# WAF evaluation checklist

## Considerations for purchasing an application and API protection solution to secure your digital assets

### Detection capabilities

- ☐ **Common threats**

An API should have built-in protection for common threats, such as OWASP Top 10 and OWASP API Security Top 10.

- ☐ **Virtual patching**

Many organizations struggle to keep up with updates and patching new vulnerabilities. A WAF should offer virtual patching which blocks attempts to exploit known vulnerabilities.

- ☐ **Regular updates**

New vulnerabilities are discovered every day. A WAF's database of known attacks should be regularly updated to provide up-to-date protection.

- ☐ **Account takeover (ATO)**

Account takeover attacks (such as brute force password guessing or credential stuffing) are increasingly common. A WAF should automatically detect and block these attempted attacks.

- ☐ **Business logic attacks**

Some attacks (such as cookie tampering and directory traversal) target the business logic of an application. A WAF should detect and block attempts to exploit these attack vectors.

- ☐ **Identify traffic from datacenters, Tor, proxies**

A WAF should be able to correctly identify and report requests coming from known data centers (in contrast to requests coming from residential IP addresses), Tor exit points and proxy servers.

### Supported services

- ☐ **Websites and web applications**

A WAF should provide comprehensive protection for websites of any type. This includes single page applications (SPA), websites, and web applications.

- ☐ **APIs**

APIs are a growing component of an organization's digital infrastructure. A WAF should support common API protocols, including XML-based (like SOAP), JSON-based (like REST), GraphQL, and gRPC.

- ☐ **Serverless**

Serverless applications are growing in popularity. A WAF should be able to protect AWS Lambda, Azure Functions, and GCP Cloud Functions.

### Deployment and scalability

- ☐ **Cloud-native deployment**

Organizations are increasingly moving their applications to the edge. A WAF should be designed to operate in cloud environments and take advantage of the virtualized deployment (e.g. to be deployed in Kubernetes as a sidecar proxy or ingress controller).

- ☐ **Multi-tenancy support**

An organization may have multiple different sites (or multiple departments/subsidiaries) that it wants to protect against attacks. A WAF should offer multitenancy to enable multiple sites to be protected by a single solution with proper user permissions management capability.

☐ **Native integration with popular web server software**

A WAF should support a module-based integration in your existing NGINX load balancer.

☐ **Scales with clusters (horizontal scaling)**

It should be easy to deploy an auto scaling cluster of WAF nodes using provided Terraform automation code.

☐ **Full environment support**

A WAF should be able to provide protection in any deployment environment and should be optimally deployed close to send users.

☐ **Infrastructure integration**

A WAF should be built to work seamlessly with the underlying infrastructure. Your cloud or CDN provider should fully support optimized WAF deployment.

## Usability

☐ **Easy configuration and updates**

An organization needs to be able to easily configure its WAF to meet its unique business needs and install updates to take advantage of new features and functionality.

☐ **Ability to access WAF documentation**

A WAF solution should provide a detailed documentation site about how to deploy and use the system.

## Cost

☐ **Should be within the budget**

Cyber protection becomes a key part of IT infrastructure today & customers should be able to afford WAF protection relevant to their size & infra maturity level.

☐ **Clear pricing model**

Some pricing structures are unclear & complicated, so customers are exposed to un-expected price increase with more traffic & usage. Ideally, WAF vendors should have a single trigger for price increases with predictable & transparent pricing model, so customers can plan ahead with their operations growth.

## Low management overhead

☐ **Low false positives**

High false positive rates commonly drive WAF users to deploy solutions in passive/monitoring mode. A WAF should offer a low false positive rate to make production deployment usable.

☐ **Signature-less attack detection capabilities**

A signature based WAF is typically more difficult to manage (add rules to avoid false positives) while keeping a high level of application protection. Your WAF should be able to block malicious requests without a need to manage signatures.

☐ **Auto-adjustment of security rules**

Per customer and per application. A WAF should automatically learn the application structure and create necessary security rules.

☐ **Managed SOC team**

WAF solutions with vendor cloud-based monitoring & protection module should provide SOC capability to the customers as a part of subscription service.

## Observability

☐ **Understandable, informative, customizable dashboards**

WAF vendors should meet SOC2 compliance requirements & have SOC2 certificate to meet customer standards and practices.

☐ **Deep dive on “why” of blocking**

Give me a pcap.

## Compliance and reporting

☐ **Regulatory compliance support**

Organizations are subject to different regulations with associated security and reporting requirements. A WAF should offer support for common regulations (like PCI DSS or GDPR) and enable users to easily collect data and generate reports for auditors or regulatory authorities.

☐ **Built-In report formats**

An organization's security team may need to generate reports for executives, auditors, etc. A WAF should have integrated support for generating common reports.

☐ **SOC2 compliance**

WAF vendors should meet SOC2 compliance requirements & have SOC2 certificate to meet customer standards and practices.

## Active checks / vulnerability scanner

☐ **Integrated vulnerability detection**

A WAF should automatically identify potential vulnerabilities within an organization's applications. Detections should be based upon active/passive scanning, threat intelligence, and knowledge of public vulnerabilities.

☐ **Misconfiguration**

A WAF should be capable of detecting and alerting on misconfigurations that impact the security or usability of an application or API.

## Integrations

☐ **Public API**

A WAF should include a publicly accessible API so that it be integrated with a variety of external solutions, such as log management with an ELK stack.

☐ **Webhooks**

A WAF should incorporate support for webhooks to enable development of custom issue tracking and analytics platforms.

☐ **SIEM integrations**

A Security Information and Event Management (SIEM) solution is designed to provide security data aggregation and analytics. A WAF should have integrations for major SIEM platforms: Splunk, Sumo Logic, IBM QRadar.

☐ **DevOps tool integrations**

Adoption of DevOps principles means that development teams need to be able to automate testing and deployment activities. A WAF should integrate into DevOps pipelines to enable rapid configuration updates. A WAF should have built-in integrations for major DevOps tools.

☐ **Messenger integrations**

Security teams need to rapidly respond to potential incidents. A WAF should include integration with common messaging platforms for instantaneous notifications.

☐ **Smart notifications**

WAF users should be able to customize & set event notifications (also known as "Triggers") including integrations with SIEM, DevOps & messenger tools. Apart from notifications on events (like attacks), WAF users should be able to use smart blocking techniques & set quick action rules.

☐ **Metrics exposed**

WAF nodes should provide helpful monitoring metrics in the popular Prometheus format.

## API protection

☐ **Protection for modern APIs**

Built on a modern tech stack and using REST, SOAP, gRPC GraphQL, and WebSockets.

☐ **API abuse protection**

Not all attacks against an API are designed to exploit known vulnerabilities. A WAF should also be able to identify and block traffic that abuses an organization's API.

☐ **Protection without a provided API schema**

A WAF should be able to automatically protect API endpoints without a need for the user to provide API schema definitions.

To learn more about Lumen application protection and Wallarm on Lumen's cloud-native approach to WAF, contact us at application.delivery@lumen.com.



Wallarm is an AI startup focused on automated protection of websites, microservices, and APIs running on public & private clouds. The Wallarm security platform automates application security with application-specific WAF rules and vulnerability security tests. The technology provides dynamic, active, and focused security for hundreds of enterprises and SaaS companies.

wallarm

## Why Lumen?

With global network scale, a secure and programmable delivery platform, and dedicated focus on improving business outcomes, Lumen is trusted by the world's leading enterprises to help them create new and differentiating user experiences all over the globe. Whether you are delivering video streams, popular games, online storefronts, or next generation applications, the broad capabilities of Lumen edge delivery services enable businesses to stand out from the crowd.





LUMEN® TECHNOLOGIES