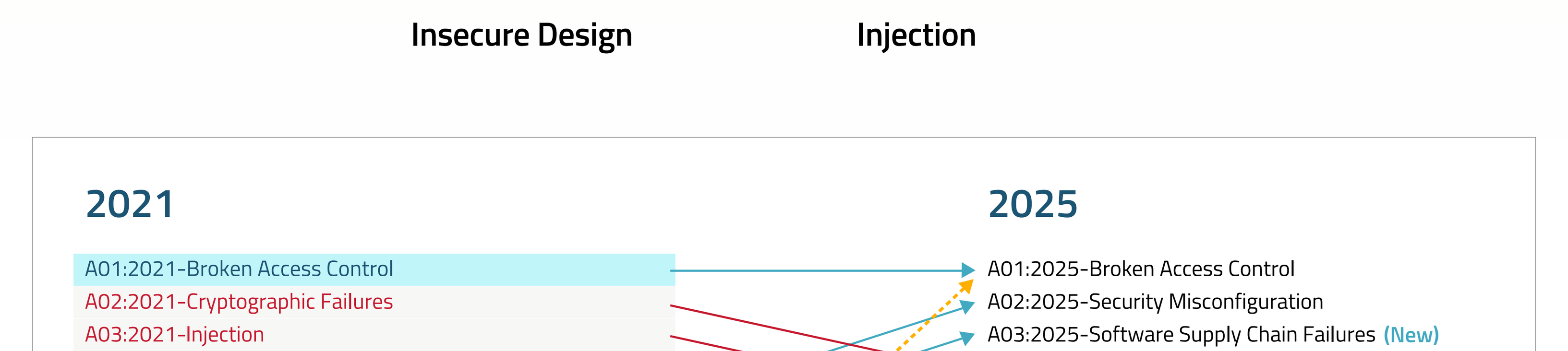
OWASP Top 10 Survival Guide: Risks vs WAFs

The risks of everyday life require your persistence and attention. You might always secure your seatbelt, but you still need to look up from your smartphone while crossing the street. There's plenty of room for both habits. In the same way, good application security requires a combination of web application firewalls (WAFs), AI, API security and bot protection to overcome old and new risks from today's threat landscape. From DevOps to new attack vectors, these changes can leave security professionals scrambling to safeguard their most prized digital assets.

The Open Web Application Security Project (OWASP) Top 10 list is an invaluable tool for accomplishing this. Since 2003, this list has sought to provide security professionals with a starting point for ensuring protection from the most common and virulent threats, application misconfigurations that can lead to vulnerabilities, as well as detection tactics and remediations. This guide has been updated to reflect the latest installment of the list, which introduces the new Software Supply Chain Failures and Mishandling of Exceptional Conditions risk categories and folds Server-Side Request Forgery into Broken Access Control. The sections below walk through what each risk means today and how WAF capabilities can help address it.

Top 10 OWASP Application Vulnerabilities



OWASP Top 10 Overview and WAF Capabilities to Mitigate Threats

A01: Broken Access Control

Improperly configured or missing restrictions on authenticated users allow them to access unauthorized functionality or data. Also, restrictions on what authenticated users are allowed to do are often not properly enforced.

It's the most pervasive risk on the list: OWASP found some form of broken access control in every application it tested. Common patterns include insecure direct object references that expose one user's data to another, privilege escalation that lets a regular user reach admin-only functions and forced browsing into hidden directories or files. Server-Side Request Forgery (SSRF), where an attacker tricks an application into sending crafted requests to an unintended destination, is also now grouped under this risk, since at its root it's still an authorization failure.

Mitigation: Fastest Time to Protection

Penetration testing is essential for detecting nonfunctional access controls; other testing methods detect only where access controls are missing. It can take several weeks to test, produce and assess these reports and then implement necessary security changes. Any WAF should serve as a catalyst for stemming unauthorized access via authentication gateway functionality, single sign-on, user tracking and access controls to the web application based on user role, profile information and security token validations. With SSRF now folded into this risk, a WAF's ability to validate outbound and server-side requests is just as important as its inbound access controls. And as more hackers turn to frontier AI tools to rapidly uncover and exploit vulnerabilities, it's important to turn pen testing data into protection with customer-specific signatures.

A02: Security Misconfiguration

Security misconfiguration remains one of the most commonly seen web application security issues to this day. This risk refers to improper implementation of controls intended to keep application data safe, such as insecure default configurations, incomplete or ad hoc configurations, open cloud storage, misconfigured HTTP headers, and perhaps most important, not patching or upgrading systems, frameworks, libraries, applications and components. This category also includes XML external entity flaws, where poorly configured XML processors evaluate external entity references and can be used for remote code execution, internal file disclosure, port scanning and denial-of-service attacks.

OWASP found some form of misconfiguration in every application it tested, with more than 719,000 mapped instances of a related weakness, the highest total of any category. Because so much of a modern application's behavior is now controlled by cloud permissions, container settings and infrastructure templates rather than code, a single unnecessary open port, exposed admin console or unremoved test account can undo every other security control in place.

Mitigation: Ability to Learn

As notable ransomware and malware outbreaks in recent years have proven, system upgrades are critical. An adaptive WAF will leverage automatic policy generation and machine learning capabilities to automatically create and apply security filters and enforcement rules where security is misconfigured. It will evaluate the structure of a web application, set relevant security filters and analyze traffic properties from a production environment to build a dynamic network profile, thereby maximizing security while minimizing false positives.

A WAF should be able to parse and inspect protocols and structured documents, including HTTP and HTTPS traffic, POST requests and XML JSON schemas. In addition, the machine learning algorithms can learn XML and JSON structures and schemas for enforcement as part of the validation phase and introduce customer-specific signatures as part of current security policies.

A03: Software Supply Chain Failures

Various components, such as libraries, frameworks, and other software modules, run with the same privileges as the application. If a vulnerable component is exploited, such an attack can facilitate serious data loss or server takeover. Developers frequently don't know which open source or third-party components are in their applications, making it difficult to update components when new vulnerabilities are discovered. These components can undermine application defenses and enable various attacks and impacts.

OWASP has broadened this category well beyond known component vulnerabilities to cover the entire software supply chain, including build systems, tampered CI/CD pipelines, and malicious or hijacked packages that make it into production. It was the community's top-voted concern in the 2025 survey, with half of respondents ranking it their number-one risk.

Mitigation: Knowledge of Where the Holes Exist

Any WAF that provides integration with programs such as Microsoft's Windows Server Update Services can protect against exploitations of vulnerable and outdated components by screening client requests and server responses. In addition, security updates and threat intelligence feeds are essential to keep security teams in the know and facilitate quicker responses to maximize protection and reduce exposure.

As this risk expands to cover the full supply chain, pairing a WAF with software composition analysis and a maintained Software Bill of Materials (SBOM) helps security teams spot vulnerable or unmaintained dependencies before attackers do. And rather than relying on a one-size-fits-all protection approach, use of tailored signatures provides protection while patching is implemented and tested.

A04: Cryptographic Failures

Cryptographic failures (formerly known as Sensitive Data Exposure) focus on cryptography-related failures, which often lead to sensitive data exposure or system compromise. Many web applications and APIs contain vulnerabilities due to coding, thereby exposing sensitive data such as financial, healthcare and personally identifiable information. Attackers may steal or modify such weakly protected data to conduct credit card fraud, identity theft or other crimes. Sensitive data may be compromised without extra protection, such as encryption at rest or in transit, and requires special precautions when exchanged with the browser.

OWASP's 2025 data ties this risk to weak or predictable random number generation and outdated cryptographic algorithms. OWASP is also flagging post-quantum cryptography as a forward-looking concern, recommending that high-risk systems prepare for quantum-resistant algorithms no later than the end of 2030.

Mitigation: Encryption

Encryption is key, both for data at rest or in transit. Leading WAFs provide inspection and encryption of data, including SSL inspection and protection capabilities to eliminate security blind spots. This includes, but is not limited to, SSL traffic decryption and encryption, masking server identities and veiling sensitive information. An adaptive WAF that leverages automatic policy generation and machine learning capabilities to automatically create and apply security configurations and policies is also critical. Finally, any enterprise-grade firewall should support the encryption of ingress and egress traffic across both on-premises and cloud-based infrastructures.

A05: Injection

Injection flaws, such as SQL, NoSQL, OS and Lightweight Directory Access Protocol (LDAP) injection, have been a perennial favorite among hackers for some time. An injection flaw occurs when suspicious data is inserted into an application as a command or query. This hostile data can trick the interpreter into executing unintended commands or accessing data without proper authorization. The most common code injection is a SQL injection, which is an attack that is accomplished by sending malformed code to the database server. Cross-site scripting (XSS) is included as part of this category as well. XSS occurs whenever an application includes untrusted data in a new webpage without proper validation, or updates an existing webpage with user-supplied data using a browser API that can create HTML or JavaScript. These flaws give attackers the capability to inject client-side scripts into the application to hijack user sessions, deface websites or redirect the user to malicious sites.

OWASP tested for some form of injection in 100% of applications reviewed, and this category has generated more reported vulnerabilities than any other—over 30,000 for Cross-Site Scripting alone and more than 14,000 for SQL injection. XSS shows up far more often but tends to be lower-impact, while SQL injection is rarer but more severe, which is why this category's overall risk score looks different than its sheer volume of findings might suggest. OWASP also notes that prompt injection against large language models is an emerging, related risk, though it's tracked separately in OWASP's LLM Top 10.

Mitigation: Positive Protection

Many web application security solutions leverage a negative security model, which defines what is disallowed while implicitly allowing everything else. Since attack signatures may generate false positives by detecting legitimate traffic as attack traffic, such rules tend to be simplistic, trying to detect the obvious attacks. The result is protection against the lowest common denominator.

A positive security model, which defines the set of allowed types and values, is required to provide proper protection where signature-based protection cannot fill the gap. In the case of a SQL injection, a positive security model screens user input for known patterns of attacks and leverages logic to tell the difference between legitimate user input and injection flaws.

Against XSS attempts, it's important to make sure any WAF can provide signature- and rule-based protection with updated signatures (similar to a blacklist), identify scripting patterns and block malicious requests. And as new AI tools begin to rapidly take advantage of injection flaws, virtual automated patching is needed more than ever to block exploit attempts at runtime.

A06: Insecure Design

This category focuses on risks related to design flaws. This means using more threat modeling for secure design patterns and principles in the earlier stages of the application development cycle. According to OWASP, secure design is a culture and methodology that constantly evaluates threats and ensures that code is robustly designed and tested to prevent known attack methods. Secure design requires a secure development lifecycle, some form of secure design pattern or paved road component library or tooling, and threat modeling.

OWASP draws a clear line between insecure design and insecure implementation: a secure design can still be undermined by a sloppy implementation, but no amount of careful coding can fix a design that never accounted for a given attack in the first place. This category also covers business-logic flaws, such as an application that never defines what an unexpected or unwanted state change should look like. The good news is that OWASP notes real, measurable improvement across the industry in threat modeling and secure-by-design practices since this category was introduced in 2021.

Mitigation: Secure Software Development

First, secure software development lifecycle (SDLC) methodologies must be adapted. Much of this revolves around the continuous integration and continuous development (CI/CD) pipeline. To that end, it's critical to ensure that any web application security solution provides a series of core capabilities that address security assessment at the early stages of application deployment, including API management to configure the web application firewall in such environments, automatic policy generation to automatically create new security policies based on any new application, and integration with dynamic analysis security testing (DAST) tools to create security policies based on a DAST security assessment.

In an unsecure SDLC environment, there is no one-size-fits-all capability to ensure application security. However, the following features can ensure a security policy is properly tailored to safeguard the application: security filters leveraging a positive/negative security model; IP-, geo-, and role-based policies; rate-limiting access to the server resources; use of a multilayered attack correlation detection engine. And with new AI tools proving to be especially adept at discovering and exploiting flaws, tailored and automated virtual patching is needed now more than ever to provide quick protection while response teams work.

A07: Authentication Failures

When an application is not implemented correctly, the door is left open for criminals to break in. Attackers can compromise passwords, keys or session tokens or exploit other implementation flaws to assume other users' identities temporarily or permanently. Sessions should be unique to individual users. Without some session management, an attacker can sneak in disguised as a user to access valuable data.

OWASP's 2025 guidance calls out specific patterns to watch for: storing passwords in plaintext or with weak hashing, missing or ineffective multi-factor authentication, session identifiers exposed in a URL or hidden field, and session tokens that aren't properly invalidated at logout or after a period of inactivity. OWASP notes that wider adoption of standardized authentication frameworks is helping to temper this risk's growth industry-wide.

Mitigation: Challenge and Validate

Securing an application in terms of access control is no easy task. Authenticating users by having them provide their identity and challenging them to verify their identity is a key first step. Single sign-on and multifactor authentication are also key steps that reduce the risk of compromised accounts.

A key second step is to have a WAF that proactively encrypts session parameters between network and client, proactively inspects login attempts and thwarts HTTP sessions via code-encrypting, cryptographic capabilities.

Exploitation of hidden flaws are more common with hacker use of AI tools. Customer-tailored virtual patching can help close the gap between discovery and full patching.

A08: Software or Data Integrity Failures

Software and Data Integrity Failures refers to code and infrastructure that fails to protect against integrity violations. This includes software updates, critical data and CI/CD pipelines that are implemented without verification. An example of this includes objects or data encoded or serialized into a structure that an attacker can modify. Another is an application that relies on plugins, libraries or modules from untrusted sources. Insecure deserialization is part of this category and often leads to remote code execution, or can be used to perform replay or injection attacks and privilege escalation.

OWASP now draws a clearer line between this risk and the newer Software Supply Chain Failures category: where Supply Chain Failures looks at your components and dependencies, this category asks whether the actual code, updates and data your application uses have been verified as legitimate and untampered, at a level below the supply chain itself. Notable examples OWASP calls out include untrusted plugins, insecure CI pipelines, and unsigned updates.

Mitigation: Best of Both Worlds

To mitigate insecure deserialization vulnerabilities, it's useful to identify WAFs that provide the best of both worlds, combining negative (defining what is forbidden and accepting the rest) and positive (defining what is allowed and rejecting the rest) security models. This winning combination should leverage various WAF access control filters such as cookie encryption, XML and JSON parsing, parameters enforcement and more.

A09: Security Logging & Alerting Failures

Security logging failures, coupled with missing or ineffective integration with incident response systems, are the bedrock for the majority of incidents, allowing attackers to run amok, attacking further systems and tampering with, extracting or destroying data. Many studies show that the time to detect is measured in weeks or months, typically detected by external parties rather than internal processes or monitoring. Attackers rely on the lack of monitoring and timely response to achieve their goals without being detected; allowing probes to continue unchecked can raise the likelihood of a successful exploit to nearly 100%.

OWASP renamed this category from "Security Logging and Monitoring Failures" to make a specific point: logging events is only half the job. As OWASP puts it, great logging with no alerting is of minimal value in identifying security incidents in time to act on them. This category is also, by nature, hard to measure through automated testing, which is why—just as in 2021—it earned its spot on the list through OWASP's community survey rather than raw incidence data alone.

Mitigation: Suite Solutions Versus Best of Breed

To address the issue of internal processes, it's vital to think like an attacker and internally test and audit to discover if an organization has sufficient monitoring. If it lacks this "white hat hacker" expertise, the cybersecurity vendor chosen as a partner must provide distributed denial-of-service (DDoS) mitigation expertise via a team of security experts.

These same experts should also play a role in the second-biggest concern, which is real-time monitoring and detection. Timely detection of malicious malware or snooping hackers comes down to best-of-breed versus suite offerings. Stopping cyberattacks in near-real time is best accomplished through a single vendor attack mitigation system. Many organizations leverage best-of-breed mitigation tools from different vendors. This hodgepodge collection results in poor communication and detection. Suite WAF and DDoS solutions can more effectively communicate, setting network traffic baselines and comparing data points to quickly detect when something is awry, in addition to providing enterprise-grade monitoring and management dashboards and analytics.

Because this category now explicitly calls out alerting, make sure any monitoring dashboard converts detections into timely, actionable alerts, not just historical logs.

A10: Mishandling of Exceptional Conditions

Mishandling of Exceptional Conditions covers what happens when an application encounters an unexpected state, such as a missing parameter, a timed-out dependency, a database error or a resource limit, and responds unsafely instead of failing closed. OWASP points to patterns like an authorization check that correctly denies access under normal conditions but grants access by default if the underlying service errors out or times out, or a verbose error message that leaks internal details, credentials or system paths back to an attacker who is deliberately probing for weaknesses. Because these issues were often waved off as generic code-quality bugs rather than security bugs, OWASP introduced this as a dedicated category to make sure they get the attention they deserve.

Mitigation: Fail Closed, Not Open

Because this is a new category, WAF capabilities should evolve alongside it. Look for a WAF that masks verbose application error messages before they reach the client, enforces rate limiting and resource quotas to prevent attackers from exhausting system resources through repeated failed requests, and can detect and block the probing patterns attackers use to trigger and study error conditions. Pairing this with centralized logging and alerting (see Risk 9) helps ensure exceptional conditions are caught quickly rather than exploited. Quickly responding with virtual patching will help minimize damage from new AI tools that can rapidly exploit flaws.

Closing Thoughts

Just like the adaptive threat landscape it seeks to define, this list continues to be updated to serve as an industry benchmark for the application security community. This piece provides an overview of the 2025 OWASP Top 10 list and technical capabilities security professionals should consider when evaluating WAFs.

The OWASP Top 10 is not intended as "one list to rule them all," but rather serves as a great starting point for application security programs and WAF evaluation. It acts as a benchmark for empowering improved people, processes and technology.

It shows us that successful organizations must establish and use repeatable processes and security controls. Testers should establish continuous application security testing, and application managers need to take charge of the application lifecycle. It's crucial to have an application security program in place that effectively coordinates across all facets of its infrastructure.

To that end, selecting the right WAF vendor to partner with is a critical step in executing these concepts. Be sure any WAF solution your organization is evaluating not only meets your organization's existing security needs but is flexible enough to adapt to future infrastructure environments, business needs, attack vectors and rapidly evolving AI tools.

RWI6 421 | 07.23.2026