MORE THAN A SIGNATURE:

Why Web Application Firewalls Aren't Enough in the Attention Economy

A TANGATE RESEARCH BRIEF

Wren's Watch LLC · tangate.com

Executive Summary

Web application firewalls have been the default answer to web security for two decades. They are good at what they were designed to do: match requests against known-bad patterns, block SQL injection, flag path traversal, reject requests from scanners that announce themselves by name. For a threat landscape defined by the OWASP Top 10 circa 2010, they are sufficient.

Of course, that threat landscape no longer exists.

Today, more than half of all internet traffic is non-human. “Bad bots” have grown as a share of traffic for six consecutive years. The dominant threat to content-rich web properties isn’t SQL injection, It’s sophisticated automated scraping by tools that run real browsers, rotate residential IP addresses, and are specifically engineered to be invisible to signature-based detection. They have no signatures to match. They look, at the HTTP layer, exactly like your best customers.

This paper examines that gap through a controlled traffic analysis comparing AWS WAF Managed Rules against multidimensional analysis across a representative mix of real-world web traffic. The results are stark: AWS WAF detects 18% of threats in this traffic model. Multidimensional analysis detects 87%.

> *The question is not whether WAF is bad. It is whether WAF alone is enough. For sites where the content itself is the asset being targeted, the answer is clearly no.*

The Traffic Reality Nobody Talks About

The industry has been measuring web traffic wrong. More specifically, the industry that charges you money to serve and measure your traffic: the barber doesn’t want you to go bald.

Security thinking has been shaped by attack categories (SQL injection, XSS, CSRF) that appear in vulnerability databases and CVE feeds. These attacks are real and must be blocked. But they are a small fraction of the hostile (or at least unwanted) traffic hitting a content-rich web property on any given day.

Imperva's 2025 Bad Bot Report, drawing on analysis of trillions of requests, found that automated traffic surpassed human activity for the first time in a decade, now accounting for 51% of all web traffic. Bad bots alone constitute 37% of all internet traffic, up from 32% the previous year: the sixth consecutive annual increase. The acceleration is directly tied to AI: large language model tools have lowered the technical barrier to bot development, putting sophisticated scraping capability within reach of non-technical operators.

The distribution matters as much as the total. In the traffic composition we analyzed, calibrated to published data from Imperva, Akamai, and Barracuda on content-heavy SaaS properties, scrapers of various types account for roughly 30% of all traffic. Traditional web attacks (SQL injection, XSS, path traversal) account for less than 3%. CMS probes and scanning, another 5-6%. The rest is either legitimate users, benign bots, or credential stuffing.

This is not an anomaly. It is the normal traffic mix for any site with valuable content.

> *A WAF built to stop the 3% while missing the 30% is solving the wrong problem.*

What Signature-Based Detection Does Well

Credit where it is due.

AWS WAF Managed Rules provide genuine, reliable protection against a well-defined class of attacks. SQL injection (including UNION-based, time-based blind, and comment-evasion techniques) is caught at 100% in our analysis. Cross-site scripting, including modern event-handler payloads, is caught at 100%. Path traversal with double-encoding and null-byte variations: 100%. Known scanning tools that identify themselves by user agent (Nikto, sqlmap, Nuclei, WPScan): caught.

This matters. Any organization running CloudFront distributions without WAF is exposed to these commodity attacks, and adding AWS WAF Managed Rules to a CloudFront distribution is a straightforward, low-cost operation. The AWSManagedRulesCommonRuleSet, SQLiRuleSet, KnownBadInputsRuleSet, and WordPressRuleSet together provide a meaningful baseline at a few dollars per month at typical traffic volumes.

AWS WAF Bot Control (Targeted mode) adds behavioral heuristics (browser interrogation, fingerprinting, and some machine learning) that improve detection of sophisticated bots beyond signature matching. The ATP (Account Takeover Prevention) rule group adds per-IP login failure rate limiting. These are genuine advances.

But all of these protections share a fundamental constraint: they operate on individual requests, in isolation, at the HTTP layer.

The Structural Blind Spot

AWS's own documentation is precise about how WAF rules work. A rule inspects an HTTP request against defined criteria: URI patterns, query string contents, header values, body content. When a request matches a pattern, it is acted upon. When it does not match any pattern, it passes.

This is the right architecture for the problem WAF was designed to solve. SQL injection payloads appear in request fields. XSS strings appear in parameters. Scanner user agents are strings in headers.

But sophisticated scrapers do not put evidence of their nature into HTTP fields. A Puppeteer instance running with stealth plugins sends requests indistinguishable from Chrome 124 on Windows 11. The user agent matches. The headers match. The TLS fingerprint can be patched to match. The request arrives from a residential IP address in Fairfax, Virginia, assigned to Cox Communications. Nothing in that request, inspected in isolation, reveals that it is automated.

The signals that reveal automation exist in three places WAF cannot see:

- **At the browser level.** Real browsers have mouse movement, keyboard events, touch patterns. Headless browsers do not. These signals differ between real browsers and headless automation, but none are visible in a server-side HTTP log.
- **Across requests.** A human user loading an article might visit three or four pages in a session, with irregular timing, following links in a non-systematic pattern. A scraper visits them all, in order, at consistent intervals, with no dwell time variation. No single request is suspicious. The pattern across hundreds is unambiguous.
- **Across IP addresses.** A credential stuffing campaign sends ten login attempts from ten different residential IPs in ten minutes. Each IP makes one failed login request. No single IP is rate-limited. The attack is only visible when login failures are correlated across the fleet.

AWS WAF Bot Control's targeted mode partially addresses the client-side signal problem through JavaScript challenges that issue a silent challenge that headless browsers may fail. This is meaningful. But it is available only in targeted mode, incurs additional per-request charges, and can be defeated by anti-detect browsers that patch the signals Bot Control checks. More fundamentally, it still cannot perform cross-request or cross-IP analysis.

Cloudflare's research team documented this precisely in their 2024 analysis of residential proxy attacks. Their ML model v8, which classifies over 17 million unique residential proxy IPs per hour, was developed specifically because traditional per-request inspection could not distinguish residential proxy traffic from legitimate residential users. The detection required new feature sets combining behavioral signals, latency patterns, and network-layer data: all outside the scope of what a request-level WAF can see.

> *The limitation is architectural. It is not a failure of execution by AWS. WAF is a tool designed for a specific job, and it does that job well. The mistake is treating it as a complete defense.*

The Analysis: Traffic Composition and Detection Rates

To quantify the gap, we analyzed a representative 24-hour traffic sample calibrated to published industry data on content-heavy SaaS properties. The traffic mix reflects Imperva, Akamai, and Barracuda research on real-world automated traffic composition.

TRAFFIC COMPOSITION

Traffic Category	Share (approx.)
Legitimate users	49%
Benign bots	9%
Sophisticated scrapers	**20%**
Primitive scrapers	5%
AI training crawlers	**5%**

Traffic Category	Share (approx.)
Scanning & recon	3%
CMS / WordPress probes	2%
SQL injection	0.5%
Cross-site scripting	1%
Path traversal	1%
Credential stuffing	1.5%
API enumeration	1.5%

Total non-human: ~50.5% · Total malicious or unwanted: ~41.5% · Scrapers of all types: ~30%

These numbers are consistent with Imperva's 2025 finding that bad bots now account for 37% of all internet traffic globally, with scraping as the dominant use case.

DETECTION RESULTS

Threat Category	AWS WAF	Tangate
SQL injection	100%	100%
Cross-site scripting	100%	100%
Path traversal	100%	100%
CMS / WordPress probes	58%	79%
Scanning & reconnaissance	81%	91%
Credential stuffing	43%	79%
API enumeration	0%	100%
Sophisticated scrapers	**0%**	**81%**
Primitive scrapers	0%	100%
AI training crawlers	0%	94%
ALL THREATS (OVERALL)	**18%**	**87%**

The headline finding: AWS WAF detects 18% of threats in this traffic model. Multidimensional analysis detects 87%.

The gap is almost entirely explained by the scraper categories. Sophisticated scrapers, primitive scrapers, and AI training crawlers together account for roughly 72% of all threats in this model. AWS WAF catches none of them, not because it has a bug or a gap in its rules, but because scrapers have no signatures to match.

THE FALSE POSITIVE QUESTION

Multidimensional analysis produces more false positives than pure signature matching: 2.05% vs 0.68% of legitimate traffic. This is an honest tradeoff, not a hidden cost. Behavioral analysis must make probabilistic judgments, and some real users will exhibit patterns that look automated. The right response is tunable confidence thresholds and a review queue, not abandonment of the approach. A 2% false positive rate with 87% threat coverage is a better security posture than a 0.7% false positive rate with 18% coverage.

Why Scrapers Are the Dominant Threat for Content-Rich Properties

For a financial services site, the dominant threat may be credential stuffing and account takeover. For an e-commerce site, it may be inventory scraping and carding. For a content-rich SaaS property (documentation, news, research, API data), the threat model is fundamentally different.

> *The asset being targeted is the content itself.*

Sophisticated scrapers are not looking for database credentials or session tokens. They are systematically extracting the content that took significant resources to create, for competitive intelligence, training data, or redistribution. The scraper visiting your documentation at 3am from a residential IP in Dallas is not trying to inject SQL. It is trying to replicate your knowledge base.

Cloudflare's data on AI crawler growth illustrates the scale: from July 2024 to July 2025, requests from GPTBot rose 147%, while Meta-ExternalAgent requests rose 843%. And these are the declared crawlers: the ones that announce themselves by user agent. The undeclared crawlers, running Chromium with stealth plugins through residential proxies, do not appear in those counts at all.

The economics reinforce the picture. Analysis from the IAB Tech Lab estimates that AI-powered search summaries reduce publisher traffic by 20–60% on average. The mechanism is straightforward: content scraped to train a model returns to users as AI-generated summaries, eliminating the click to the original source. The content creator's business model is hollowed out, one scraper visit at a time, while their infrastructure serves every one of those visits at full cost.

For a property where content is the business, allowing scraping to proceed undetected is not a neutral outcome. It is a slow transfer of value from creator to extractor.

Four Detection Capabilities That Require Multidimensional Analysis

The following capabilities cannot be replicated by request-level pattern matching, regardless of how sophisticated the rule set:

1. Session trajectory analysis

Automated sessions cover content systematically, with consistent inter-request timing, high page coverage, and no behavioral variation. Human sessions are irregular, interest-driven, and exhibit natural dwell time patterns. Identifying this requires observing a session over time, not evaluating a single request. An access log provides this view; a per-request WAF rule does not.

2. Query parameter semantic analysis

Sequential pagination, high-limit API requests, and systematic sort/filter enumeration are scraper fingerprints that appear in aggregate log data. No individual request reveals intent; the pattern across dozens does. A limit=500&page=7 request from a user agent that otherwise looks like Chrome is unremarkable in isolation; the same IP making page=1, 2, 3 ... in sequence at consistent intervals is unambiguous.

3. Cross-IP credential stuffing correlation

A credential stuffing campaign that distributes login attempts across a residential proxy pool will never trigger per-IP rate limiting. Each IP makes one or two attempts, below any sensible threshold. Fleet-wide failure correlation, tracking aggregate login failure rates across all IPs in a time window, surfaces the campaign from signals that are invisible per-IP.

4. API resource enumeration

Sequential access to incrementing resource identifiers (invoice numbers, user IDs, order numbers) is a clear signal of automated enumeration. Detecting it requires cross-request state: tracking that the same IP has accessed resources INV-08033, INV-08034, INV-08035 in sequence. A WAF evaluating each request in isolation sees three unremarkable GET requests.

These capabilities share a common requirement: persistent state across requests and a view of traffic patterns over time. They are architecturally incompatible with stateless per-request inspection, regardless of how sophisticated the rule set.

The Honest Limits of Log-Based Analysis

Multidimensional analysis working from server-side logs is not a complete solution. Acknowledging that is important.

The most sophisticated scraping operations are specifically engineered to be invisible in access logs. They use residential proxy pools where each IP appears only once or twice, defeating any per-IP rate or pattern analysis. They execute JavaScript fully, load assets selectively to mimic human behavior, and vary request timing to avoid consistent interval patterns. In a log-based analysis, these sessions may be indistinguishable from legitimate users.

Our analysis models this honestly: approximately 15-20% of sophisticated scrapers in this dataset use evasion techniques that server-side analysis cannot reliably catch. Human-in-the-loop scraping, where real people handle occasional CAPTCHAs or anomaly responses, is another category that log analysis alone cannot address.

Closing that gap entirely requires what log-based systems cannot provide: client-side signals. Tools like AWS WAF Bot Control (Targeted mode), Cloudflare Bot Management, and dedicated bot detection platforms issue JavaScript challenges that probe browser behavior. Cloudflare's ML model v8 combines these signals with latency-based and network-layer data to classify over

17 million residential proxy IPs per hour. These capabilities cannot be replicated by a server-side log analyzer working from CloudFront access records.

What log-based multidimensional analysis can do: catch the majority of scrapers who are not running fully patched anti-detect stacks, and most operators are not. The primitive scraper running python-requests is caught by user agent analysis. The AI training crawler on datacenter IPs is caught by IP classification plus crawl pattern analysis. The credential stuffing campaign is caught by cross-IP failure correlation. The API enumerator is caught by sequential resource access patterns.

> *The ceiling for this approach is real: it does not reach 100%, and it will not close the gap on the most evasive actors without client-side augmentation. What it does is remove the large, addressable portion of the threat landscape that signature-only detection leaves entirely untouched.*

Implications for CloudFront Deployments

Organizations running content-rich SaaS properties on CloudFront face a specific architectural question: where does behavioral analysis live?

AWS WAF is well-integrated with CloudFront and simple to deploy. It is the right starting point, and its managed rule groups provide the signature-based baseline that no deployment should be without. Bot Control's targeted mode extends this with some behavioral capability.

The gap is the cross-request analysis that WAF cannot perform by design. Closing the majority of it requires:

- **Session state:** A persistent store that accumulates request patterns across a session and across IPs, enabling trajectory and enumeration analysis.
- **AI-assisted pattern recognition:** Log analysis that identifies scraper behavior from combinations of signals (request sequences, query parameter patterns, user agent anomalies, timing) that no single rule captures.
- **Fleet-wide aggregation:** A view across all sessions and all IPs to surface distributed credential stuffing and systematic enumeration.
- **Threat intelligence:** IP reputation data, known-bad user agents, and botnet C2 indicators updated continuously from open-source feeds and operator telemetry.

Closing the remaining gap on the most evasive actors requires client-side augmentation: JavaScript challenges that probe browser-level signals invisible to server-side logs. AWS WAF Bot Control Targeted mode provides some of this. The right architecture layers these capabilities rather than choosing between them.

Conclusion

Web application firewalls were designed for a threat model defined by signatures: known-bad strings in known locations. That model covered the dominant threats of the early web, and it still covers a real class of attacks today.

But the dominant threats to content-rich web properties in 2025 are sophisticated automated scrapers that have no signatures: they run real browsers, use residential proxy networks, and are architecturally engineered to look like legitimate users. Imperva's 2025 research places bad bot traffic at 37% of all internet traffic, up for the sixth consecutive year, with AI-driven scraping responsible for significant growth. The content these bots extract is the same content that funds the organizations they are targeting.

Signature-based detection catches 18% of this threat environment. Multidimensional analysis (cross-request pattern detection, AI-assisted log analysis, fleet-wide correlation, and continuously updated threat intelligence) raises that to 87%.

The remaining 13% is the honest limit of log-based detection: the most evasive actors, anti-detect browsers, and human-in-the-loop operations. Closing that gap entirely is not a realistic goal. Closing the gap from 18% to 87% is.

> *For organizations where content is the asset, that gap is the business.*

ABOUT TANGATE

Tangate is a CloudFront security platform that deploys directly into your AWS account. It combines Lambda@Edge enforcement, AI-powered log analysis, cross-request pattern detection, and nightly-updated threat intelligence in a serverless architecture that requires no data to leave your infrastructure. Tangate does not replace AWS WAF. It extends it with the cross-request and AI-assisted analysis that WAF was not designed to perform.