Pragmatic DDoS detection on AWS cloud with lightweight machine learning*

Vitalii Molnar[1,*,†] and Dmytro Sabodashko[1,†]

[1] Lviv Polytechnic National University, 12 Stepan Bandera str., 79000 Lviv, Ukraine

Abstract

This study examines how training data composition and model capacity shape the reliability and generalization of an edge-based distributed denial-of-service (DDoS) detector on Amazon Web Services (AWS). We build a minimal pipeline from Amazon CloudFront logs using one-minute aggregates—request rate, unique client IPs, and 4xx/5xx counts. A separate validation set is used to select and fix a single operating threshold before testing. Two supervised detectors—Random Forest (RF) and Logistic Regression (LR)—are trained on a comprehensive dataset that combines benign background, a known high-requests-per-second (RPS) attack, and a benign flash crowd, and are evaluated on a continuous A→B→C→D timeline in which D is an unseen attack. With representative training coverage, both detectors remain stable on benign traffic with no false positives. Under distribution shift, only the nonlinear RF maintains sensitivity to the unseen attack, whereas the linear LR fails. These results motivate a practical design for availability-focused detection at the content delivery network (CDN) edge: rely on edge-visible signals, fix the operating threshold on a validation set, and deploy a lightweight nonlinear ensemble that integrates with a Web Application Firewall (WAF) under an explicit false-positive budget.

Keywords

DDoS detection, cloud security, AWS, application-layer (L7), Random Forest, CloudFront logs, flash crowd, false positives, specificity, precision–recall

1. Introduction

Cloud computing has become the backbone of contemporary digital ecosystems, enabling services to scale elastically and deliver content with unprecedented speed and global reach. Its promise of cost efficiency and reliability has accelerated adoption across critical domains, ranging from e-commerce and finance to healthcare and government platforms. With this deep integration into daily life, the expectation of continuous availability has become non-negotiable: even short disruptions can cascade into operational paralysis, financial damage, and reputational loss. As emphasized in cybersecurity frameworks, availability must be treated as a cornerstone of digital resilience and safeguarded with the same rigor as confidentiality and integrity [1].

Availability in large-scale cloud infrastructures is most critically challenged by Distributed Denial-of-Service (DDoS) attacks, which remain among the most disruptive incidents affecting online services [2–4]. The defining tactic of DDoS is overwhelming a target's capacity with surges of traffic that render legitimate services inaccessible [2, 5]. While early attacks were largely volumetric, recent campaigns increasingly combine network-level floods with application-layer (L7) request patterns that mimic genuine user behavior, complicating discrimination at the entry points of cloud platforms. Public reporting continues to rank DDoS as a systemic threat to cloud availability, with impacts that are not merely technical but translate directly into lost revenue, interrupted operations, and erosion of user trust [6].

Cloud providers respond with layered defenses—content delivery networks, Web Application Firewalls (WAFs), managed anti-DDoS rule sets, and rate-limiting—establishing essential baselines by filtering known signatures, enforcing quotas, and absorbing sudden surges. These controls are

* CPITS-II 2025: Workshop on Cybersecurity Providing in Information and Telecommunication Systems, October 26, 2025, Kyiv, Ukraine

* Corresponding author.

† These authors contributed equally.

vitalii.v.molnar@lpnu.ua (V. Molnar); dmytro.v.sabodashko@lpnu.ua (D. Sabodashko)

0009-0001-3183-0117 (V. Molnar); 0000-0003-1675-0976 (D. Sabodashko)

indispensable; yet they are predominantly reactive and often parameterized by static thresholds and coarse heuristics. When traffic is non-stationary, such thresholds become brittle: they may fail open during attacks or, conversely, trigger false positives during legitimate flash crowds. The resulting operational trade-off—under-protection versus over-protection—remains a recurring risk for availability-critical services [7–9].

This study moves beyond a simple performance comparison to investigate a more fundamental prerequisite for building a reliable DDoS classifier: the composition of its training data. The central hypothesis is that the ability of a lightweight classifier to distinguish between attacks and legitimate flash crowds is critically dependent on the representativeness of the dataset it was trained on. To test this, the research contrasts two distinct training methodologies. The first is a naive approach, where a model is trained only on benign background and attack patterns—a common practice that, as demonstrated, leads to poor generalization. The second is a comprehensive approach, where the training data is strategically enriched with examples of high-volume benign surges.

The evaluation is conducted on a continuous timeline that includes benign background traffic, a known attack pattern, a legitimate flash crowd, and a previously unseen attack scenario to rigorously test for stability and generalization. This work's primary contribution is the quantitative demonstration of the naive approach's failure modes and the subsequent proof that a targeted enrichment of training data is sufficient to achieve high stability without compromising attack detection. By restricting the feature set to minimal, edge-available signals from Amazon CloudFront logs and adhering to a strict frozen-threshold protocol, this study provides a pragmatic and reproducible framework for developing more robust DDoS detectors, emphasizing the critical role of data curation over mere algorithmic complexity.

2. Related Work and Literature Review

Research on mitigating DDoS in cloud environments spans from static perimeter defenses to learning-based detection and adaptive control. A widely cited starting point is the taxonomy by Somani et al., who mapped infrastructural and application-layer vectors in cloud settings and argued that purely reactive mechanisms and fixed thresholds are inadequate under elastic, Internet-scale demand [10]. This framing motivates methods that can discriminate malicious surges from legitimate traffic growth under non-stationary conditions.

Building on such taxonomies, the literature turned to supervised detection on labeled network data. Golduzian reported that classical models—including logistic regression, random forests, and gradient boosting—achieve strong detection accuracy on modern benchmarks and outperform simple statistical thresholds on precision and recall [11]. In a related vein, Abiramasundari and Ramaswamy compared several supervised classifiers (e.g., SVM, k-nearest neighbors) and observed near-perfect accuracy in controlled experiments, underscoring the promise of classification-based approaches when labels are reliable [12].

Parallel to classical supervised learning, deep architectures and sequence-aware methods have been explored to capture temporal structure in traffic. Ouhssini et al. proposed hybrid deep learning models for anomaly detection in cloud computing and highlighted the value of learning temporal dependencies for real-time decisions [13]. Complementing this direction, Song and Karri integrated early detection with software-defined networks and combined clustering with time-series modeling to support proactive reaction within SDN control planes [14].

Orthogonal to detection per se, some work emphasizes proactive surface dynamism. Kansal and Dave advanced a moving-target defense for cloud–fog environments that shifts exposed resources to raise attacker cost and disrupt campaign persistence [15]. While such strategies add resilience, they do not by themselves settle how to set auditable operating thresholds or how to separate flash crowds from application-layer bursts using minimal, provider-visible signals at content-delivery entry points.

Finally, comprehensive surveys in SDN contexts synthesize the breadth of techniques and open issues. Su et al. reviewed statistical anomaly detectors and AI-based classifiers for DDoS detection and mitigation in SDN, emphasizing persistent challenges of scalability, false positives, and tight integration with real-time enforcement pipelines [16]. This body of work collectively points to a gap: beyond achieving high offline accuracy, practical cloud defenses need reproducible operating-point selection under a false-positive budget, evaluation on benign flash-crowd regimes, and a transparent path to policy actuation in provider controls. The present study addresses this gap by testing lightweight classifiers on a minimal cloud-log feature set with a frozen-threshold protocol and by sketching auditable enforcement as a separate, conservative step.

3. Research Methodology

This section details the experimental design used to evaluate a minimal, edge-centric anti-DDoS approach. Our methodology prioritizes reproducibility and operational realism by combining a controlled web application stack with traffic scenarios that separately exercise legitimate flash crowds and adversarial L7 bursts. We first present the cloud architecture and logging/observability setup, then define the traffic scenarios and the dataset assembly pipeline based exclusively on minute-level CloudFront access logs. We next describe the learning protocol—feature construction, train/validation/test splits, model selection, and threshold freezing—and the baseline rule for comparison (a plain RPS threshold). Finally, we specify the evaluation metrics used under a strict false-positive constraint on a clean flash-crowd test. Forecasting and online policy tuning are deliberately out of scope here and are discussed separately as a safe operational sketch and future work rather than as part of the offline methodology [17].

3.1. Experimental Infrastructure Architecture

The experiment uses a small, realistic cloud-native stack that keeps the data path simple and the signals close to the perimeter. The target application is OWASP Juice Shop on Amazon ECS. Incoming requests terminate at Amazon CloudFront and are then forwarded to an Application Load Balancer (ALB), which distributes them across ECS tasks. CloudFront serves as the public entry point, caching static assets and shielding the origin from direct Internet exposure [18]. AWS Web Application Firewall (WAF) is attached at the CloudFront layer as the enforcement plane; rate-based and challenge actions are available but are not used for model training. In this methodology WAF remains a control surface for validation and audit rather than a data source for learning [19]. The overall setup and data flows are summarized in Figure 1: requests flow Clients → Amazon CloudFront (AWS WAF Web ACL attached) → Application Load Balancer (ALB) → ECS service; logging sends CloudFront access logs → S3 (training input; minute-level features: RPS, unique IPs, 4xx/5xx), while ALB/WAF logs → S3/CloudWatch for audit/enforcement only (not used for training); the detector's operating threshold is frozen on a disjoint validation slice (A+B). An accompanying public artifact repository is available [20].

Observability is configured to privilege signals available at the CDN perimeter. CloudFront standard access logs are enabled and delivered to Amazon S3 in structured form, exposing fields required to derive minute-level aggregates such as request rate (RPS), unique client IP count, and 4xx/5xx response totals [21]. ALB access logs are likewise written to S3 and support operational checks on latency and status codes; they are explicitly excluded from the training feature set to preserve the CloudFront-only learning assumption [22]. WAF logging is activated and streamed to S3 via Amazon Kinesis Data Firehose. These records capture rule evaluations and enforcement actions and are retained for auditability, avoiding leakage from control decisions into learning targets [23–28]. All logs are stored in a dedicated S3 bucket with a clear prefix for CloudFront records, which facilitates deterministic dataset assembly and later inspection.

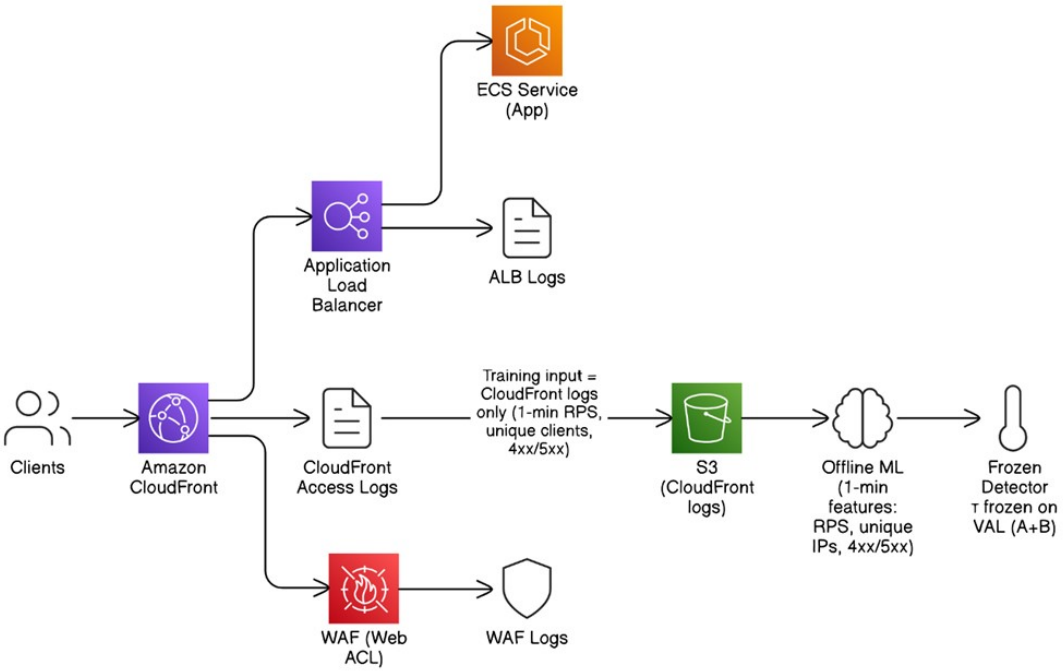

Figure 1: Minimal setup for edge-based DDoS detection on AWS

The architecture is modular. Each component—CloudFront distribution, WAF web ACL, ALB, and ECS service—can be scaled or replaced without altering the logging pipeline. The evaluation relies only on edge-observable features aggregated at one-minute cadence: RPS, unique IP count, 4xx/5xx counts, and a raw row count used as a coverage indicator. This design keeps the classifier transparent, reduces dependence on application-internal signals, and matches constraints typical of availability-critical systems. In practice, the setup mirrors a common production topology while remaining small enough to support rigorous and reproducible experiments.

3.2. Attack and Legitimate Traffic Scenarios

To facilitate a comprehensive training and evaluation process, four distinct, methodologically clean traffic scenarios were generated. All scenarios targeted the same CloudFront-ALB-ECS stack hosting the OWASP Juice Shop application. The design of these scenarios was crucial for creating a robust training set and for rigorously evaluating the final classifier's sensitivity, stability, and generalization capabilities.

A: Benign Background. This scenario provides a baseline of normal, low-intensity user activity. The traffic profile was intentionally kept stable and at a low volume (RPS consistently below 150) to create an unambiguous representation of a system under normal operating conditions. All data points generated in this scenario were labeled as benign (label = 0).

B: Known DDoS Attack. This scenario emulates a concentrated, application-layer DDoS attack, intended to be part of the training data. It consists of a scripted ramp-peak-cooldown sequence, characterized by a high volume of requests originating from a limited number of IP addresses. This low-diversity, high-volume pattern is a classic indicator of a denial-of-service attack. Data points within the core attack window were labeled as malicious (label = 1).

C: Benign Flash Crowd. This scenario models a legitimate, high-volume traffic surge, such as from a successful marketing campaign. Crucially, the traffic was generated from a diverse pool of source IP addresses, simulating a large number of unique users. The profile included a rapid ramp-up, a sustained high-volume peak (RPS > 400), and a gradual cooldown. Despite the high request rate, all data points in this scenario were labeled as benign (label = 0), serving as a critical example of "good" high-volume traffic for the training set.

D: Unseen DDoS Attack. This scenario was created to serve as a held-out test set for evaluating the model's generalization capabilities. While similar in structure to the "Known DDoS Attack" (B), it was generated using different parameters, request patterns, and source IP addresses. The model was never exposed to this data during training or validation, making it a true "blind" test of the classifier's ability to identify a novel threat. As with scenario B, data points within the core attack window were labeled as malicious (label = 1).

3.3. Dataset Preparation for Machine Learning

The dataset was constructed strictly from Amazon CloudFront standard access logs. Per-request records were processed to generate minute-level feature tables, while logs from other services like Application Load Balancer were excluded to prevent data leakage. The feature engineering process was intentionally kept minimal, focusing on a compact vector of edge-available signals: request rate (RPS), unique client IP count (cf_uniq_ip), and response error counts.

To ensure model robustness, a comprehensive TRAIN set was created. Preliminary experiments showed that a model trained naively on only background and attack data failed to generalize to legitimate traffic surges. The final training set therefore incorporates examples from three distinct scenarios: the benign background (A), the known DDoS attack (B), and the benign flash crowd (C). This representative dataset was shuffled to prevent the model from learning spurious temporal dependencies and to teach it the structural differences between malicious and legitimate high-volume events. A small, consistent VAL set was used for stable threshold tuning, composed of unambiguous positive samples from the core of attack B and negative samples from background A. The final TEST set was constructed as a continuous timeline concatenating all four scenarios (A $\rightarrow$ B $\rightarrow$ C $\rightarrow$ D), with the unseen DDoS attack (D) serving as a rigorous, held-out evaluation of generalization capability. The composition of these final datasets is summarized in Table 1. The inclusion of the fourth scenario, an unseen DDoS attack (D), served as a rigorous, held-out evaluation of the model's generalization capability on a novel threat.

Table 1
Dataset Composition Summary

Dataset Split	Total Minutes	Malicious Minutes (%)	Benign Minutes (%)
Train	89	17 (19.1%)	72 (80.9%)
VAL	8	4 (50%)	4 (50%)
TEST	159	34 (21.4%)	125 (78.6%)

3.4. Models and Evaluation Protocol

The primary model is a Random Forest (RF) classifier, chosen for its ability to capture nonlinear interactions between features without requiring extensive tuning. As a baseline for comparison, a linear Logistic Regression (LR) model is also evaluated to determine if a simpler decision boundary is sufficient for this task. Both machine learning models were trained on the comprehensive TRAIN set, which includes diverse examples of benign and malicious traffic. The third model is a non-learning RPS Threshold comparator, which serves as a simple benchmark to anchor the performance of the ML-based approaches.

To ensure a rigorous and operationally realistic evaluation, the analysis adheres to a strict frozen-threshold protocol. For both the RF and LR classifiers, an optimal operating point was determined by selecting a single decision threshold (τ). This threshold was chosen by maximizing the F1-score on the VAL set, which contains a balanced mix of unambiguous attack and benign samples. Once selected, this threshold was "frozen" and applied without any further adjustments to the final TEST set. This methodology is critical for preventing data leakage from the test distribution and for mirroring production environments where detection policies must be stable, auditable, and are not continuously re-tuned.

The evaluation metrics were selected to align with the primary operational goal of maximizing attack detection while strictly controlling for false positives on legitimate traffic. Performance on the VAL set was primarily assessed using precision, recall, and the F1-score. Operating points were selected using precision–recall analysis to align with an FP-dominant utility under class imbalance, while ROC/partial-ROC were also inspected to contextualize PR-based choices [26]. The final evaluation on the comprehensive TEST timeline focuses on direct, interpretable outcomes: the count of True Positives (TP) achieved during the known (B) and unseen (D) attack scenarios, and the count of False Positives (FP) generated during the benign background (A) and flash crowd (C) scenarios. This provides a clear, quantitative measure of each model's sensitivity and stability.

3.5. Evaluation Metrics

Evaluation follows a production-style constraint in which false positives during benign surges dominate operational risk. All quantities are computed on one-minute windows, and threshold selection on validation is kept strictly separate from testing to avoid leakage. On the frozen validation split, a scalar operating threshold τ is chosen at the F1 maximum. At that operating point, the report includes precision (precision = TP/(TP+FP)), recall—also referred to as the true positive rate (TPR = TP/(TP+FN))—and F1 (F1 = 2·precision·recall/(precision+recall)). Detection delay is measured as the difference, in minutes, between the first alarm and the onset of the contiguous attack core; a delay of zero indicates immediate detection at one-minute cadence. Operating points were selected using precision–recall analysis to align with an FP-dominant utility under class imbalance, while ROC/partial-ROC were also inspected to contextualize PR-based choices [26]. The flash-crowd stability subset consists exclusively of benign minutes and therefore measures stability rather than sensitivity. With τ frozen from validation, any alarm on test constitutes a false positive; results are summarized by the false-positive rate (FPR = FP/(FP+TN)) and by specificity (specificity = 1 − FPR), and are reported alongside the confusion outcomes (which reduce here to FP and TN because there are no attack-positive minutes by construction). To keep baselines comparable, the plain RPS rule is evaluated under the same discipline: its scalar threshold is selected on validation using the same precision–recall/F1 criterion and then applied unchanged on the flash-crowd test to obtain FPR and specificity. Finally, per-minute alarm traces for all models and baselines are retained so that precision, recall, F1, specificity, and delay can be recomputed and audited on the exact windows used in the study.

4. Results

This section presents the results of the final experiment. The classifiers were trained on the comprehensive dataset (composed of scenarios A, B, and C) and then evaluated on a continuous timeline that included all three training scenarios followed by a fourth, previously unseen attack scenario (D). This design allows for a thorough assessment of the models' stability on known benign traffic, their sensitivity to known attacks, and, crucially, their ability to generalize to novel threats.

4.1. Comparative Performance on the Test Timeline

The comparative performance of the Random Forest classifier, the Logistic Regression (LR) baseline, and a simple RPS threshold is visualized across the full experimental timeline in Figure 2. The figure plots the RPS profile and overlays markers indicating alarms from each detector, providing a direct visual comparison of their behavior in different traffic regimes.

An analysis of the timeline, quantitatively summarized in Table 2, reveals critical differences in model performance. A primary finding is that both the RF and LR models, after training on the comprehensive dataset, achieved perfect stability. During the Benign Background phase (A) and the Benign Flash Crowd phase (C), neither classifier produced a single false positive, correctly identifying all legitimate traffic.

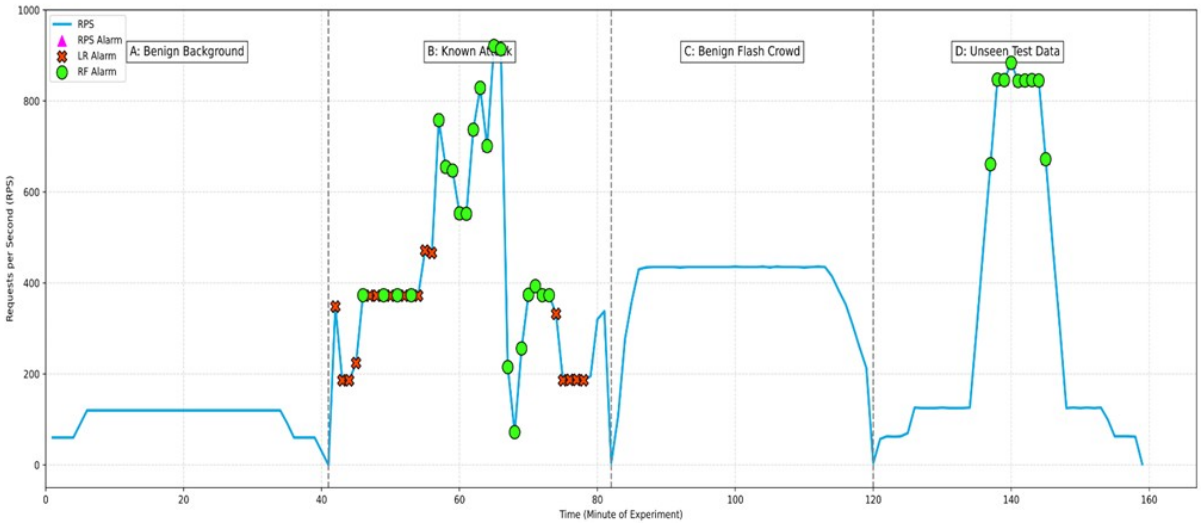

Figure 2: Full Experiment Timeline: Model Performance on Known and Unseen Data

In the Known Attack phase (B), both machine learning models proved effective. The LR model demonstrated slightly higher sensitivity, detecting 20 out of 21 attack minutes (95% recall). The RF model was also highly effective, identifying 17 out of 21 minutes (81% recall). Both significantly outperformed the simple RPS threshold.

The decisive test of robustness came in the final Unseen Test Data phase (D). Here, the Logistic Regression model completely failed to generalize, detecting 0 out of 13 minutes of the novel attack. In stark contrast, the Random Forest classifier successfully generalized its learning, identifying 9 out of 13 minutes (69% recall) of the new threat. This result highlights the fundamental inability of the linear model to adapt, whereas the nonlinear RF model proved effective against a previously unseen attack pattern.

Table 2
Quantitative Performance Summary

Model	False Positives (FP) on Benign Traffic (A + C)	True Positives (TP) on Known Attack (B)	True Positives (TP) on Unseen Attack (D)
Random Forest	0	17 / 21	9 / 13
Logistic Regression	0	20 / 21	0 / 13
RPS Threshold (500)	0	10 / 21	9 / 13

In summary, the results demonstrate a clear hierarchy of model performance. While both machine learning models learned to be perfectly stable on known benign traffic patterns, only the Random Forest classifier possessed the capacity to generalize its knowledge to effectively identify a novel, unseen threat. The complete failure of the Logistic Regression model on the unseen test data underscores the limitations of linear models for complex security tasks. These findings strongly suggest that a combination of a representative training dataset and a nonlinear model architecture is essential for building a truly robust DDoS detection system.

5. Discussion

The final evaluation indicates that reliability in edge-based DDoS detection on AWS is shaped jointly by training-data coverage and model capacity. After expanding TRAIN to include background (A), a known high-RPS attack (B), and a benign flash-crowd regime (C), both detectors operated at a validation-frozen threshold with zero false positives on benign minutes—evidence

that stability under a strict FP budget is primarily a property of representative coverage. Operating points were fixed on a disjoint validation slice using precision–recall analysis; ROC/partial-ROC was consulted only for context [24]. Model class nevertheless mattered under shift: at the same frozen threshold, Random Forest (RF) detected 17/21 minutes on B and 9/13 on the blind attack D, while Logistic Regression (LR) reached 20/21 on B but 0/13 on D, with all methods keeping FP=0 on A+C. A simple RPS=500 rule, tuned for zero FP, detected 10/21 on B and 9/13 on D. In short, the blind scenario exposes a generalization gap: RF retains sensitivity under distributional change whereas LR does not. These results support a pragmatic recipe for CDN edge detection: maintain a representative training envelope [6], use a lightweight nonlinear learner on tabular telemetry [27, 28], and fix thresholds on a separate validation set with PR-based selection [26].

5.1. Interpretation of results and operational meaning

Benign stability (FP=0 on A+C) after widening TRAIN shows that flash crowds share marginal signatures with attacks (elevated RPS, burstiness) yet differ in joint structure (client-IP diversity, error mix). Exposing those legitimate surges during training teaches safe regions that an attack-only curriculum cannot provide [6]. LR's collapse on D follows from its single linear boundary: with one-minute aggregates (RPS, unique clients, coarse 4xx/5xx), separability is interaction-drive —e.g., high RPS with low diversity and altered errors (attack core) versus high RPS with high diversity and stable errors (flash crowd). RF's piecewise partitions capture such interactions, preserving FP=0 while retaining partial sensitivity to novel morphology [29, 30].

5.2. Architectural Trade-offs: Minimal Edge Features vs. Detector Complexity

Learning was restricted to CloudFront-only signals—minute-level RPS, unique clients, and 4xx/5xx counts—while ALB and WAF telemetry was reserved for audit and enforcement [21–23]. This kept the pipeline portable and reproducible yet still discriminative once weak interactions were modeled by RF. A univariate RPS rule remained easy to audit but could not exploit cross-feature corroboration, forcing a trade-off between flash-crowd safety and recall. For parity, the RPS threshold was also selected and frozen on the disjoint validation slice using the same precision–recall criterion [28]. The minimal-features, frozen-threshold discipline aligns with guidance on traceability and auditability for AI-enabled controls [31].

5.3. Re-evaluating the Research Hypotheses

Both hypotheses are supported. A CloudFront-only classifier outperformed a plain RPS threshold on recall without violating a strict FP budget on the flash crowd, and a nonlinear learner proved more robust than a linear separator under the same frozen-threshold protocol. RF exceeded the RPS baseline on B and retained partial sensitivity on D, where LR failed completely—consistent with recent evidence on the strength of tree/ensemble methods for tabular data [29, 30]. PR-based operating-point selection reflects an FP-dominant utility under class imbalance [28].

5.4. Threats to Validity

Internal validity. Scenarios used a single-domain topology, one-minute aggregation, and bounded IP diversity. These choices prioritize label clarity and reproducibility but may narrow within-scenario variance. A single, small validation slice, even when disjoint from test, can overstate separation if its boundary is unusually clean; fixing the operating point before test mitigates but does not eliminate this risk [24].

External validity. The CloudFront⟶ALB stack is laboratory-bound and may not capture multi-origin, multi-region routing or broader adversary tactics. Evolving protocol-level stressors such as HTTP/2 Rapid Reset (CVE-2023-44487) challenge control paths in ways not exercised here [33]. Benign regimes beyond the single flash-crowd morphology (e.g., geo-distributed campaigns, device heterogeneity, cache-behavior variation) remain to be tested [6, 18].

Construct validity. Metrics are minute-level and edge-visible by design; they do not capture connection-churn micro-bursts or user-experience proxies beyond specificity. Precision–recall alignment is appropriate for FP-dominant, class-imbalanced protection, but alternative utility weightings (e.g., explicit FP/FN costs) could shift the chosen threshold [23]. Feature importance in ensembles is associational; ablation or counterfactual analyses would be required for stronger attribution [30].

Conclusion validity. The superiority of nonlinear capacity under distribution shift is shown at the pattern level; repeated trials with seed variation and bootstrap intervals would quantify uncertainty on TP/TN counts and specificity.

5.5. Future work

Operationalize with a conservative WAF control loop: score each minute; raise an alarm only after K consecutive exceedances; start with CHALLENGE/CAPTCHA or temporary BLOCK as a canary; escalate only if the condition persists; rollback on recovery, with all actions logged for audit [16, 33]. Strengthen external validity by extending flash-crowd tests to multi-region campaigns, heterogeneous devices, and varied cache behaviors, and by quantifying drift robustness under these conditions [18, 21, 22]. Enrich the adversary model with contemporary stressors (e.g., HTTP/2 Rapid Reset, path-focused L7 campaigns) while retaining edge-only observability [21, 32]. Tighten reliability estimates via ablations (seed variation, feature drop-outs), variance analyses of specificity/recall, longitudinal audits of threshold stability, and full experiment documentation with pinned dependencies and change control consistent with secure software development guidance [34]. Finally, include shallow yet diverse baselines (e.g., calibrated trees or ensembles with monotonic constraints) to map the minimal capacity required for benign-surge stability while keeping the implementation lightweight and auditable [29, 30].

6. Conclusion

This study examined whether the reliability and generalization of a pragmatic, edge-based DDoS detector on AWS are governed more by training data composition or by model choice. The evidence indicates that both elements are necessary and complementary. When the training set explicitly covered low, stable background traffic, a known high-RPS attack, and a benign flash-crowd regime, machine-learning detectors operated at a fixed, validation-frozen threshold with zero false positives on benign minutes—demonstrating that stability is primarily a property of representative training coverage. At the same time, model capacity determined performance under distribution shift: the Random Forest preserved stability and achieved partial sensitivity to a novel attack pattern, while Logistic Regression—despite strong performance on the known burst—failed to register any true positives on the blind scenario, consistent with the limits of linear decision surfaces for interaction-driven tabular telemetry.

These results support a pragmatic recipe for availability-centric DDoS detection at the CDN perimeter. First, keep features minimal and edge-visible—CloudFront-only aggregates, with ALB/WAF reserved for audit and enforcement rather than training — so that the pipeline remains portable, auditable, and decoupled from origin specifics. Second, select and freeze the operating point using precision–recall analysis aligned to a strict false-positive budget, which is more informative than ROC when classes are imbalanced and availability costs dominate [24]. Third, prefer a lightweight nonlinear learner to capture weak cross-feature interactions without resorting to deep or sequence models. This design aligns with contemporary governance guidance that emphasizes traceability and risk-aware controls for AI-enabled systems and is compatible with staged, reversible enforcement via documented WAF actions (canary → enforce → rollback).

Limitations qualify the scope of the claim. The evaluation used a single-domain topology, one-minute cadence, and bounded IP diversity; broader benign regimes (multi-region campaigns, heterogeneous devices, cache behavior variation) and richer adversarial tactics (e.g., protocol-level stressors such as HTTP/2 Rapid Reset) warrant systematic inclusion before generalizing further.

Nevertheless, the core conclusion is robust within the examined constraints: representative training data ensures stability, and modest nonlinear capacity enables generalization to novel attack morphology at fixed thresholds. In operational terms, this amounts to a conservative but effective blueprint: edge-only features, PR-frozen thresholds, and lightweight nonlinearity, implemented under auditable change control and evaluated against an explicit false-positive budge —a configuration consistent with current security and AI risk-management guidance.

Declaration on Generative AI

While preparing this work, the authors used the AI programs Grammarly Pro to correct text grammar and Strike Plagiarism to search for possible plagiarism. After using this tool, the authors reviewed and edited the content as needed and took full responsibility for the publication's content.

References

[1] National Institute of Standards and Technology, Framework for Improving Critical Infrastructure Cybersecurity, Version 1.1, 2018. https://www.nist.gov/cyberframework

[2] A. Ilyenko, et al., Practical Aspects of using Fully Homomorphic Encryption Systems to Protect Cloud Computing, in: Cybersecur. Prov. in Inf. and Telecomm. Syst. II, vol. 3550, 2023, 226–233.

[3] Y. Martseniuk, et al., Universal Centralized Secret Data Management for Automated Public Cloud Provisioning, in: Cybersecur. Prov. in Inf. and Telecomm. Syst. II, vol. 3826, 2024, 72–81.

[4] Y. Martseniuk, et al, Research of the Centralized Configuration Repository Efficiency for Secure Cloud Service Infrastructure Management, in: Cybersecur. Prov. in Inf. and Telecomm. Syst., vol. 3991, 2025, 260–274.

[5] O. Mykhaylova, M. Korol, R. Kyrychok, Research and Analysis of Issues and Challenges in Ensuring Cyber Security in Cloud Computing, in: Cybersecur. Prov. in Inf. and Telecomm. Syst. II, vol. 3826, 2024, 30–39.

[6] European Union Agency for Cybersecurity, ENISA Threat Landscape 2023, 2023. https://www.enisa.europa.eu/publications

[7] Amazon Web Services, Best Practices for Anti-DDoS, in: AWS WAF and AWS Shield Developer Guide, 2025. https://docs.aws.amazon.com/waf/latest/developerguide/

[8] I. Opirskyy, et al., Modern Methods of Ensuring Information Protection in Cybersecurity Systems using Artificial Intelligence and Blockchain Technology, Kharkiv: Technology Center PC, 2025. doi:10.15587/978-617-8360-12-2

[9] P. Petriv, I. Opirskyy, N. Mazur, Modern Technologies of Decentralized Databases, Authentication, and Authorization Methods, in: Cybersecur. Prov. in Inf. and Telecomm. Syst. II, vol. 3826, 2024, 60–71.

[10] G. Somani, et al., DDoS Attacks in Cloud Computing: Issues, Taxonomy, and Future Directions. Comput. Comm., 107 (2017) 30–48. doi:10.1016/j.comcom.2017.03.010

[11] A. Golduzian, Predict and Prevent DDoS Attacks using Machine Learning and Statistical Algorithms, arXiv, doi:10.48550/arXiv.2308.15674

[12] V. Abiramasundari, R. Ramaswamy, Distributed Denial-of-Service (DDoS) Attack Detection using Supervised Machine Learning Algorithms, Sci. Rep., 15 (2025) 13098. doi:10.1038/s41598-024-84879-y

[13] M. Ouhssini, et al., DeepDefend: A Comprehensive Framework for DDoS Attack Detection and Prevention in Cloud Computing, J. King Saud Univ.—Comput. Inf. Sci., 36 (2024) 101938. doi:10.1016/j.jksuci.2024.101938

[14] A. V. Songa, G. R. Karri, An Integrated SDN Framework for Early DDoS Detection and Mitigation, J. Cloud Comput., 13 (2024) 64. doi:10.1186/s13677-024-00625-9

[15] V. Kansal, M. Dave, Proactive DDoS Attack Mitigation in Cloud-Fog Environment using Moving Target Defense, doi:10.48550/arXiv.2012.01964

[16] Y. Su, D. Xiong, K. Qian, Y. Wang, A Comprehensive Survey of Distributed Denial of Service Detection and Mitigation Technologies in Software-Defined Network, Electronics, 13 (2024) 807. doi:10.3390/electronics13040807
[17] V. Shapoval, et al., Automation of Data Management Processes in Cloud Storage, in: Cybersecurity Providing in Information and Telecomm. Systems, vol. 3654, 2024, 410–418.
[18] Amazon Web Services, Using CloudFront to Protect Applications, in: Amazon CloudFront Developer Guide, 2025. https://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/
[19] Amazon Web Services, How AWS WAF Works, in: AWS WAF Developer Guide, 2025. https://docs.aws.amazon.com/waf/latest/developerguide/how-aws-waf-works.html
[20] V. Molnar, Pragmatic DDoS Detection on AWS Cloud with Lightweight ML (artifact repository), GitHub, 2025. https://github.com/j9mbo/pragmatic-ddos-detection-on-aws-cloud-with-lightweight-ml
[21] Amazon Web Services, Access Logs for Amazon CloudFront, in: Amazon CloudFront Developer Guide, 2025. https://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/AccessLogs.html
[22] Amazon Web Services, Access logs for Application Load Balancer, in: Elastic Load Balancing User Guide, 2025. https://docs.aws.amazon.com/elasticloadbalancing/latest/application/load-balancer-access-logs.html
[23] Amazon Web Services, Logging Web ACL Traffic Information, in: AWS WAF Developer Guide, 2025. https://docs.aws.amazon.com/waf/latest/developerguide/logging.html
[24] O. Vakhula, I. Opirskyy, O. Mykhaylova, Research on Security Challenges in Cloud Environments and Solutions based on the “Security-as-Code” Approach, in: Cybersecurity Providing in Information and Telecommunication Systems II, vol. 3550, 2023, 55–69.
[25] V. Susukailo, I. Opirsky, O. Yaremko, Methodology of ISMS Establishment against Modern Cybersecurity Threats, in: Lecture Notes Electr. Eng., Springer Int. Publ., Cham, 2021, 257–271. doi:10.1007/978-3-030-92435-5_15
[26] National Institute of Standards and Technology, The NIST Cybersecurity Framework (CSF) 2.0, NIST CSWP 29, 2024. doi:10.6028/NIST.CSWP.29
[27] D. Kalambe, A. Sureka, S. Sengupta, A comprehensive Plane-Wise Review of DDoS Attacks in SDN: Leveraging Detection and Mitigation through Machine Learning and Deep Learning, J. Netw. Comput. Appl., 235 (2024) 104081. doi:10.1016/j.jnca.2024.104081
[28] E. Richardson, et al., The Receiver Operating Characteristic Curve Accurately Assesses Imbalanced Datasets, Patterns, 5 (2024) 100994. doi:10.1016/j.patter.2024.100994
[29] L. Grinsztajn, E. Oyallon, G. Varoquaux, Why do Tree-based Models Still Outperform Deep Learning on Tabular Data?, arXiv:2207.08815, 2022. doi:10.48550/arXiv.2207.08815
[30] D. McElfresh, S. Jain, F. Sha, Z. C. Lipton, When Do Neural Nets Outperform Boosted Trees on Tabular Data?, arXiv, 2023. doi:10.48550/arXiv.2305.02997
[31] P. Goldschmidt, D. Chudá, Network Intrusion Datasets: A Survey, Limitations, and Recommendations, Comput. Secur., 156 (2025) 104510. doi:10.1016/j.cose.2025.104510
[32] Cybersecurity and Infrastructure Security Agency, HTTP/2 Rapid Reset Vulnerability (CVE-2023-44487), Security Alert, 2023. https://www.cisa.gov/news-events/alerts/2023/10/10/http2-rapid-reset
[33] Amazon Web Services, Using Rule Actions in AWS WAF; CAPTCHA and Challenge in AWS WAF, in: AWS WAF Developer Guide, 2025. https://docs.aws.amazon.com/waf/latest/developerguide/waf-rule-actions.html
[34] National Institute of Standards and Technology, Secure Software Development Framework (SSDF) Version 1.1: Recommendations for mitigating the risk of software vulnerabilities, NIST SP 800-218, 2022. https://csrc.nist.gov/publications/detail/sp/800-218/final