IONIX

State of the WAF

October 2026

How often web application firewalls block attacks on live, internet-facing applications: an external measurement across hundreds of organizations

IONIX Research. Published October 2026.

Foreword

Ask a security leader whether their web applications are protected and the answer is usually yes, followed by the name of a web application firewall (WAF) vendor. That answer describes what was purchased. It says little about what happens when an attack arrives.

We built this study to measure exactly that. We sent attack traffic to live applications from the internet and recorded what was refused. The results challenge assumptions that are common in security programs and board reporting: that a WAF in the inventory covers every application, that a firewall bundled with a CDN or cloud platform is switched on, that a WAF blocking some attacks blocks the ones that matter, and that every request passes through it. In practice, applications on subdomains and acquired properties often sit outside the WAF program, bundled modules are often left in logging mode, default rule sets can miss common injection attacks, and origin servers can be reached directly.

These gaps come from how estates grow and how platforms ship, and most cannot be seen from inside a single WAF console. This report gives WAF teams a peer view they can check against their own environment, and gives security leaders evidence of what was observed from outside. Most of the fixes are configuration changes on platforms organizations already use. The first step is knowing where to make them.

Marc Gaffan
CEO, IONIX

Key Finding

In a typical organization, half of live web applications that should be behind a WAF have nothing blocking attacks

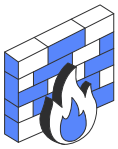
Across all organizations, **47.3%** of live, internet-facing web applications have no blocking firewall in front of them. Most live web applications sit away from the main website: **78.6%** are on subdomains rather than the main domain.

IONIX , State of the WAF 2026 IONIX

Executive summary

Web application firewalls are widely deployed, but in the median organization half of live web applications have nothing that blocks an attack. IONIX Research sent five attack probes to each of the live web applications across hundreds of organizations and recorded what was refused. The data suggests the gaps come mostly from coverage and default configuration, rather than from WAF technology.

Missing WAF Coverage



47% of live web applications have no firewall blocking attacks. In the median organization the figure is 50%. We found that 79% of live web applications sit on subdomains rather than the main domain.

Unprotected Login Pages

47% of live login pages do not have a WAF blocking any attack probe.

WAF Present, but not effective

On hosting that includes a WAF capability, **44% of applications have nothing blocking attacks.**

Bypassing WAF Completely

Of WAF-protected applications with an origin we could test, **29% answer requests sent directly to the origin**, and **11%** serve the full application that way. **81%** of those bypasses needed only the correct Host header.

Unblocked SQL Injection

Among applications behind a blocking WAF, **38% let our SQL injection probe through.** Dedicated WAF products blocked it on **97%** of their applications; WAF modules bundled into CDN and cloud platforms blocked it on **45%.**

Protection Varies 2x by Operator

Externally operated applications are more than twice as likely as organization-operated ones to have a blocking WAF: **40% vs. 18%.**

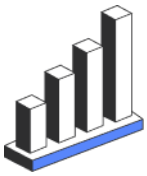
What this means for defenders

A WAF listed in an inventory or a contract does not show that applications are protected. Coverage should be measured against a complete list of live web applications, including subdomains and applications run by third parties, and blocking should be verified from outside on a schedule. Most of the fixes are configuration changes on platforms organizations already use: moving WAF modules from logging mode to blocking, activating SQL injection and file-inclusion rules, and restricting origins to the WAF provider's address ranges.

"Behind the WAF" no longer means protected

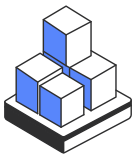
Most organizations with a public web presence have a WAF somewhere in their estate, whether they bought it, inherited it through an acquisition or received it as part of a CDN or cloud platform. Security programs often record that as coverage: the application is listed as "behind the WAF" and the control is marked as in place for the auditor. This report asks a different question: when an attack arrives from the internet, does anything stop it?

Three changes make the gap between having a WAF and blocking attacks more consequential:



1. Internet-facing estates have grown:

Business units, agencies, and SaaS vendors spin up internet-facing applications continuously, and many do not pass through a security review.



2. WAFs increasingly ship as bundled modules:

Arrive inside CDNs, cloud load balancers and hosting platforms as modules that may or may not be switched on, and that may block one attack class while passing another.

3. Exploitation is faster:

The interval between a web vulnerability being published and being exploited at scale is now routinely measured in days, and a firewall that is switched on and blocking is often the only control that can respond inside that window.

The deceptive nature of WAF Coverage

When most organizations say that something "has a WAF", they mean there is some kind of security control in the request path that inspects HTTP traffic and can refuse a request because of its content. This functionality can come from dedicated WAF services, the security modules of CDNs and cloud platforms, appliances, and open-source engines running on the web server itself. But the status of a WAF, and how it actually behaves in practice, can be far more nuanced. It can be present but in logging mode, where it logs attacks without blocking them. It can block one attack class and pass another. It can protect the main domain while the login page sits on a subdomain or different path that escaped protection It can be perfectly configured and still be bypassed by a request sent straight to the origin.

Because of that, this report does not ask whether a WAF is present. Inventories, fingerprints and vendor dashboards already answer that, and as Finding 06 shows they answer generously. Instead, every valid web application was sent five attack probes (cross-site scripting, SQL injection, local file inclusion, remote code execution and XML external entity) alongside a benign baseline request, and we recorded what was actually refused. A firewall counts as protecting an application only when it demonstrably blocked at least one attack. Everything else is counted as unprotected, whether or not a product is visible in the response.

Full details of how we collected the data, and how to interpret it, are in Appendix A.

The Coverage Gaps: How many live web applications have a WAF that blocks attacks

We measured how many applications that should be behind a WAF have nothing blocking an attack: across the whole estate, on login pages and on hosting that already includes a WAF capability.

47% of live web applications have no firewall blocking attacks

Nearly half of valid web applications have no blocking firewall in front of them. Every application counted here is live, serving real content over HTTPS, not parked and showing no signs of abandonment.

The median organization sits at **49.5%.** Because the population figure and the median are close, the result is not driven by a few very large estates.

Finding 01

47%of valid web applications have nothing blocking attacks

IONIX , State of the WAF 2026 IONIX

Valid web applications (live, content-serving HTTPS applications), by what happened to our attack probes. 28.9% blocks, vendor named; 23.8% blocks, vendor not identified; **47.3% nothing blocks**. Applications that block but cannot be attributed to a named product are counted as protected, the conservative reading.

How the 47% relates

Finding 03 narrows this to applications already sitting on hosting that ships a firewall: there, 44.4% have nothing blocking attacks. Applications on hosting with no firewall module at all fare worse, at 62.4% unprotected. Weighted together, the two populations give the 47.3% above.

In a median estate, roughly 19 of every 100 hostnames are live web applications worth protecting, and, at the 47.3% rate above, about 9 of them have nothing blocking.

Most of these applications sit away from the main domain: 78.6% are on subdomains, and among applications with a login page the share is 92.2%. Fewer than one in four valid web applications is on the main domain. We assess that this is a main reason coverage lags: WAF deployments usually start with the primary website, and most of the estate that needs protection sits elsewhere.

There are two limits here. A firewall running in logging mode, or with rules scoped to paths we do not probe, is indistinguishable from none. And externally operated applications make up the large majority of this population, so the figure is largely a statement about applications that third parties run on the organization's behalf, and for which the organization remains accountable.

What this means for security leaders

In the median organization, about half of the applications that should have a WAF have nothing blocking attacks, and most live applications sit on subdomains that an inventory built around the main website will miss.

What to do

See the first recommendation in each audience list (set a coverage policy; build the inventory from outside the WAF console).

Finding 02

47.4% of live login pages show no firewall blocking

IONIX , State of the WAF 2026 IONIX

Login pages are where credential stuffing, brute-force and injection attempts are directed. **47.4% of live login pages show no firewall blocking.** Only 18.8% sit behind a firewall we can affirmatively name as blocking.

In the median organization, 47.1% of login pages are unprotected. Organizations that protect none of theirs do exist, but they are rare once very small estates are set aside: among organizations with at least 20 login pages, it is roughly one in sixty.

Basic-auth login pages carry a different risk. HTTP basic authentication is the browser's built-in username-and-password prompt. The scheme has no native support for multi-factor authentication, rate limiting or account lockout, so any defense against password guessing depends on the server or on a control in front of it. Credentials are base64-encoded rather than encrypted, so they rely entirely on HTTPS for protection in transit. Basic-auth pages are unprotected less often than login pages overall (33.7% against 47.4%), but on a third of them, nothing blocked our attack probes.

47.4%

of live login pages have no firewall blocking

Of live login pages

33.7%

of HTTP basic-auth pages have no firewall blocking in front of them

Of login pages using HTTP basic authentication

Detection is per hostname and probed at the site root, not at the login path, so a WAF with rules on /login but not on / is recorded as absent. The figure therefore means “the host shows no WAF blocking”, not “the login form is unfiltered.” We assess path-only protection as uncommon because WAF attack rules are usually applied site-wide, but we did not measure it. Login-specific controls such as rate limiting, bot management, and account lockout were outside the scope of this test.

What this means for security leaders

Nearly half of login pages have no WAF blocking at the host, and single sign-on pages, which protect access to everything else, are the least protected.

What to do

See “Put identity on the list” in the recommendations for WAF teams.

Finding 03

44.4% of applications have a WAF available on their hosting platform, but nothing blocking attacks

IONIX , State of the WAF 2026 IONIX

Roughly five in six valid web applications sit behind a platform that ships a firewall. On nearly half of them (44.4%), nothing blocks an attack. On these applications, the WAF capability is already in the request path and handling live traffic. Two causes are common in practice: the module was enabled in logging mode during rollout and never switched to blocking, or the WAF policy covers the apex domain but not the subdomains under it.

The pattern follows the commercial model more than the technology. Where the WAF is the product, it is usually enabled. Where it is an add-on to a general-purpose cloud platform, it mostly is not (see Finding 04 for SQL injection block rates by commercial model).

The cloud platform figures show the same pattern. Across applications hosted on four of the major public cloud platforms, 52.8% have no firewall blocking at all, from either the platform’s own module or any other product in front of it. For these applications, a WAF is a configuration change away.

44% of applications on WAF-capable hosting have nothing blocking attack

33.7% proven blocking, vendor named, 21.9% blocks, vendor unidentifiable, **44.4% nothing blocks**

52.8%

of cloud-platform applications have no firewall blocking at all — not the platform's own module, not anyone else's

46%

of the median organization's WAF-capable applications have no firewall blocking

62.4%

unprotected on hosting with no firewall module at all

Applications on each cloud platform with no firewall blocking at all

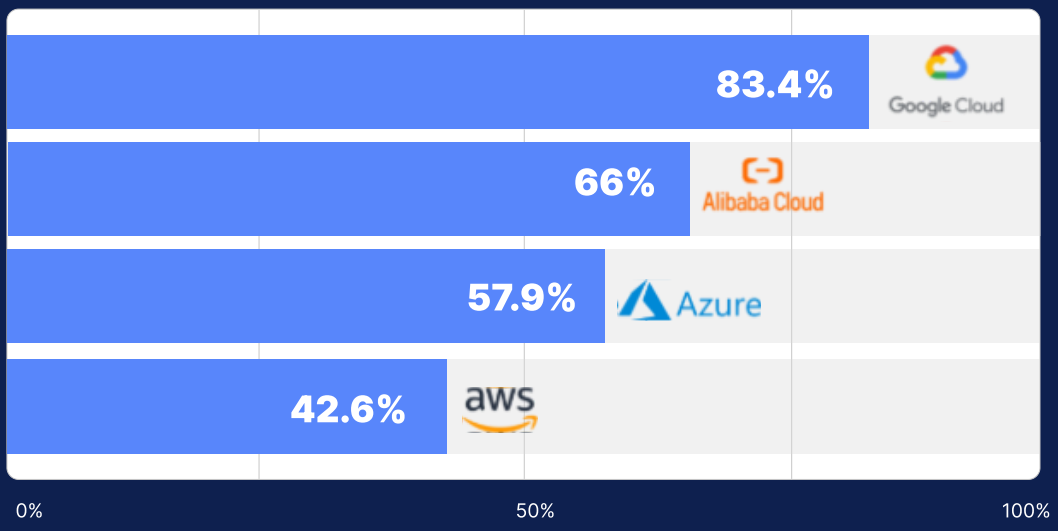


Figure 2

Source: IONIX Research, State of the WAF 2026.

Limits of the cloud platform measurement

Blocking is established by comparing each attack response against a healthy baseline for the same URL, so a denial that carries no vendor string (or no body at all) is still recorded as a block. The share of applications with nothing blocking at all is therefore measured for every platform, not inferred from whether a block page could be attributed to a product.

What external measurement cannot settle is the other side of the same numbers. Among the minority on each platform that do block, we cannot always tell whether the platform's own firewall did the blocking or something else sitting in front of it. That is a limit on attribution between products, not on the unprotected share.

The headline unprotected rate is reported twice for the same reason: 44.4% overall, 38.8% with the highest-rate platform's applications set aside. The second figure is a robustness check showing the finding does not rest on any single platform.

From inside the organization the check is cheap. Every major platform exposes, per distribution, load balancer or gateway, whether a firewall policy is attached and whether its rules are in logging mode or block mode. The difficult step is building the list of applications to check.

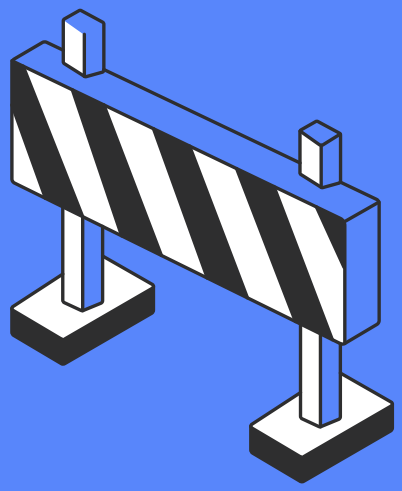
What this means for security leaders

On 44.4% of applications where the hosting already includes a WAF, nothing blocks attacks. Closing that gap is a configuration change on a platform the organization already pays for.

What to do

See “Review WAF modules on the largest platform” in the recommendations for CISOs and “Set an exit date for logging mode” in the recommendations for WAF teams.

The Configuration Gaps: What blocking WAFs miss and how protection varies across organizations



This part of the report looks at applications behind a blocking WAF, meaning a WAF that refused at least one of our five probes. It covers which attack classes those WAFs let through and whether the origin behind them answers requests that skip the WAF. It then moves to the organization level: how many WAF vendors an estate appears to run compared with how many are proven to block, and how blocking differs between applications the organization runs itself and those that third parties run on its behalf.

Finding 04

38% of applications behind a blocking WAF let our SQL injection probe through

IONIX , State of the WAF 2026 IONIX

Every application in this finding has a WAF that blocked at least one of our five probes. The question is which attack classes it blocks.

Injection, the category that includes SQL injection, has appeared in every edition of the OWASP Top 10, yet it is the attack class that most often got through. Two in five apps behind a blocking WAF passed our SQL injection probe.

XML external entity (XXE) injection, a less common attack class, is blocked on 96.7% of these applications. One likely reason is that an XML-shaped payload is easier to match than one that has to be judged in context; the caveat below on how the XXE probe was delivered offers another.

The result depends on the commercial model. Grouped by commercial model rather than brand, firewalls sold as security products block the SQL injection probe on **97.0%** of their applications. Firewall modules bundled into a CDN or cloud platform the organization bought for delivery or hosting reasons block it on **45.0%**. The all-population figure lands at 61.6% because bundled modules carry roughly two thirds of the applications we could attribute to a product. We read the SQL injection figure as a measure of what is enabled by default rather than of what WAF technology can do: on more than one bundled platform, the rule group that covers SQL injection is a separate item from the baseline rule set and has to be attached deliberately.

Partial protection is common, and it does not show on a dashboard. A firewall that blocks traffic all day, reports healthy, and quietly passes SQLi looks identical to one that blocks everything.

One gap pattern alone (blocks XSS, RCE and XXE but misses both SQLi and LFI) covers **23.7%** of all WAF-protected apps we measured. A pattern shared by nearly a quarter of applications points to default rule sets rather than individually misconfigured deployments.

SQL injection is blocked least often of the five attack classes

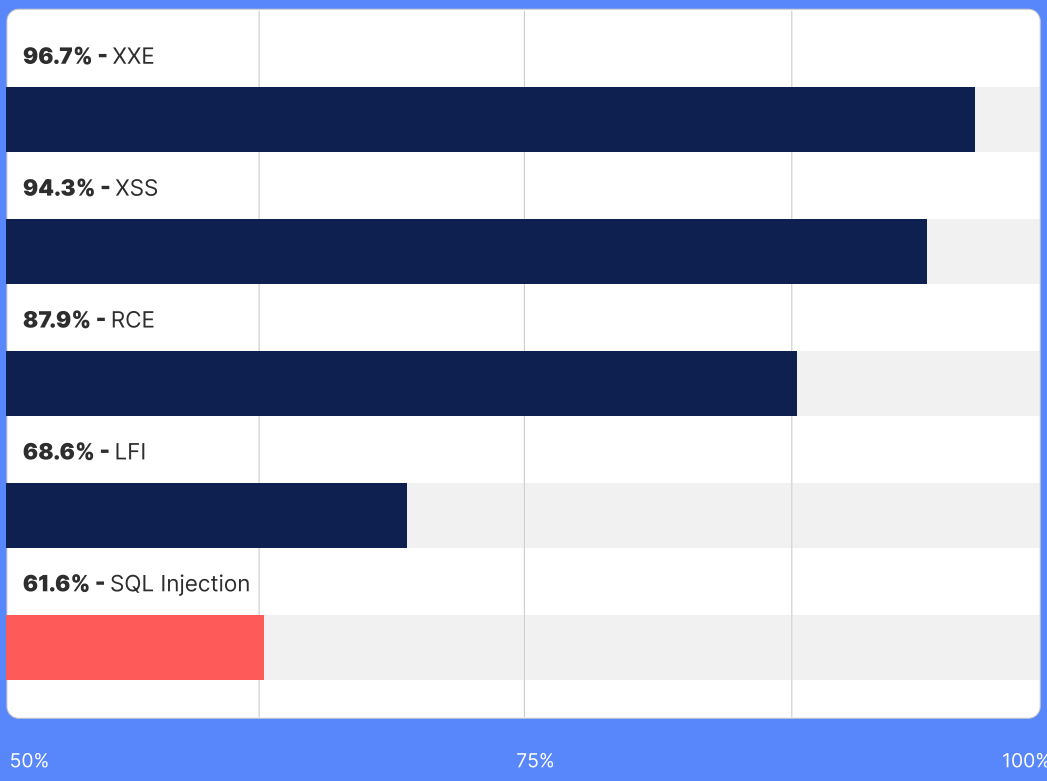


Figure 3

Source: IONIX Research, State of the WAF 2026.

51.7%

of applications behind a blocking WAF had all five probes blocked

Of firewalls that block something

38.4%

miss SQL injection while blocking something else

Of firewalls that block something

23.7%

share one identical gap: XSS+RCE+XXE blocked, SQLi and LFI missed

Of firewalls that block something

One caveat on reading these: each class is exercised with a single probe, so “did not block SQL injection” means it did not block that probe. A ruleset may treat a given request as low-confidence in the context it arrived in while still stopping real exploitation. The XXE probe in particular is delivered in a form a genuine attack would not use, which likely explains why it scores highest of the five. This caveat has a limit. The 23.7% gap pattern above is a firewall that blocks cross-site scripting, remote code execution and XXE on the same application, so it is demonstrably in the request path and evaluating those requests, and still lets the SQL injection payload through. The measurement therefore reflects rule sensitivity to a deliberately plain payload rather than the absence of a WAF.

What this means for security leaders

A WAF that reports healthy can still pass common attacks. On bundled platforms, SQL injection protection often has to be switched on separately.

What to do

See “Check rule set coverage per attack class on bundled platforms” in the recommendations for WAF teams.

Finding 05

29% of WAF-protected applications answer requests sent directly to the origin, and 11% serve the full application

IONIX , State of the WAF 2026 IONIX

A WAF only protects an application if every route to it passes through the WAF. Nearly a third of the protected apps we could test can be reached directly at their origin server.

Of apps with a proven blocking firewall and an origin we could test, 29.4% answer a request sent straight to the origin, without passing through the WAF. For 10.9% the origin serves the identical application, which is a complete, confirmed bypass of the WAF.

The remaining 18.6% return something different at the origin such as a health check, a maintenance page or an admin interface. These responses do not expose the protected application, but they reveal the origin’s real IP address and sidestep CDN-level rate limiting and DDoS absorption.

The bypass technique was simple. 80.5% of successful bypasses needed only a request to the origin address carrying the correct Host header. The remaining 19.5% needed a spoofed client-IP header. Published bypass research tends to focus on this second technique, but it accounts for barely a fifth of the bypasses we observed.

How the origin bypass works

The origin server answers a direct request that simply names the site it is hosting, and returns the live application unfiltered. On confirmed bypasses the response taken straight from the origin was typically indistinguishable from the one served through the firewall: the same application, with nothing inspecting the traffic.

81% of confirmed origin bypasses needed only the correct Host header

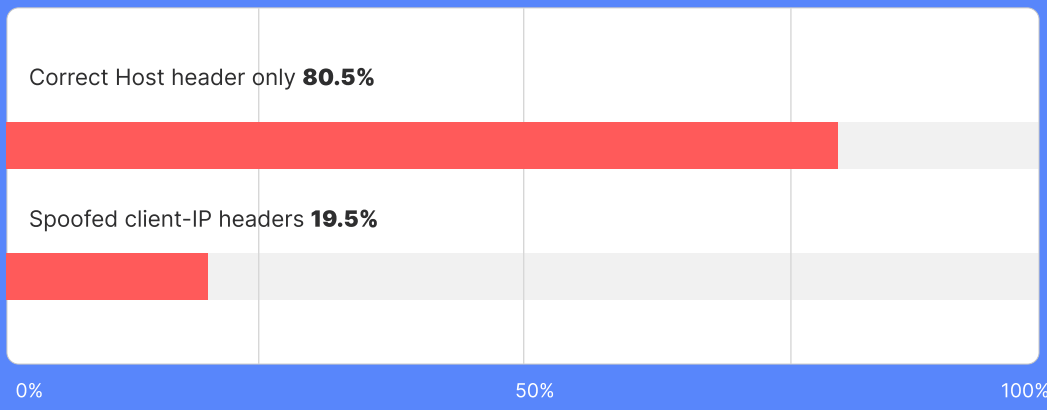


Figure 4

Source: IONIX Research, State of the WAF 2026.

Each application carries one recorded attempt, so these are the share of bypasses each approach opened. They are not an efficacy test between them.

The 29.4% counts bypass findings currently open against an application, and some of those have not been re-confirmed by a recent scan. The origin-bypass measurement above also covers only organizations that connected their firewall vendor's API, since that is what tells us where the origin is. It is therefore a biased sample, and the bias likely runs against the finding: organizations that set up an integration would be expected, if anything, to be better configured than average.

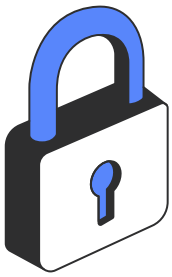
The fix is well known and vendor-neutral: restrict the origin to the CDN or WAF provider's published address ranges, require authenticated origin pulls or mutual TLS where the platform supports it, and treat any direct answer from the origin as a finding to be closed. Origin lockdown is often deferred because it requires changes by the application owner, not only by the WAF team.

What this means for security leaders

For 10.9% of the protected applications we could test, the WAF can be bypassed by anyone who finds the origin address, which makes the WAF spend on those applications ineffective.

What to do

See “Lock down the origin” in the recommendations for WAF teams.



Finding 06

Fingerprints overstate WAF coverage: The median organization has 11 WAF or CDN vendors fingerprinted and 3 proven to block

IONIX , State of the WAF 2026 IONIX

Counting firewall vendors by fingerprint suggests a crowded, well-defended estate. Counting only those that provably block an attack gives a much smaller number.

The median organization has 11 distinct firewall or CDN vendors fingerprinted across its valid web applications. The median number proven to block an attack is 3. Both counts are taken over the same population, so the gap is real, but it is mostly a gap in evidence rather than in enforcement. Roughly four fifths of it is vendors for which no external block-page evidence exists either way; only about a fifth is vendors that were testable and demonstrably did not block. Counts here are restricted to organizations with at least 20 valid web applications, since on a very small estate proving zero vendors is close to a sampling artifact.

The distribution matters more than the median. One organization in nine proves none at all: no vendor anywhere in the estate blocked one of our probes. A further 15% prove exactly one, so 26% prove one or fewer. At the other end a small group proves ten or more, which is its own operational problem: each additional vendor adds a console and a rule syntax where policy can drift.

11

WAF or CDN vendors fingerprinted, median organization

Present in the request path

3

proven to block an attack, median organization

Demonstrated, not inferred

11%

of organizations prove no blocking vendor at all

Across the whole estate

Vendors counted per organization across its qualified valid web applications, restricted to organizations with at least 20 valid applications. A vendor counts as proven only where a signature matched inside a response that actually blocked an attack.

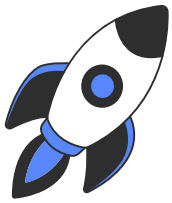
The gap between 11 and 3 helps explain why coverage figures derived from fingerprinting run ahead of what is enforced. A vendor that can be named from a fingerprint may or may not have a rule engine enabled behind it. Nearly half the fingerprinted labels are CDNs, proxies, platforms or unknown-vendor placeholders rather than enforcement products; counting only plausible enforcement products, the median is about 6. None of this shows the missing vendors are switched off. It shows that presence is not evidence of blocking, which is why every other figure in this report is built on observed blocking.

What this means for security leaders

A list of WAF vendors in an inventory, an audit response or an insurance questionnaire does not show which applications are protected. Only a test of blocking does.

What to do

See Recommendations for CISOs.



Finding 07

Externally operated applications are almost twice as likely to have a blocking WAF: 40% vs. 18%

IONIX , State of the WAF 2026 IONIX

Externally operated applications are more than twice as likely to sit behind a proven blocking firewall as the ones the organization runs itself. Where a team deliberately buys a dedicated WAF, it configures it thoroughly: on one such platform, organization-operated applications reach 97.5% proven-blocking coverage, against 72% for externally operated applications on the same platform. We assess the low aggregate figure below as a portfolio effect. More than half of all organization-operated applications sit on self-serve cloud and CDN platforms where the WAF is an opt-in feature that was not enabled, and those applications outnumber the well-configured deployments.

A common assumption is that third-party-operated assets are the weak point: shadow IT, forgotten agency microsites and vendors outside the security team's control. In this data they are better protected: 40.2% of externally- operated apps have a proven blocking WAF, against 18.4% of organization-operated ones.

We tested whether composition explains it, for example whether externally operated assets land on better platforms. It does not. Both groups sit on WAF-capable hosting at almost exactly the same rate (86.0% and 87.6%). The gap is in activation.

We also tested whether it is a measurement artifact, for example organizations running enterprise WAFs our signatures cannot name. Counting every unattributable block as protection leaves the gap intact.

What the decomposition does not tell us is why. The most plausible explanation is mundane: externally operated assets are largely SaaS and managed hosting that ship a firewall as part of the product, so protection is inherited rather than decided, while organization-operated assets get whatever someone actively configured. We measured the gap, not that explanation.

Both groups sit on WAF-capable hosting at similar rates; externally operated applications are 2.14× as likely to have a blocking WAF there

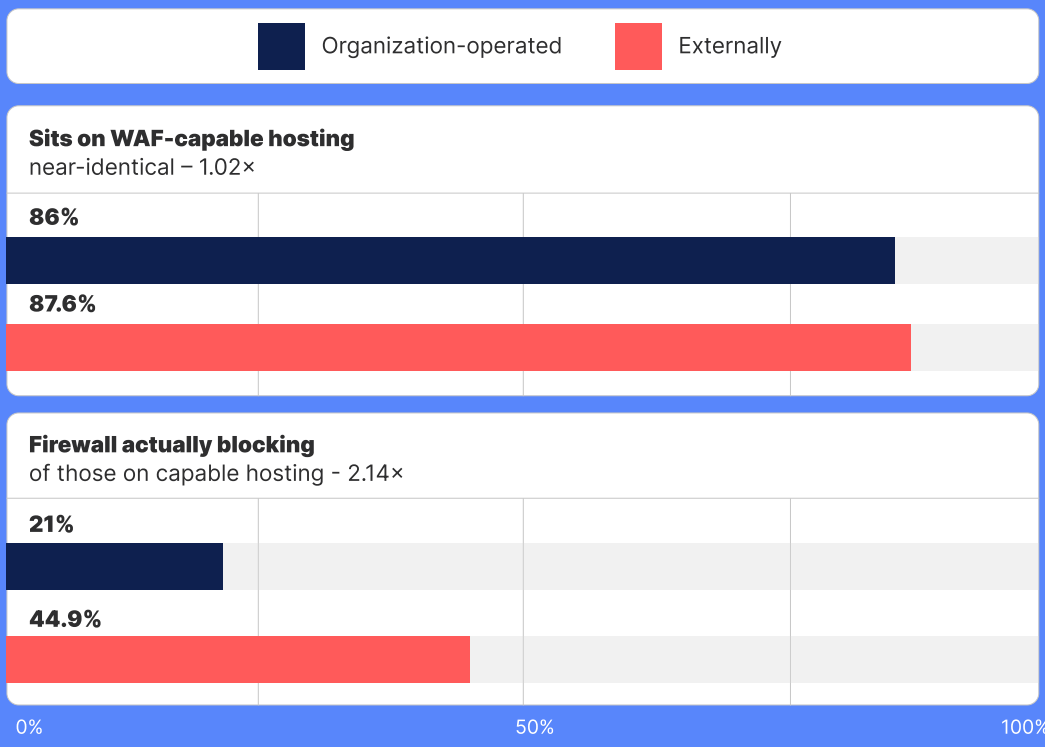


Figure 6

Source: IONIX Research, State of the WAF 2026.

The practical conclusion is that **the default state on an organization's largest platform is often unprotected.** Where the WAF is the product, it is usually switched on and tuned. Where it came bundled with a platform, it mostly is not.

What this means for security leaders

The applications an organization's own teams run are the least protected, mostly because the platforms they use do not enable a WAF by default. The fix is a platform default and an owner, not more tooling.

What to do

See "Review WAF modules on the largest platform" and "Assign an owner to every internet-facing application" in the recommendations for CISOs.

Key findings at a glance

Across the organizations analyzed:

Half of live web applications have nothing blocking attacks.

47% of live login pages do not have a WAF blocking an attack probe. SSO-backed login pages are the least protected: **58%** have no firewall blocking attacks, compared with **46%** of other login pages.

11% of WAF-protected applications with a testable origin serve the full application directly from the origin, bypassing the WAF. **80%** of these bypasses needed only the correct Host header.

44% of applications on hosting that includes a WAF have nothing blocking attacks.

The median organization has 11 WAF or CDN vendors fingerprinted but only 3 proven to block an attack.

38% of applications behind a blocking WAF let a deliberately plain SQL injection probe through.

Externally operated applications are almost twice as likely to have a blocking WAF as organization-operated ones: **40% vs. 18.%.**

Recommendations

For CISOs and security leaders

Set a coverage policy and report against it.

Decide which applications must have a blocking WAF (for most organizations, every live web application) and report coverage as the share of those applications with verified blocking, not the share listed as protected. In the median organization in this study, about half would fall short. This is the one WAF metric worth taking to the board. (Findings 01, 06)

Assign an owner to every internet-facing application.

Include applications run by business units, subsidiaries, acquired companies and third parties. In large enterprises, ownership is usually the reason a gap stays open; in mid-size organizations, it is usually staff time, so the owner may be the same person for every application. (Findings 01, 07)

Ask for evidence of blocking in audits and questionnaires.

PCI DSS 4.0.1 requirement 6.4.2 calls for an automated technical solution in front of public-facing web applications, configured either to block web-based attacks or to generate an alert that is immediately investigated. A WAF in logging mode whose alerts are not investigated immediately, or a WAF on the wrong hostname, may not meet that bar. The same applies to cyber insurance questions that ask only whether a WAF is deployed. (Findings 03, 06)

Review WAF modules on the largest platform.

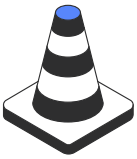
Find out whether the WAF included with your main cloud or CDN platform is enabled in blocking mode, and on which properties. For many organizations this closes more of the gap than any new purchase. (Findings 03, 04, 07)

For WAF and application security teams

Build the inventory from outside the WAF console.

List every live web application, including subdomains and third-party-operated properties: 78.6% of what needs protecting is not on the main domain. Decide explicitly which applications should be behind a WAF (for most organizations the policy is “all of them”) and measure coverage against that list, not against what the WAF console knows about. (Finding 01)

Prove blocking from outside, on a schedule.



Console state is not evidence. Send a benign request and a handful of attack-shaped requests to every application on the list and record what is refused. Done regularly, this catches rules left in logging mode, path exclusions, silently degraded configurations and newly deployed applications alike. (Findings 03, 04)

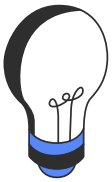
Check rule set coverage per attack class on bundled platforms.

If your largest platform ships its firewall as a module, confirm that the SQL injection and file-inclusion rule groups are attached and in block mode, not only the baseline set. This is the single configuration most likely to explain a firewall that blocks XSS and passes SQLi. (Finding 04)

Lock down the origin.

Restrict origin servers to the CDN or WAF provider’s address ranges, use authenticated origin pulls or mutual TLS where the platform supports it, and make “the origin answered a direct request” a tracked finding. Apply the same firewall on HTTP as on HTTPS, or redirect. (Finding 05)

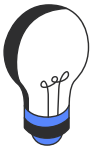
A hygiene item



If your firewall names itself in its block page, consider turning that off. It costs nothing and removes an attacker’s starting point (see the supplementary finding below).

Set an exit date for logging mode.

When a rule set goes live in logging mode for tuning, record the date it should move to blocking and review false positives against that date, so that “for now” does not become permanent. (Finding 03)



And a hygiene item:

If your firewall names itself in its block page, consider turning that off. It costs nothing and removes an attacker's starting point (see the supplementary finding below).

How this differs by organization size: A mid-size organization with one WAF vendor and one engineer should start with the first three recommendations for WAF teams, which need only a hostname list and a small set of scripted requests. A large enterprise with several WAF and CDN vendors will find that ownership and the inventory take longer than the configuration changes, and should run the CISO recommendations first. The checks in the first, second and fourth recommendations for WAF teams can be run by hand. For estates with hundreds or thousands of applications, IONIX WAF Posture Management runs them continuously.

Supplementary finding: 55% of applications behind a blocking WAF name the vendor in the block response

Published bypass research is organized by product. A firewall that identifies itself narrows an attacker's starting point. Of apps that demonstrably block attacks, 54.9% name their vendor in the block response itself. The other 45.1% return a bare rejection and give an attacker nothing to look up.

The spread between organizations is wide: a quarter of them run under 18% of their firewalls silently, and a quarter run over 76% silently. We read that range as a sign that branding follows the products an organization runs rather than a deliberate choice.

This is a hygiene finding. The recommendation is to reduce branding in block responses rather than to change products.

Outlook

We expect the coverage gap to persist while WAF capability continues to arrive bundled with CDN and cloud platforms and is enabled one property at a time. We assess that the figures would improve fastest if platforms enabled their core and SQL injection rule sets in blocking mode by default, and if audit and insurance questions moved from whether a WAF is present to evidence that it blocks. We will repeat this measurement in 2027 and report changes against these baselines.

About the research team

This report was produced by IONIX Research, the team who studies how organizations' internet-facing assets are exposed and how attackers reach them. The team's work centers on validation from the outside: testing what an attacker would encounter on live systems, with no access to internal inventories or configuration data. That approach shapes this report, which measures how web application firewalls respond to attack traffic on live applications across hundreds of organizations. IONIX Research also tracks newly disclosed vulnerabilities and how attackers can exploit them, and its findings inform the IONIX platform and the security teams that rely on it.

About IONIX

IONIX makes an organization's attack surface self-healing. Our External Exposure Management platform continuously discovers your external attack surface and digital supply chain, validates which exposures attackers can exploit, and fixes them through the edge controls security teams already run, before attackers get there first. Agentless and non-intrusive, IONIX delivers the attacker's view from day one. Many of the world's largest organizations, across pharma, healthcare, financial services, retail and technology, rely on IONIX to protect some of the most complex attack surfaces in the world.

IONIX WAF Posture Management

IONIX WAF Posture Management shows security teams which internet-facing assets have a WAF that blocks attacks and which do not, across every vendor, cloud and business unit. It tests each WAF from the outside, the way an attacker would, and gives WAF teams and CISOs a defensible record of coverage for audits and executive reporting.

Coverage. IONIX automatically classifies every web-facing asset as Protected, Underprotected or Unprotected, and identifies the WAF product in front of it across Akamai, Cloudflare, AWS, Azure, Imperva, Barracuda, Fortinet, Palo Alto Networks, Fastly and more than 50 other vendors. Coverage is reported as a single percentage, broken down by business unit, geography, asset criticality and vendor.

Effectiveness. IONIX continuously validates that deployed WAFs are in blocking mode, hold up against common evasion techniques, and block real-world XSS, SQL injection and RCE payloads. For every protected asset, it reports rule count, last update date and pass rate against attack scenarios, and flags configuration drift when rules are added, removed or downgraded between assessments.

Prioritization and response. IONIX scores unprotected assets on six factors, from traffic volume and business criticality to regulatory scope, and turns the result into a tiered deployment roadmap. Teams can confirm a critical rule is live everywhere from one place. When a new CVE is published, IONIX translates it into a deployable WAF rule and delivers it within 12 hours of publication.

Learn more about IONIX WAF Posture Management

WAF Program Maturity Quiz

Data Collection Methodology

Appendix A: Data Collection Methodology

Every figure is a population-level proportion measured from outside, against live production applications. Where a number is reported for "the median organization", each organization is scored separately and the middle one is reported, so a few very large estates cannot drive it. Where a caveat matters, it sits next to the number it qualifies. Several findings are deliberately conservative: where a block is observed but the product behind it cannot be named, the application is counted as protected.

External

Measurement
Active probing

5 classes

Attack probes
XSS · SQLi · LFI · RCE · XXE

Glossary

WAF (web application firewall)	Any control in the request path that inspects HTTP traffic and can refuse a request because of its content. Used interchangeably with “firewall” in this report
Blocking WAF (also “proven blocking”)	A WAF that blocked at least one of the five attack probes. An application behind one counts as protected
Valid web application	A live domain or subdomain, reachable over HTTPS on port 443, returning a non-error status, serving real content, not parked and showing no signs of abandonment.
Logging mode	A WAF setting in which rules log matching requests without blocking them. Also called monitor, log-only or detection mode
WAF-capable hosting	A hosting, CDN or cloud platform that offers a WAF module, whether or not it is enabled
Dedicated WAF product	A WAF sold as a security product in its own right
Platform-bundled WAF module	A WAF offered as an add-on to a CDN or cloud platform bought for delivery, hosting or bandwidth
Origin	The server that hosts the application behind the CDN or WAF
Origin bypass	A request sent directly to the origin that returns the application without passing through the WAF
Host header	The HTTP header that names the site a request is for. A request to the origin’s address with the correct Host header can return the site directly
Externally operated application	An application run by a third party, such as a SaaS vendor, agency or managed host, on the organization’s behalf
Organization-operated application	An application the organization runs itself
Median organization	The middle value when each organization is scored separately
Fingerprint	Evidence in responses, headers or DNS that a vendor’s product is in the request path. A fingerprint does not show that the product blocks
SSO-backed login page	A login page that hands authentication to a single sign-on identity provider, for example through SAML
XSS, SQLi, LFI, RCE, XXE	Cross-site scripting, SQL injection, local file inclusion, remote code execution and XML external entity injection: the five attack classes probed

How to cite this report

IONIX Research. State of the WAF 2026. IONIX, October 2026.
https://www.ionix.io/state-of-the-waf-report/

For press and analyst contacts

Press and questions about the data: Ilya Kleyman, IONIX CMO
ilya.kleyman@ionix.io

WAF POSTURE MANAGEMENT

Take Control of Every WAF Across Every Domain

Book a Demo