: the windows are few, generated from one benchmark environment, and share a controlled composition. The score-bin degeneracy described above further limits interpretation of individual PSI values.

B. Sample size and update timing interact

TABLE V. FREELY ADAPTIVE SAMPLE-SIZE RESULTS

Attack sample (%)	Incoming recall (%)	Refits mean	Attack labels mean
1	89.858	3.2	80
5	92.008	1.8	240
10	92.120	1.8	500
25	92.591	2.0	1,500

Table V reports the experiment in which the trigger is allowed to respond separately for each sample share. At 1%, the policy acquires only 80 attack labels on average, but retrains 3.2 times and reaches 89.858% incoming recall. At 5%, it uses 240 labels, retrains 1.8 times, and reaches 92.008%. Increasing to 25% uses 1,500 labels and reaches 92.591%, an improvement of 0.583 points over 5% in this setting. The measured cost in labels therefore rises much more than recall over that comparison.

Sample size also changes future scores and trigger decisions, so the freely adaptive comparison is not solely a test of training-set size. For this reason, a separate follow-up reuses a fixed retraining schedule. Its incoming recalls for 1%, 5%, 10%, and 25% are 80.201%, 92.008%, 92.262%, and 92.615%, respectively. The difference between the two 1% results shows that additional update opportunities can partly compensate for smaller samples. It does not establish that either schedule is universally preferable.

The freely adaptive runs record mean fitting times of 6.180, 3.467, 3.583, and 4.151 seconds for the four sample shares. These small measurements are sensitive to machine load and implementation details. The number of acquired labels and refits is therefore more interpretable than a claim of precise runtime savings. Five percent is a practical candidate within the tested setup, not an optimal sample fraction for live traffic or a known reduction in analyst working time.

C. Duplicate leakage and grouped retention

The audit finds a mean old-test exact-feature match rate of 6.834% against training, with a range of 6.590–7.135%. Mean recall is 99.977% among matching old attack rows and 99.819% among non-matching old attack rows. The introduced attack rows have no exact matches to the initial old training set, but the stream contains a mean of 415.2 duplicate rows and 326.6 rows repeating earlier windows. These observations show why disjoint row identifiers alone are insufficient for independence.

Grouping the old split removes measured exact-feature overlap and changes initial old-attack recall only slightly: 99.854% for the original split and 99.818% for the grouped split. Initial introduced-attack recall is similarly close, 28.298% and 28.310%. This supports the stability of the initial detection weakness under that particular correction. It does not establish that the behaviour of later updates is equally stable.

TABLE VI. GROUPED OLD-HOLDOUT POLICY RESULTS

Policy	Incoming recall (%)	Old recall (%)	Old FPR (%)
No retraining	28.310	99.818	0.061
Every 1 window	96.810	96.571	0.066
Every 2 windows	92.248	97.872	0.064
Every 3 windows	82.968	99.169	0.065
Adaptive	92.133	97.230	0.061

Table VI reveals the important difference after retraining. The grouped adaptive policy preserves similar incoming recall, 92.133%, but its mean old recall falls to 97.230%, compared with 99.838% in the original row-based evaluation. Updating every window gives grouped old recall of 96.571%. These values average over seeds and windows; inspecting individual runs is necessary to identify the source of the deterioration.

Seed 44 accounts for the concentrated adaptive failure. Old-attack recall falls from 99.978% before its cycle-2 update to 80.601%, remains there through cycle 5, and returns to 99.978% after cycle 6. Of 4,557 old test attacks, 883 change from correctly detected to missed, with no offsetting newly detected attacks at cycle 2. Those 883 rows occupy only three

exact-feature groups, containing 560, 322, and one row respectively.

Source-label inspection identifies the attack rows in these groups as DoS Hulk. Their model attack scores move from 0.51, 0.53, and 0.52 before the update to 0.44, 0.46, and 0.46 afterwards. One corresponding feature group also occurs once with a BENIGN label in the full cleaned source. The failure is therefore consistent with a few heavily repeated, near-boundary feature patterns changing class after refitting. It should not be described as broad forgetting of all old attacks or as a proven causal effect of label conflict alone.

D. What the acceptance gate demonstrates

The acceptance experiment changes the available training data by reserving a grouped validation partition. Its outcomes are consequently compared within that experiment, rather than directly against the earlier grouped run. Gated and ungated adaptive policies both achieve 91.920% mean incoming recall and 99.180% mean old-test recall. Each averages two attempts and two acceptances, with zero rejections. Across five seeds, all ten candidates pass the gate.

Seed 44 is especially instructive. Its initial old-test recall is already 80.579%, before retraining, whereas anchor validation recall is 99.721%. The cycle-2 candidate reaches 99.777% validation recall and 99.978% old-test recall; cycle 6 also passes. Thus, the worst old-test value in this run is an initial-model weakness, not damage caused by an accepted update. The gate cannot be credited with repairing the previous experiment's failure, and its ability to reject a harmful candidate remains untested.

The combined result answers RQ3 cautiously. Stable controls support a limited no-trigger observation, duplicate-aware evaluation exposes an important retention sensitivity, and validation/test separation is implemented for candidate decisions. However, a reliable retention guarantee is not established. Validation representativeness is itself a research issue, particularly when a few large feature groups dominate the test outcome and cannot be split across partitions without reintroducing overlap.

VI. PCAP-TO-PREDICTION CASE STUDY

A. Feature compatibility checks

A separate prototype processes a local FTP PCAP using Python CICFlowMeter 0.5.0 and a saved 78-feature Random Forest. This saved classifier is not the adaptive experimental model: the provided training script uses a stratified 80/20 split of a processed dataset, 100 trees, and no explicit balanced class weighting. The offline policy results therefore cannot be attributed to this saved model or used as evidence of its PCAP performance.

Inspection of the installed extractor shows time values expressed in seconds. The checked converter scales 23 duration, inter-arrival, active, and idle fields to microseconds while leaving rate fields unchanged. It enforces the saved model's feature order, treats missing columns as errors, and records invalid rows. The duplicated forward-header field is checked against 200,000 training rows before being supplied from its corresponding value. All 30 extracted flows pass the conversion checks.

Remaining semantic differences include duplicated or simplified feature calculations in the extractor, such as its CWR-related field and whole-flow values assigned to subflow features. Therefore, passing conversion proves only that a defined adapter produced numerically valid, correctly ordered inputs. It does not prove equivalence with the extractor used to construct the training dataset. Testing both extractors on the same packets is needed before asserting such equivalence.

B. Model output and protocol evidence

TABLE VII. FTP CASE: OUTPUTS AND OBSERVATIONS

Measure	Observed count
Packets / control sessions	606 / 30
Valid model input flows	30 of 30
Model benign / suspicious	30 / 0
USER / PASS commands	30 / 30
FTP 331 / 530 replies	30 / 30
FTP 230 successful login replies	0
Reported capture gaps	0

The classifier predicts BENIGN for all 30 flows. The displayed first ten model attack scores range from 0.35 to 0.37; the available output does not establish the score range for all flows. Independent packet inspection reads 606 packets and reconstructs 30 FTP control sessions from one client to one server. It observes 30 USER commands, 30 PASS commands, 30 password-required replies, 30 not-logged-in replies, and no successful-login replies. The diagnostic reports no capture gaps or ambiguous directions under its checks.

Repeated unsuccessful authentication is consistent with attempted password guessing, but these counts are not independently verified attack labels for every flow. The filename is also not ground truth. Accordingly, this case does not support a numerical false-negative rate or a claim of proven live brute-force detection. It demonstrates a disagreement between benign model outputs and protocol evidence that warrants investigation. Credentials are omitted from diagnostic output.

A repeated-login rule has been prepared as a separate detector, but no completed rule evaluation is reported here. Threshold lowering is not justified solely to make this capture produce alerts: doing so could increase false positives without resolving feature incompatibility. The next empirical requirement is a labelled capture collection with benign authentication failures and controlled malicious scenarios, processed consistently from packets through feature extraction and scoring.

VII. DISCUSSION AND THREATS TO VALIDITY

A. What has and has not been answered

RQ1 is answered for the specified offline stream: adaptive updates approach the every-two-window schedule with fewer refits, at a small but consistent recall cost across the five

seeds. RQ2 shows diminishing recall gains beyond the tested 5% sample share and an interaction between sample size and trigger timing. RQ3 remains only partly answered because the grouped analysis exposes retention fragility and no candidate is rejected by the validation gate. RQ4 yields a design and a limited PCAP feasibility check, not an integrated operational evaluation.

A label-independent trigger should not be confused with label-free learning. The detector's outputs can indicate change without labels, but the subsequent improvement depends on correct attack labels arriving immediately. Furthermore, output monitoring is model-dependent: a detector that confidently assigns the same score to unfamiliar attacks may fail to signal a meaningful change. No output-based threshold can be assumed to detect every damaging shift merely because it responds to this constructed mixture.

B. Internal and external validity

The experiment uses one benchmark family and two introduced attack families. Random seeds alter partitions and model realisations, but do not supply independent organisations, network conditions, or attack campaigns. The attack fractions and policy thresholds were investigated within the same project, so final performance should be confirmed on a separate, untouched evaluation set. This work reports exploratory evidence rather than a preregistered confirmatory comparison.

Duplicate grouping addresses exact-feature overlap in the old-data split, not all contamination routes. Near duplicates, related sessions, common hosts, and repeated introduced patterns may remain. Follow-up diagnosis repeatedly examines the old test set; although the gate's code excludes it from promotion, this retrospective development history means it is not a fully untouched final benchmark. The retention findings are valuable diagnostics, but should guide a new validation design rather than selective reporting of favourable seeds.

Changes in split composition affect interpretation. Grouped training and test sets have slightly different row counts and class proportions from their row-based counterparts. Reserving validation groups further changes the initial model. These interventions can alter which influential patterns are seen during training. The seed-44 gate result illustrates why before-update baselines must be inspected rather than interpreting every low old-test recall as retraining damage.

The six-window horizon is short. It does not evaluate sustained adaptation, bounded memory, recurring attacks after long absences, seasonal benign variation, or resource contention during capture and fitting. A policy that performs fewer fits here may behave differently over hundreds of windows. Similarly, zero alarms in the stable controls should not be extrapolated to a deployment-wide false-trigger guarantee.

C. Reproducibility and reporting boundaries

The method description is based on the project's experiment scripts and recorded run outputs. The tabulated values are the supplied aggregate and per-seed results; this manuscript does not represent a fresh rerun of the benchmark data. The source scripts identify seeds, sampling, policies, and result directories. A submission-ready archive should additionally fix the repository commit, dependency versions, raw-data provenance and checksums, processed split manifests, and all per-cycle and per-seed CSV outputs.

The recorded fitting durations are not accompanied by a complete hardware and background-load description, so no general runtime or energy claim is made. Confidence intervals, hypothesis tests, new precision values, and unreported metrics are not reconstructed from rounded summaries. A statistical follow-up should retain paired per-seed results and use independent scenarios as well as more seeds; additional repetitions of one constructed stream cannot alone resolve external validity.

The literature positions the contribution rather than supplying a cross-paper leaderboard. Prior studies use different representations, datasets, labels, and tasks. Their accuracy, IOC extraction, or ATT&CK mapping figures are not directly comparable to pooled incoming attack recall here. In particular, the full SOC workflow is not claimed as the first integration of detection and investigation, given the substantial precedents in [2]–[4], [7], and [13].

D. Operational safeguards and evaluation priorities

The proposed evidence chain is intended to reduce unsupported interpretation. Model predictions, observed protocol events, extracted values, and language-model explanations should retain distinct provenance. Human approval is required for response actions because an erroneous detection or explanation could disrupt legitimate traffic. Sensitive packet content and authentication data require controlled access and appropriate retention. The present prototype demonstrates credential omission, but not a complete production privacy or access-control system.

The next evaluation should first establish consistent packet-to-feature semantics on labelled captures. It should then test retraining with delayed, imperfect labels and a realistic candidate-selection process. Retention checks should include class-specific outcomes and coverage of important feature groups on validation data, without moving final-test groups into training. A gate stress test should include genuinely degraded candidates and measure both harmful acceptance and unnecessary rejection on independent outcomes.

The broader SOC stages need their own ground truth. Indicator extraction requires precision and recall by observable type and context. Timelines require known event order and incident membership. AI analysis requires unsupported-claim rates and evidence-link correctness. ATT&CK suggestions require technique-level evaluation against expert annotations. Response assessment should measure recommendation correctness, analyst overrides, and adverse actions. End-to-end usefulness would then require a controlled analyst study measuring investigation time and decision quality.

VIII. CONCLUSION

This study evaluates adaptive retraining as a practical cost–recall choice within a proposed evidence-guided SOC architecture. Across five paired offline runs, the adaptive trigger obtains 92.008% pooled incoming attack recall with 1.8 mean refits, compared with 92.302% and three refits for updating every two windows. Sample-size experiments support a useful 5% operating point within the tested oracle-labelling setup, while showing that update timing and sample size interact.

The reliability checks materially qualify that finding. Exact-feature grouping reveals a large old-attack recall drop concentrated in three repeated patterns for one seed. The acceptance gate does not yet demonstrate protection because it rejects no candidates, and a separate FTP case shows that valid feature conversion can coexist with benign model outputs on suspicious protocol activity. The contribution is therefore an empirical trade-off, a diagnosis of its limits, and an architecture for further integration. Robust live adaptation and evidence-grounded incident response remain objectives to be tested, not completed outcomes.

REFERENCES

[1] S. Hore, J. Ghadermazi, A. Shah, and N. D. Bastian, “A sequential deep learning framework for a robust and resilient network intrusion detection system,” Computers & Security, vol. 144, Art. no. 103928, 2024, doi: 10.1016/j.cose.2024.103928.

[2] M. Khayat, E. Barka, M. A. Serhani, F. Sallabi, K. Shuaib, and H. M. Khater, “Empowering security operation center with artificial intelligence and machine learning—A systematic literature review,” IEEE Access, vol. 13, pp. 19162–19197, 2025, doi: 10.1109/ACCESS.2025.3532951.

[3] X. Cadet et al., “Retrieval-augmented LLMs for security incident analysis,” in Proc. ACM Conf. AI and Agentic Systems (CAIS), 2026, doi: 10.1145/3786335.3813136.

[4] F. Ferazza, C. Melella, K. Mersinas, R. Lugo, and R. Ottis, “From logs to tactics: Unsupervised reconstruction of APT campaigns with MITRE-enriched meta-alerts,” International Journal of Information Security, vol. 25, Art. no. 79, 2026, doi: 10.1007/s10207-026-01254-w.

[5] M. Pingili, G. S. Sravanthi, K. Srujan Raju, V. Rajesh, A. Deepika, and P. Patel, “A novel approach of enhancing and developing IDS to use ML for analyzing malicious attacks,” in Innovations in Information and Decision Sciences, Smart Innovation, Systems and Technologies, vol. 422, Springer, 2025, ch. 31, doi: 10.1007/978-981-96-0147-9_31.

[6] L. Chourasiya et al., “Advanced system log analyzer for anomaly detection and cyber forensic investigations using LSTM and transformer networks,” Journal of Cloud Computing, vol. 14, Art. no. 60, 2025, doi: 10.1186/s13677-025-00789-y.

[7] X. Liao, K. Yuan, X. Wang, Z. Li, L. Xing, and R. Beyah, “Acing the IOC game: Toward automatic discovery and analysis of open-source cyber threat intelligence,” in Proc. ACM SIGSAC Conf. Computer and Communications Security (CCS), 2016, doi: 10.1145/2976749.2978315.

[8] S. Mishra, R. A. Alfahidah, and F. Alharbi, “BERT-spaCy hybrid NLP and blockchain-enhanced adaptive CTI for IOC extraction and threat prediction,” Scientific Reports, article in press, supplied unedited manuscript, 2026, doi: 10.1038/s41598-025-34505-2.

[9] V. Orbinato, M. Barbaraci, R. Natella, and D. Cotroneo, “Automatic mapping of unstructured cyber threat intelligence: An experimental study,” arXiv:2208.12144v1, 2022.

[10] A. Kuppa, L. Aouad, and N.-A. Le-Khac, “Linking CVE’s to MITRE ATT&CK techniques,” in Proc. 16th Int. Conf. Availability, Reliability and Security (ARES), 2021, doi: 10.1145/3465481.3465758.

[11] B. Abdeen, E. Al-Shaer, A. Singhal, L. Khan, and K. Hamlen, “SMET: Semantic mapping of CVE to ATT&CK and its application to cybersecurity,” in Proc. 37th IFIP Conf. Data and Applications Security and Privacy (DBSec), pp. 243–260, 2023, doi: 10.1007/978-3-031-37586-6_15.

[12] Y. Jiang et al., “MITRE ATT&CK applications in cybersecurity and the way forward,” arXiv:2502.10825v1, 2025.

[13] A. Tellache, A. A. Korba, A. Mokhtari, H. Moldovan, and Y. Ghamri-Doudane, “Advancing autonomous incident response: Leveraging LLMs and cyber threat intelligence,” arXiv:2508.10677v1, 2025.

[14] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, “Toward generating a new intrusion detection dataset and intrusion traffic characterization,” in Proc. 4th Int. Conf. Information Systems Security and Privacy (ICISSP), pp. 108–116, 2018.