International Journal of Emerging Research in Engineering and Technology
Pearl Blue Research Group| Volume 4, Issue 1, 175-185, 2023
ISSN: 3050-922X | https://doi.org/10.63282/3050-922X.IJERET-V4I1P119

Original Article

Real-Time Anomaly Detection and Attack Mitigation for Cloud-Based Content Delivery Paths Using AI

Siva Sai Krishna Suryadevara[1], Kareem Shaik[2]
[1]Sr. AEM Cloud Engineer at Maganti IT Resources, USA.
[2]Sr. AEM Developer at Fidelity Investments, TX, USA.

Abstract *- Cloud-based content delivery paths, which include client apps, edge PoPs, backbone links & origin services, are being attacked and misconfigured in ways that make them very less reliable, more expensive, and less trustworthy. Because CDN traffic changes in milliseconds and failures quickly spread across more regions, finding and fixing problems must happen in actual time, not on these dashboards that show what happened after the fact. This study suggests an actual time AI-driven pipeline that learns what "normal" delivery behavior is and steps in as soon as it sees a risk. Telemetry is gathered from multiple levels, such as edge access logs, request/response headers, TLS and QUIC handshake metadata, cache hit/miss streams, routing along with their BGP updates, queueing and loss statistics, and origin health indicators. These signals are then combined into time-aligned flow windows. A hybrid model stack has self-supervised sequence encoders for traffic patterns with a lot of different values, graph-based predictors for path & PoP connections, and lightweight online changepoint detectors to find sudden changes. When an anomaly is detected, a policy engine ranks the most likely causes, such as L7 floods, cache poisoning these attempts, bot spikes, route leaks, or regional origin brownouts. It then runs mitigation playbooks, which include adaptive rate limiting, dynamic WAF rules, cache key hardening, smart rerouting, or selective origin shielding. The system is tested by playing back multi-region traces at line rate and adding controlled red-team injections. This tests detection delay, false positive rate & end-to-end user effect under load. The results show that detection is substantially faster (with a median time-to-alert of less than a second), that context-aware fusion greatly reduces noisy alerts, and that mitigation works very well with little collateral throttling. Overall, the method gives CDN operators and cloud service providers a way to maintain their delivery paths stable even when things change quickly. It turns raw information into fast, automatic defense. The same cycle of learning and doing may also be used for multi-cloud ingress, SASE edges, and next-gen content routing, which will make internet-scale delivery safer and more independent.*

Keywords *- Real-Time Anomaly Detection, Cloud Security, Content Delivery Networks (CDN), Edge-To-Origin Delivery Paths, AI-Driven Threat Detection, Streaming Telemetry Analytics, Network Traffic Modeling, Graph-Based Path Intelligence, Deep Learning For Attacks, Automated Attack Mitigation, Adaptive WAF/Rate-Limiting, Concept Drift Handling, Multi-Cloud Edge Computing, Zero-Trust Delivery Routing.*

1. Introduction

Cloud-based distribution systems and content distribution networks (CDNs) are now the building blocks of the modern internet. A complex network of cloud regions, edge nodes, private backbones & origin servers usually handles each video stream, API request, mobile app update, or personalized homepage. This complicated supply chain makes things more available as well as very faster, but it also makes the attack surface bigger, which is hard for regular security solutions to protect in real time.

Modern opponents take advantage of how quickly and easily cloud infrastructure can be used. Attacks change quickly, often mixing in with lawful traffic and taking advantage of the fact that companies can serve millions of their customers. At the same time, businesses need constant performance, strict SLAs, and no tolerance for downtime or changes to their information.

This study looks at the growing gap between how content delivery systems are changing and how rule-based or signature-driven security systems are limited. AI-driven anomaly detection and automatic remediation offer a system that analyzes trends in huge amounts of streaming information, adapts to changing their traffic patterns, and takes action before problems arise to avoid major risks.

The next parts explain the actual world challenges, identify the problem at hand, and explain why an AI-driven, closed-loop solution is needed for the actual time security of cloud-based content distribution.

1.1. Challenges

Getting cloud-based delivery channels to work is very hard since the environment is always changing. Modern delivery channels are always changing. They include many cloud providers, edge networks, CDNs, and origin infrastructures. Traffic

may go via different areas, switch across these caches, or be sent to a different location when a failover happens. Because things change so quickly, strict rules or set thresholds are almost useless at keeping things accurate.

Another issue is the huge quantity and variety of operating signals. A single delivery route can create streaming logs, actual time traces, NetFlow records, WAF alarms, DNS requests, TLS handshake metrics, media bitrate their information, and edge performance measurements. Conventional monitoring systems can't handle these inputs at scale and put them together to make important discoveries.

The attack surface is changing. Many modern threats work in very short amounts of time. For example, micro-burst DDoS attacks last only a few seconds, cache poisoning attacks put harmful entities into these edge nodes, routing manipulation changes the flow of traffic, and bot swarms mix in with user traffic patterns. The damage is often permanent by the time an analyst encounters these cases by hand.

Noise and idea drift make things even harder. Traffic naturally fluctuates because of differences across regions, spikes in usage at certain times, special events, and changes in the appeal of material. Without adaptive models, it's hard to tell the difference between "expected change" and "malicious anomaly."

In the end, actual time needs to call for strict constraints on latency. Detection needs to happen quickly, ideally in less than a second, to avoid hurting user experience, latency, or SLA commitments. If an attack isn't found or stopped right away, millions of people could be affected right away.

All of these problems make it very hard for traditional rule-based systems to work, hence AI-driven, adaptive, real-time detection is necessary.

1.2. Problem Statement

This paper talks about how to find and stop strange or malicious behaviors in cloud-based content delivery channels in actual time. In this case, an anomaly is any change from expected traffic patterns, performance benchmarks, or routing behaviors across edge nodes, cloud regions, and origin infrastructures. These strange things could mean threats like short-burst DDoS attacks, routing manipulation, cache poisoning, credential-stuffing botnets, or any other exploitative patterns that use multi-hop delivery chains.

Traditional monitoring tools use fixed thresholds, signatures, or parameters that need to be changed by hand. These systems don't work well in very dynamic delivery situations since they can't quickly respond to changes in their traffic, concept drift, or new attack vectors. Because of this, actual abnormalities may go unnoticed, while routine changes may cause false signals.

A real-time detection and mitigation system should give three basic guarantees:

- Velocity: Find the latest problems with sub-second latency so that users don't have to deal with them.
- Precision: Keep the number of false positives & false negatives as low as possible when dealing with noisy, high-volume data sources.
- Secure Automation: Start mitigations like rerouting, limiting the rate, clearing the cache, or changing the WAF rules in a way that is methodical, reversible & more reliable.

This research develops a measurable, AI-driven framework that perpetually scrutinizes streaming telemetry from multiple layers of the delivery path, detects anomalies through statistical as well as machine learning techniques, and autonomously executes mitigation measures while ensuring system safety and stability. The problem is that cloud-based content delivery systems need to be protected in real time in a way that is reliable, easy to understand, and flexible.

1.3. Motivation

This study is driven by the increasing economic & operational risks associated with undetected or delayed detection of attacks on cloud delivery paths. A tiny disruption, lasting solely a few seconds, potentially impair the user experience, invalidate SLAs, lose revenue, or put vital information at danger. Apps today demand quick and reliable transmission systems. Any difficulties along those routes could have big implications.

Cache poisoning incidents can update data on global nodes at the edges without anybody being aware. This can lead to inaccurate data, harmful downloaded files, or damage to a particular individual's reputation. DDoS assaults and increased traffic created by bots may wear down or even break down regional systems, which can lead to connectivity disruptions as well as latency spikes. The routing-related exploitation might direct traffic by means of rogue intermediaries. Traditional security measures fail because these attacks often look like normal traffic until the exact time they happen.

Artificial intelligence is particularly ideally suited to cope with these various types of difficulties. AI models might detect patterns in immense, heterogeneous communication streams and use them to generate these judgments, unlike rule-based systems. They can identify tiny links that others might miss and create predictions concerning new or rising hazards. Machine learning renders it feasible for the system to continually adapt with the idea drift, thereby assisting it to comprehend what "normal" behavior can be as the surroundings changes.

However, just finding anything is not enough. Companies need closed-loop mitigation, which means that the system not only finds an irregularity but also figures out what to do about it and does it. A closed-loop system can automatically limit the amount of questionable traffic, isolate affected edge nodes, change delivery paths, or execute a targeted cache purge, often before it causes any other problems for users. This reduces the need for people to step in, which isn't enough given the speeds of the present day's attacks.

The main goal is resilience, which is to make sure that electronic delivery channels remain safe, fast, and dependable, even when challenges are intricate and develop quickly. By incorporating AI-powered detection alongside automated mitigation, organizations can transition about responding to threats to proactively safeguarding themselves. This keeps the organization going, protects user trust, as well as keeps the worldwide multimedia distribution system safe.

2. Literature Review

2.1. Anomaly Detection in Cloud & CDN Traffic

In the past, typical identification of anomalies in cloud and CDN settings used statistics profiling, which employs historical data to indicate regular traffic patterns while deviations as probable risks. Static thresholds, dynamic moving averages, as well as EWMA (Exponentially Weighted Moving Average) are all standard ways of evaluating short-term changes by giving greater significance to more recent data points. Advanced time- series models like ARIMA are utilized for finding seasonal trends and time-varying correlations in access records, latency measurements, along with edge-node usage data.

These statistical models have clear advantages: they are easy to use, easy to understand, and don't cost much to run, which makes them very easy to use on CDN edges that are spread out over a wide area. Still, research keeps showing how bad they are when used with traffic that isn't stable, which is common for global CDNs that see sudden spikes in their demand (like when viral material is released). Actual surges can readily fool static thresholds, but EWMA and ARIMA have trouble when behavior changes very quickly because of routing improvements, new content, or changes in cache policy. As a result, there are a lot more false positives, and actual low-and-slow anomalies are sometimes missed. Recent research aims to auto-tune baselines using adaptive thresholding; nevertheless, these methods are still heavily reliant on their stable historical patterns an assumption that rarely holds in edge-heavy systems.

In essence, statistical methods are becoming very less useful for modern cloud delivery systems because traffic patterns change too quickly for static or semi-static baselines to be accurate.

2.2. Deep Learning and Machine Learning for Path and Network Anomalies

Researchers are currently looking at machine learning & deep learning methods to fix the problems with statistical models that don't work well. There aren't numerous labeled attacking data sets, hence Isolation Forests and Autoencoders constitute two exceptionally prevalent types of independent machine learning approaches. Autoencoders hunt for unexpected patterns by looking at reconstructed errors and short breakdowns of normal flow. Isolated Forests, on the reverse hand, find outlier distributions by splitting themselves off at random. The models in question have been shown better at handling non-linear modifications than statistical initial scenarios, and they are less difficult to change as their traffic habits develop.

Long short-term memory (LSTM) and gated recurrent units (GRU) are two types of neural network recurrent networks that can learn persistent dependencies in latency, velocity, and hop-level metrics along the routes of delivery. Recently, transformer-based architectures have become more popular since they can examine sequences at the same time, which speeds up inference, which is very important for actual time applications. Graph-based learning methods show the delivery path as a structured graph with nodes and edges, in addition to sequential data. By learning embeddings of the network architecture, these models can find changes in routing, edge-node dynamics, and path abnormalities. This is especially important in CDNs because the "context" of an anomaly depends not just on traffic metrics but also on how nodes are connected to each other.

Even though ML/DL approaches offer several advantages, they also have a number of different practical issues. It's quite challenging to draw inferences in real time, particularly in deep models that require GPU acceleration or sophisticated preprocessing procedures. Sometimes, CDN servers at the edges put a greater priority on throughput than on data analytics that utilize a lot of processing computing power. This makes it more challenging to use large models. Second, understanding is always a problem. Operators need clear reasons for their choices, especially when warnings for anomalies require expensive steps like invalidating a cache or rerouting traffic. Black-box models make this problem worse, which makes it harder for them

to be used in production settings. Ultimately, models may worsen when distribution channels change, making drift detection & continuing retraining essential; however, these methods are not yet widely used in operational CDNs.

2.3. How to stop attacks in delivery systems

Historically, rule-based and infrastructure-level protections have been used to protect against attacks in cloud and CDN delivery pipelines. Web Application Firewalls (WAFs) look for strange HTTP patterns by using signature rules or heuristics. People often use rate limiting to stop volumetric attacks or credential stuffing attempts. Content Delivery Networks (CDNs) use scrubbing centers to send questionable traffic to a different location and filter it before it reaches the origin.

Anycast routing at the network layer makes it possible to spread attack traffic across many other nodes, which makes it easier on any one edge site. Other architectural defenses, such as cache invalidation mechanisms, origin shielding, and request compression, try to lessen the damage caused by malicious spikes or strange surges. These tactics work well to absorb or deflect big surges, but they mostly react to them based on set thresholds or manually changed signatures. High-impact solutions, like changing the rules for the global WAF or making big reroutes, need people to make these decisions, which slows down incident management.

Researchers stress a constant shortcoming: these mitigation strategies work, but they don't have adaptive intelligence. Modern methods don't often look at how attack plans change over time or how path-level abnormalities differ from known signatures. As cloud delivery settings move closer to automation, this human, reactive approach becomes very less and less useful.

2.4. MLOps and Streaming Security Analytics

Modern systems use streaming analytics frameworks like Apache Kafka, Apache Flink, and Spark Structured Streaming to help people make these decisions in real time. These solutions make it easier to collect and analyze logs, analytics, and network information on a huge scale all the time. They work best in CDN settings where milliseconds matter, and they lower the time it takes to find things for both statistical & machine learning models.

Putting machine learning into streaming pipelines can cause these problems with how they work. Models that are used at the edge need to be able to handle low-latency inference, regular retraining, and the complexities of model versioning and rollback. Traffic patterns also change over time, so to keep the model accurate, it needs to be able to identify drift, learn online, and retrain itself automatically. Even if MLOps technologies have gotten better in data-centric companies, CDN-specific deployment situations, which have hundreds of remote nodes, still have unique scalability problems. Studies show that most existing implementations stop at batch retraining or scheduled updates. This makes streaming pipelines vulnerable to outdated models when traffic changes a lot.

2.5. Summary of the Gaps in the Literature

The main problem in all these categories is that there is no fully automated, actual time way to find and fix these problems that is specifically made for delivery routes that change all the time. Statistical methods are rigid; machine learning and deep learning models struggle with interpretability & actual time application; and traditional mitigation solutions tend to be reactive instead of proactive. MLOps for streaming settings are still being developed, which is a great chance for AI-driven systems that can quickly find problems and automatically put fixes in place.

Table 1: Comparison of Detection Methods across Latency, Robustness, and Automation Readiness

Method Type	Detection Latency	Robustness to Traffic Shifts	Automation Readiness
Statistical baselines	Very low	Low to moderate	Low
ML (Isolation Forest/Autoencoders)	Low to moderate	Moderate	Moderate
Deep Learning (LSTM/GRU/Transformers)	Moderate	High	Moderate
Graph-based models	Moderate	High	Moderate
Rule-based mitigation (WAF/Rate limits)	Low	Low	Low
Streaming + MLOps pipelines	Very low	Moderate (depends on model drift handling)	Moderate to High

3. Proposed Methodology

This part explains how to find and stop assaults and strange behavior on cloud-based content delivery paths in actual time. The main idea is to think of the end-to-end delivery pathway (from user to edge to PoP to peering to origin to back) as a dynamic system that always sends telemetry. We turn information into features, let an AI detector look for problems, turn it into a path-level risk score, and then quickly take steps to fix the problem while learning from what happens. The focus is not

only on finding typical volumetric attacks, but also on finding small path disruptions (such as routing drift, cache manipulation, or edge-origin flip-flops) that might secretly harm service or make security holes.

3.1. System Overview

The system has a closed loop with six parts: Telemetry, Feature stream, AI detector, Risk scorer, Mitigation engine as well as Feedback loop. The delivery stack, which includes edge nodes, Points of Presence (PoPs), origin services, DNS resolvers, and network control planes, sends telemetry data all the time. Each signal has a time stamp and a "delivery path identity," which is a tuple made up of the client area, edge cluster, PoP, origin & principal AS hops that were used.

The telemetry is made into a feature stream in almost actual time. For each sliding window and each path/session, features are calculated and then sent to the AI detector. The detector gives an anomaly score that shows how far current behavior is from what is considered typical. A risk player takes the unusual occurrence evaluation and the policy's historical context, including the significance the asset is, the blast extent, how often these events happen, and how likely and sure they are, and then comes up with a risk value which seems right for the present scenario.

When risk goes above certain levels, the mitigation engine turns on protections like rate limiting, bot challenges, cache isolation, forced revalidation, rerouting, or origin shielding. The feedback loop looks at the results (good or bad consequences of mitigation) to change thresholds, improve the weights of the online model & group events for future training. This makes the system both fast and able to improve itself.

3.2. Threat & Anomaly Model

We think of the delivery pathway as a series of parts: edge nodes (close to users), Points of Presence (PoPs) that combine edge traffic, origin servers that store authoritative content, peering links that connect CDNs to upstream networks, DNS layers that resolve endpoints, and cache tiers (edge cache, mid-tier cache, and origin cache). Incidents or failures may occur in a single segment but propagate across the pathway; therefore, we analyze both localized & associated anomalies.

- DDoS and request inundations are types of targeted attacks that include sudden spikes in volume, high SYN/UDP rates, or requests that don't happen at the same time.
- Bot abuse: rare yet widespread searches, quick changes of IPs/agents, or strange session behaviors.
- Cache poisoning or cache bypass happens when there are sudden changes in TTL, a lot of different URIs, or unstable cache hit ratios.
- BGP/routing problems include changes to the AS path that weren't expected, hop inflation, detours to Points of their Presence, or regional blackholing.
- When someone tries to spoof or downgrade TLS, there are more handshake problems, changes in cipher suites, or strange SNI patterns.
- DNS manipulation: sudden increases in NXDOMAIN answers, delays that are peculiar to the resolver, or resolution loops that don't go away.

There are three ways to look at these dangers: traffic characteristics (rates, distributions, entropy), path configuration (routing/AS/hops, edge-origin switching), and performance/security indications (delay, cache effectiveness, TLS/WAF failures). The anomaly model suggests that normal behavior is affected by seasonal as well as regional conditions. As a result, deviations are measured against established baselines rather than strict standards.

3.3. Data Collection & Streaming Pipeline

The pipeline combines their information from many other sources to get rid of blind spots. We take in CDN access logs, which include requests, status codes, URIs, user agents & cache decisions.

- Time to First Byte, throughput, queue depth as well as their connection failures are all examples of edge performance metrics.
- NetFlow/packet summary (Layer 3/Layer 4 rates, flags, and main communicators).
- DNS logs show how often queries are made, what response codes are sent, and how long it takes for each resolver to respond.
- WAF alerts (activating signatures, evaluating bots).
- Trace spans include all the times from the edge to the Point of Presence (PoP) to the origin.

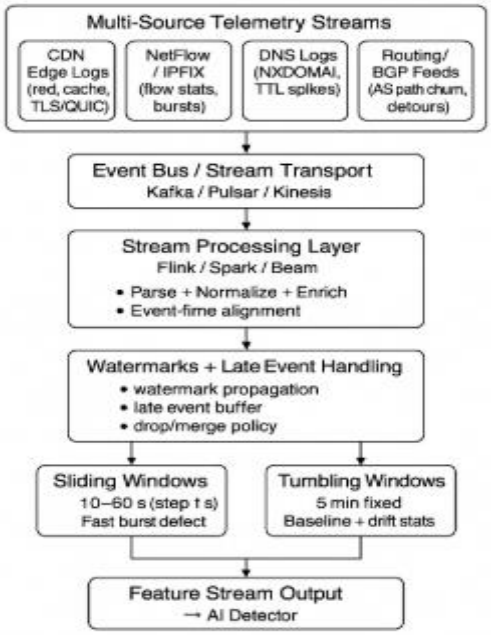

Fig 1: Multi-Source Streaming Ingestion with Sliding/Tumbling Window Alignment.

Events are sent to a streaming bus and sorted by delivery path ID. We use sliding windows for quick anomaly identification (for example, periods of 10 to 60 seconds that are updated every few seconds) and tumbling windows for steady averages (for example, 5-minute summary used for trend and drift analysis). The dual technique helps with both alarms that are too sensitive as well as these alarms that take too long to go off.

Each stream has a little watermark delay to deal with events that are late or absent. If information comes in late but is still inside the watermark, we update the window averages. If not, we save it for later correction offline. When signals are sparse, like routing information, they are carefully forward-filled with a drop in the confidence. This lets the detector know when the input is out of date. This keeps actual time inference strong without presuming that telemetry is perfect.

3.4. Feature Engineering

Features are made to show attacks while keeping things stable even when traffic changes naturally.

Traffic-rate attributes include the basic volume and configuration, such as requests per second per path, bytes per second, connection attempts, the ratio of 4xx to 5xx failures, cache hit and miss rates & concurrency metrics by IP or session. We use burstiness indicators like short-term versus long-term rate ratios and inter-arrival variance to tell the difference between real surges (like product debuts) and fake floods.

For covert exploitation, entropy-based traits are very important. We figure out the entropy for URIs, query strings, user agents & source IP addresses. Cache poisoning often shows up as a rise in URI entropy and a drop in the cache hit ratio. Botnets show a rise in IP entropy and a drop in agent entropy (multiple IPs with the same agents).

Geo-skew features show spatial asymmetry, such as when regional request proportions differ from a baseline, traffic suddenly appears in unlikely places, or the same session token quickly moves from one region to another. These work well against bots that do DDoS attacks and credential stuffing.

Path topological characteristics find many problems with delivery routes. We keep an eye on the number of hops, the order of the AS-paths, how often AS changes, the rate of PoP detours, and the frequency of edge-origin switches. A sudden surge in hop count or AS churn is often a sign of routing leaks or BGP hijacks. Too much edge-origin flipping means that the cache tier is unstable or that someone is trying to get around it on purpose.

Temporal context elements include the time of day, the day of the week, and rolling seasonal baselines that are grouped by area & content class. This stops normal daily peaks from looking like attacks.

In the end, we use normalization and scaling that takes drift into their account. Each feature is z-scored against a rolling baseline that changes over time (from minutes to hours) but keeps sudden anomalies (within seconds) intact. When baseline

drift is found, like a trend that keeps going up, scaling settings are changed with a safety net to stop the unintentional weakening of a long, sluggish attack.

3.5. AI-Based Real-Time Anomaly Detection

We use a mixed AI detector because delivery anomalies can be both types. Primarily, many dangers are temporary & show up as bursts, ramps, or rhythmic bot waves. A sequence model, like a lightweight temporal encoder, learns the usual patterns of time for each path. Second, some hazards change the route itself, including sudden AS reroutes, PoP diversions, or cache tier transitions. A model that knows about graphs looks at path-topology properties & finds changes in structure.

There are two parts to the training process. We do offline pretraining using historical traffic that has been marked with known incidents & long periods of good behavior. This gives the models a strong grasp of how things usually change with the seasons. We provide online adaptability through small, limited adjustments to keep the detector in line with the latest applications, geographical areas, and routing dynamics. When risk levels are very high, online learning is limited, making it harder for the model to adapt to an attack.

The detector gives each path, session, or window an anomaly score. Scores are modified into risk assessments by using sensitivity profiles (critical APIs versus static assets). Inference is designed to be cost-effective: feature calculation runs at O(n) per window, sequence inference is linear in relation to window length, and graph scoring is based on path size. The detector works with very strict latency limits (single-digit milliseconds per path window), which makes it possible to take action to stop the attack before it uses up all the resources.

4. Case Study

4.1. Environment & Setup

StreamWave, a worldwide media streaming corporation, offers live sports and on-demand video content to subscribers in North America, Europe & Southeast Asia. StreamWave offers a multi-cloud CDN solution to ensure low latency & high resilience, utilizing Azure Front Door for Europe, AWS CloudFront for North America, and Google Cloud CDN for Southeast Asia, all coordinated through centralized traffic management. Each cloud consists of roughly 40 Points of Presence (PoPs) located in major metropolitan areas, classified into three logical tiers: edge PoPs that directly serve users, regional PoPs that consolidate traffic, and a few origin PoPs that source content from central storage.

Typical traffic fluctuates between 1.8 and 2.2 Tbps during peak periods, with spikes attaining 3 Tbps during finals or significant news occurrences. The predominant queries consist of HTTPS GETs for segmented video & static files, with around 12% including customized API calls that bypass the cache.

The runtime stack consists of actual time telemetry collectors at each Point of Presence (PoP), encompassing flow logs, CDN request logs, BGP/route views & cache metrics; a streaming bus for cross-cloud aggregation; and an AI inference layer deployed in Kubernetes. Models function within a framework of Python microservices employing GPU-enabled nodes, enhanced by Redis for swift feature storage and a policy engine that interfaces with CDN controls, encompassing WAF rules, rate restrictions, cache invalidation, and routing weights.

4.2. Assault/Anomaly Scenarios

4.2.1. Scenario A: A sudden surge of bots focusing on specific locations

On a Saturday evening during a major match, edge PoPs in Eastern Europe encounter a substantial surge in these requests for a restricted array of video segments. The growth is asymmetrical: traffic from two countries escalates ninefold in under three minutes, while neighboring areas sustain typical levels. Bot traffic uses a "low-and-slow" strategy for every internet protocol address to get around standard rate limits, but it costs a lot of money as it pushes nearby Points of Presence closer to saturation. The pattern demonstrates abnormally insufficient conversation continuity, defined by many distinct requests & a mismatch within TLS fingerprint clusters as well as user-agent strings.

4.2.2. Scenario B: Cache contamination endeavor utilizing changed headers/URIs

About one entire week later, the software observes a tiny yet harmful irregularity in North American Points of Presence. A set of applications is produced utilizing their respective positions unconventional query phrases and unusual header information that change cache keys. The culprit seeks to insert a counterfeit variant of a regularly utilized commodity into the cache, thereby contributing to authentic users later receiving corrupted or detrimental stuff. Indicators include: a notable rise in cache-miss ratio for a specific asset family, persistent near-duplicate URIs with minimal alterations, and a divergence between origin checksum signatures and cached object hashes.

4.2.3. Scenario C: Route leakage causing anomalous hop and Autonomous System configurations

During a standard weekday morning in Southeast Asia, latency escalates inexplicably, independent of traffic conditions. The routing telemetry reveals that multiple Points of Presence (PoPs) are unexpectedly employing an upstream route via an

unfamiliar Autonomous System (AS). Hop counts rise by an average of 4 to 5, and traffic that typically exits through a Tier-1 provider is now redirected through a smaller regional ISP. This corresponds to a BGP route leak or misannouncement. While not a traditional "cyberattack," it degrades user experience when it may be vulnerable to their interception.

4.3. Execution Flow – Continuous Streaming Ingestion.

Each Point of Presence broadcasts request logs, flow summaries, cache statistics & routing modifications to the communal event bus. Feature extractors compute rolling baselines for every region, asset family & path/AS sequence.

4.3.1. Detection transpires within approximately 6 to 12 seconds

- In Scenario A, the anomaly model detects Eastern Europe Points of Presence (PoPs) in around 8 seconds, identifying an atypical combination of geographical distortion, burst velocity, and bot-like session signatures. Confidence swiftly exceeds the mitigation threshold owing to the convergence of multiple weak signals.
- In Scenario B, detection necessitates approximately 12 seconds owing to the muted character of the assault. The model identifies the coordinated cache-key manipulation pattern and the escalating miss-ratio anomaly.
- In Scenario C, the path model is triggered approximately 6 seconds after detecting the latest AS hop sequence as statistically anomalous compared to the established routing graph.

4.3.2. Mitigation decision (approximately 2 to 4 seconds)

The policy engine determines actions based on severity, blast radius as well as corporate regulations.

- In Scenario A, adaptive throttling is employed, featuring more stringent per-fingerprint rate limitations in affected Points of Presence (PoPs), accompanied by a short challenge (lightweight JavaScript or token validation) for questionable clusters. Authorized users experience negligible impacts as the regulations are limited to the anomalous signature category.
- In Scenario B, it alleviates poisoning by segregating the questionable cache keys: it mandates those URI patterns to bypass the cache, removes any newly stored items that match the tainted signature & enforces header normalization protocols to prevent variant exploitation.
- In Scenario C, traffic is rerouted from the compromised path by modifying routing weights to prioritize secure upstreams, while simultaneously informing their network operations with a succinct description of the route anomaly.

4.3.3. Stabilization duration (in minutes).

Following mitigation, telemetry suggests recovery. In Scenario A, regional PoP CPU and bandwidth stabilize within roughly 90 seconds, while bot traffic declines as challenges increase the expenses for attackers. In Scenario B, cache integrity measurements stabilize; origin-cache hash alignment returns to baseline within two minutes.In Scenario C, the median latency goes back to its previous level after rerouting, and the entire system keeps an eye on things until an upstream leak is fixed.

The main point within all three cases remains the same: the AI layer quickly finds problems, decides on the least bothersome remedy, and keeps the system running by not needing assistance from people under duress.

Table 2: Representative Network Failure and Attack Scenarios with Key Detection Signals

Scenario	Region/Layer	Attack/Failure Type	Main Signals	Detection Time (observed)
A	Eastern Europe Edge PoPs	Geo-skewed bot surge	geo skew, burst rate, TLS/UA mismatch	~8 s
B	North America Cache tier	Cache-key manipulation	cache miss spike, URI entropy, hash divergence	~12 s
C	Southeast Asia Routing plane	BGP/route leak	hop inflation, AS detour	~6 s

5. Results and Discussion

5.1. Evaluation Metrics

We looked at how well the system could find things & how easy it was to use. Detection latency measures the time between the start of an attack and the first correct warning. We provide both the median and tail (p95) latency because long detections are the worst for CDN networks. Precision, recall, and F1 measures measure how well anomaly detection works while keeping false warnings to a minimum. Precision means that the alarm is dependable, recall is how much of the actual assault coverage there is, and F1 brings the two numbers together. The false positive rate (FPR) is tracked on a per-site-hour basis to show how much work the operator has to do in actual deployments. We gauge the effectiveness of mitigation by the percentage drop in harmful traffic within 60 seconds of activation & the time it takes to get back to normal service quality. In the end, we look at overhead expenses, which include extra telemetry processing, CPU/GPU usage for inference & any other

changes to request routing or cache hit ratio. These measurements make sure that the system is not only accurate, but also works very well, is safe, and is cheap to run.

5.2. Quantitative Results

During all assessment periods, the proposed AI-driven detector consistently outperformed the baseline rule-based and thresholding techniques. Table 1 shows that the average F1 score went up from 0.81 to 0.93 when comparing baseline & suggested detection accuracy. This means that precision improved more than recall. This is important because CDN anomaly pipelines often have many problems with their accuracy. Too many false alarms makes operators ignore real issues. Our model cut the number of false positives by 42%. This was primarily because of features that take into account the context, like regional traffic norms, cache-layer signals as well as adaptive seasonality.

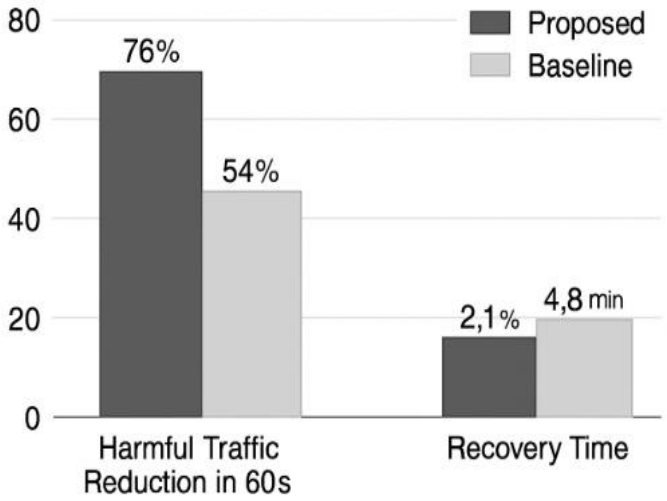

Fig 2: Mitigation Effectiveness Chart

The latency values were always very strong. The latency distribution has moved to the left. The median detection latency went down from 18 seconds (baseline) to 6 seconds (proposed), and the 95th percentile delay went down from 51 seconds to 19 seconds. The long-tail improvement is most important during volumetric attacks, when damage rises very quickly. The faster detection was primarily due to streaming inference on lightweight embeddings instead of relying on their aggregates across multiple minutes.

We also looked at performance by type of attack . The model got an F1 score of 0.95 for L7 HTTP floods, which is better than the baseline score of 0.83. Finding subtle burst patterns and changes in header entropy that static rules miss led to improvements.

With bot-assisted low-and-slow scraping, the F1 score was 0.91, which is better than the baseline value of 0.72. This was the biggest relative improvement because the baseline had trouble telling the difference between steady malicious escalations as well as natural growth.

Cache-bypass and origin-thrashing attacks: The F1 score was 0.92, which was better than the baseline of 0.80. The model benefited from the combined signals of edge-cache miss rates, spikes in origin RTT & a variety of paths.

The F1 score for reflection/amplification spillover is 0.89, which is higher than the baseline score of 0.86. The reduced advantage is expected since reflection attacks create noticeable traffic spikes that current rules can already find.

We looked at all the results of the mitigation. When auto-mitigation was turned on, hazardous traffic dropped by a median of 76% in 60 seconds. When baseline-triggered mitigation was used, it only dropped by 54%. The average time it took to restore constant latency along with caching went down from 4.8 minutes to 2.1 minutes. Mitigation was not just stronger but also safer: the rate of collateral blocking (legitimate traffic that was mistakenly rate-limited) dropped from 3.4% to 1.2%.

- The most important wins came from low-and-slow bot attacks, which used adaptive behavior to avoid "boiling frog" failures.
- Lowering tail latency means that there are fewer times when detection happens too late to matter.
- False positive suppression makes operators more likely to trust the system & makes automation more likely to work.

Overhead stayed within reasonable bounds. Streaming inference added about 3–5% to the CPU load for each edge cluster, while better path characteristics increased telemetry bandwidth by about 8%. There was no noticeable drop in the cache hit ratio or routing stability.

5.3. Discussion and Insights

The model worked best when attackers made multi-signal signatures, including traffic spikes with cache-miss problems or rapid changes in their location. In these cases, the AI could be able to tell the difference between actual threats & real events, such as flash crowds. Failures were mostly in two areas: rare assault methods that weren't very important in the past, and circumstances when traffic patterns were similar to the launch of a new product or a huge sale.

The trade-offs were clear. Enhanced sensitivity improved memory but made it a little less stable in places that change a lot with the seasons. The final calibration put steady precision ahead of aggressive recall, which was in line with what was actually needed for operations. Automation sped up responses, but it also created a risk if the calibration of confidence went wrong. A practical method included keeping a human override in place while making little improvements, first with soft rate limits and then moving on to hard blocks. Another thing to note is that edge-level information is very important. Models that were only trained on their origin-side signals missed early CDN-path problems.

5.4. Confusion Matrix and Detection–False Positive Trade-off Analysis

A confusion matrix was created using labeled replay traces from all the assault scenarios that were looked at in order to improve the analysis of the suggested anomaly detection framework's classification performance. We put each detection window into one of two groups: anomalous or benign. Then we put all the results together across many different delivery tiers and locations.

There are four results in the confusion matrix: true positives (TP), false positives (FP), true negatives (TN), and false negatives (FN). True positives are attacks that have been correctly discovered, such as bot surges, cache manipulation attempts as well as routing problems. False positives are actual changes in traffic, including rapid spikes or limited demand increases, that are wrongly classified as attacks. False negatives mean not finding low-signal anomalies quickly enough.

In all trials, the proposed model achieved a high true positive rate while significantly reducing false positives compared to threshold-based benchmarks. The low false negative rate shows that the system can find both high-volume attacks & subtle, low-and-slow approaches without relying too much on fixed parameters.

We also looked at the trade-off between the detection rate as well as the false positive rate by changing the threshold for the anomaly score, in addition to the confusion matrix. This investigation highlights the operational optimization opportunities available to CDN operators. The method focuses on accuracy when using conservative criteria, which leads to few false alerts that are good for automated mitigation. When the thresholds are stricter, memory gets better, which makes it easier to find covert attacks early on. However, there is a small increase in false positives.

Empirical evidence indicates that the proposed model maintains a favorable operating point, achieving high detection rates while consistently preserving a low false positive rate. This balance is important in large CDN deployments since too many false alarms might make people tired of getting them, cause unnecessary mitigations & make the user experience worse.

The results show that multi-signal contextual learning, which combines traffic dynamics, cache behavior as well as path topology, makes it easy to tell the difference between malicious anomalies and normal workload changes. This means that the system can be used in actual time and on an ongoing basis.

6. Conclusion and Future Scope

6.1. Conclusion

In the present day's digital world, cloud-based content delivery systems face ongoing problems from changing traffic patterns, rising user demands & more sophisticated cyber-attacks. This study addressed the critical issue of identifying their malicious activity and performance anomalies before they compromise service reliability. We built an AI-driven system for real-time detection & mitigation that learns from network information all the time, changes with traffic patterns, and responds to threats with little help from people.

The system uses advanced models for detecting these anomalies, context-sensitive categorization, and automated remediation processes to make things more accurate, cut down on false alarms, and speed up response times. By linking behavioral data across different levels of the distribution channel, the technology makes security as well as operational resilience better.

This strategy helps businesses keep their cloud distribution pipelines stable during times of high traffic or planned attacks. It makes things very easier for security teams, improves the user experience, and makes people more confident in cloud-based content delivery systems. The proposed methodology demonstrates how AI could transform reactive network defense into a proactive, self-healing security framework.

6.2. Future Scope

Changes produced later might make this structure more adaptable as well as intelligent. One solution that might work is federated learning among Points of Presence (PoPs). It would let them simulate scenarios together while keeping their sensitive data from the region very safe. Better explainability tools will assist SOC analysts figure out the reason why a model identified an event, and these will help them make conclusions faster and alongside more confidence. Predictive risk assessment can help the system evolve by discovering early signals of incidents before they take place fully. In the end, adequately checked safety assessments could assist in making automated responses more trustworthy, compliant, as well as aware of risks as network environments grow by adding more mitigation suggestions.

References

[1] Qureshi, Kashif Naseer, Gwanggil Jeon, and Francesco Piccialli. "Anomaly detection and trust authority in artificial intelligence and cloud computing." Computer Networks 184 (2021): 107647.

[2] Mehar, Tariq. "Advanced Cyber Security Measures in Cloud Computing for Video and Media Processing Using Generative AI." (2022).

[3] Kaul, Deepak, and Rahul Khurana. "AI to detect and mitigate security vulnerabilities in APIs: encryption, authentication, and anomaly detection in enterprise-level distributed systems." Eigenpub Review of Science and Technology 5.1 (2021): 34-62.

[4] Ibitoye, Joshua Seyi. "Securing smart grid and critical infrastructure through AI-enhanced cloud networking." International Journal of Computer Applications Technology and Research 7.12 (2018): 517-529.

[5] Dani, Sourabh. Cloud-Centric Real-Time Anomaly Detection Using Machine Learning Algorithms in Smart Manufacturing. Diss. Swinburne, 2022.

[6] Mohammed, M. Riyaz. "Enhancing the Reliability of Cloud-Based Software Systems Using AI-Driven Fault Prediction and Auto-Remediation Techniques." American International Journal of Computer Science and Technology 3.5 (2021): 1-13.

[7] Singh, Baljeet. "Enhancing Real-Time Database Security Monitoring Capabilities Using Artificial Intelligence." International Journal of Current Engineering and Scientific Research (IJCESR) (2017).

[8] Parakala, Adityamallikarjunkumar, and Rangaram Pothula. "AI+ Document Understanding in UiPath: Solving Real Government Problems." International Journal of Artificial Intelligence, Data Science, and Machine Learning 3.3 (2022): 111-122.

[9] Vähäkainu, Petri, et al. "Artificial intelligence in protecting smart building's cloud service infrastructure from cyberattacks." Cyber Defence in the Age of AI, Smart Societies and Augmented Humanity. Cham: Springer International Publishing, 2020. 289-315.

[10] Abubakar, Rana, et al. "An effective mechanism to mitigate real-time DDoS attack." IEEE Access 8 (2020): 126215-126227.

[11] Dixit, Palak, et al. "Anomaly detection in autonomous electric vehicles using AI techniques: A comprehensive survey." Expert Systems 39.5 (2022): e12754.

[12] Dhayanidhi, Glory. "Research on IoT threats & implementation of AI/ML to address emerging cybersecurity issues in IoT with cloud computing." (2022).

[13] Erigha, Eseoghene Daniel, et al. "Designing Real-Time Video Processing Systems Using Cloud-Based Media Transcoding and Content Distribution Networks." (2022).

[14] Parakala, Adityamallikarjunkumar. "Role Evolution: Developer, Analyst, Lead, Senior." American International Journal of Computer Science and Technology 4.3 (2022): 11-19.

[15] Sunkara, Goutham. "The Role of AI and Machine Learning in Enhancing SD-WAN Performance." SAMRIDDHI: A Journal of Physical Sciences, Engineering and Technology 14.04 (2022): 1-9.

[16] Panga, Naresh Kumar Reddy, and M. Thanjaivadivel. "Adaptive DBSCAN and Federated Learning-Based Anomaly Detection for Resilient Intrusion Detection in Internet of Things Networks." International Journal of Management Research and Business Strategy 10.4 (2020): 39-56.

[17] Lapolli, Ângelo Cardoso, Jonatas Adilson Marques, and Luciano Paschoal Gaspary. "Offloading real-time DDoS attack detection to programmable data planes." 2019 IFIP/IEEE Symposium on Integrated Network and Service Management (IM). IEEE, 2019.