CHECK POINT™

WAF COMPARISON PROJECT 2026

How does your WAF stack up?

YOU DESERVE
THE BEST SECURITY

CONTENTS

INTRODUCTION

This article describes the results of our annual WAF Efficacy comparison. For the third year in a row, we tested leading WAF solutions in rigorous, real-world conditions (see the 2024-2025 blog here), triggering both malicious and legitimate web requests to measure exactly how well - or how poorly - these vendors protect modern applications.

While previous years focused on the decline of ModSecurity and updates to the OWASP Core Rule Set, 2025 marked a shift toward exposing the **architectural limitations** of traditional WAFs. As vulnerabilities become more complex, the limitations of legacy, signature-based engines are becoming impossible to ignore.

This year's test introduces a new focus on **Padding Evasion** - inspired by the critical React2Shell vulnerability - and highlights how fixed-buffer inspection limits are leaving modern applications exposed.

THE CORE CHALLENGE: SECURITY VS. DETECTION

The two most important parameters when selecting a Web Application Firewall remain:

- **Security Quality (True Positive Rate) -** the WAF's ability to correctly identify and block malicious requests is crucial in today's threat landscape. It must preemptively block Zero day attacks as well as effectively tackle known attack techniques utilized by hackers.
- **Detection Quality (False Positive Rate) –** aka the WAF's ability to correctly allow legitimate requests is also critical because any interference with these valid requests could lead to significant business disruption and an increased workload for administrators as much tuning is required.

A very comprehensive data set was used to test the products:

- **1,040,242** legitimate HTTP requests from **692** real websites in **14** categories
- **74,284** malicious payloads from a broad spectrum of commonly experienced attack vectors

Loyal to the spirit of open-source, we provide in this GitHub repository all the details of the testing methodology, testing datasets, and open-source tools that are required to validate and reproduce this test and welcome the community's feedback.

PRODUCTS TESTED AND RESULTS

This year's test was conducted in December 2025, and compared the following popular WAF solutions:

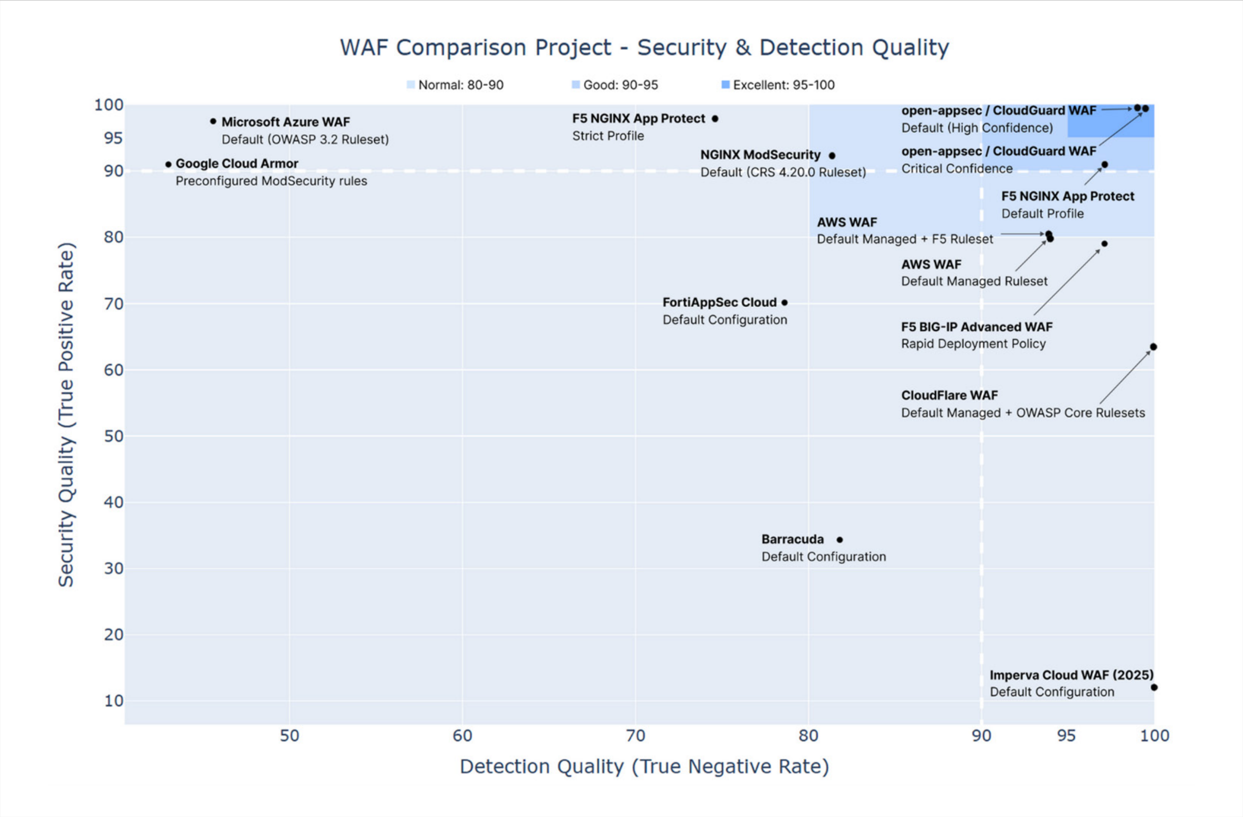
- **Microsoft Azure WAF –** OWASP CRS 3.2 Ruleset
- **AWS WAF –** AWS Managed Ruleset
- **AWS WAF –** AWS Managed Ruleset and F5 Ruleset
- **CloudFlare WAF –** Managed and OWASP Core Rulesets
- **F5 NGINX App Protect WAF –** Default Profile
- **F5 NGINX App Protect WAF –** Strict Profile
- **NGINX ModSecurity –** OWASP CRS 4.20.0 (updated from previously tested version 4.3.0)
- **CloudGuard WAF –** Default Configuration (High Confidence)
- **CloudGuard WAF –** Critical Confidence Configuration
- **F5 BIG-IP Advanced WAF –** Rapid Deployment Policy
- **Fortinet FortiWeb –** Default Configuration
- **Google Cloud Armor –** Preconfigured ModSecurity Rules (Sensitivity level 2)
- **Imperva Cloud WAF (2024-2025) –** Default Configuration
  Note: The results displayed for Imperva Cloud WAF are based on the 2024-2025 test data. We were unable to update the score for this year due to the specific blocking behavior described in the introduction

This year we also added a new vendor:

- **Barracuda WAF –** Default Configuration

The two charts below summarize the main findings.

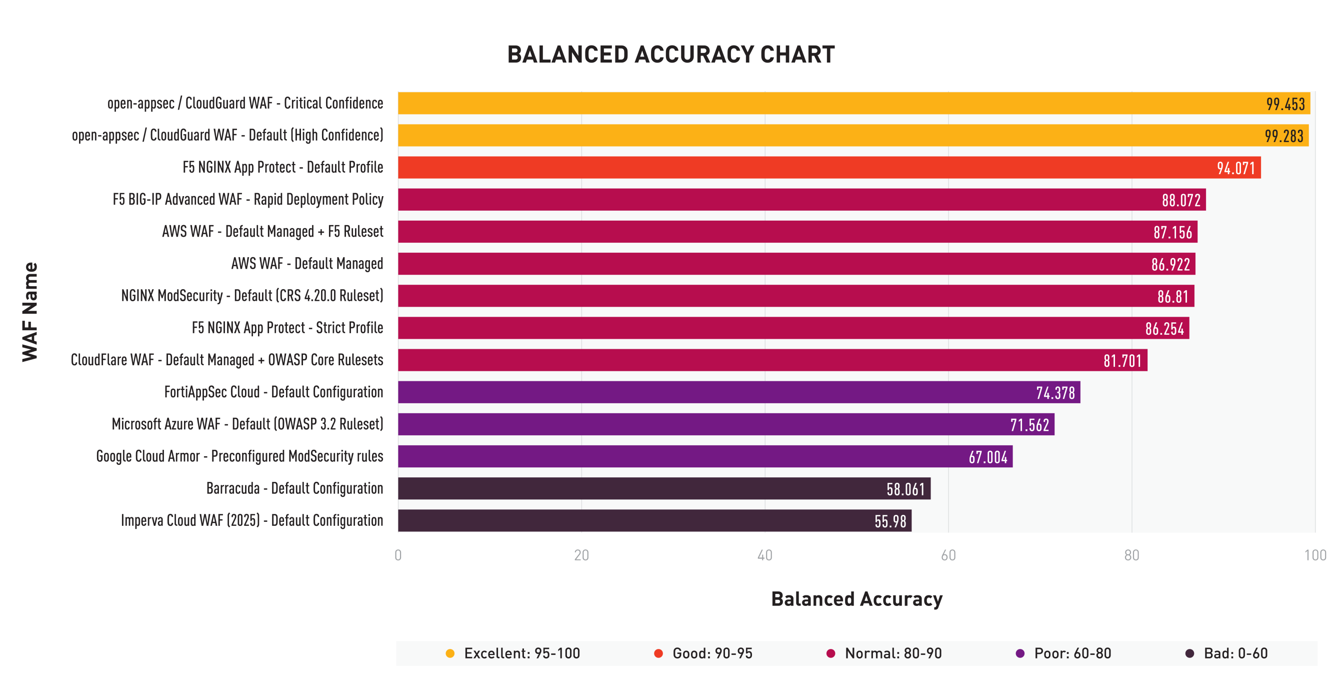
Security Quality and **Detection Quality** are often a tradeoff within security products. The first chart shows visually how different products perform in each category.



The test reveals significant differences in product performance. For example:

- Imperva's WAF has a near-perfect Detection Quality (False Positive Rate) of **0.009%** correctly allowing almost all legitimate traffic. However, surprisingly it achieves only an **11.97%** Security Quality (True Positive Rate), missing 88.03% of actual threats and reducing its overall security effectiveness.
- Azure WAF offers very high Security Quality (**97.537%**) but has an extremely high false positive rate of **54.412%**, potentially blocking legitimate requests and disrupting normal operations. These results suggest that some products may pose security risks due to missed detections or require substantial tuning to balance security effectiveness with usability.

To provide security and allow minimal administration overhead, the optimal WAF solution should strike a balance, exhibiting high performance on both **Security Quality and Detection Quality**. This is aptly represented by a measurement called Balanced Accuracy - an arithmetic mean of the True Positives and True Negatives rates.



For the third year in a row, **CloudGuard WAF leads in balanced accuracy**, achieving the highest scores with **99.453%** in the Critical Profile and **99.283%** in the Default Profile. This performance outpaces all other WAF solutions, with NGINX AppProtect in the Default Profile following at **94.071%**.

KEY UPDATES & FINDINGS FOR 2026

Before diving into the full data, several major developments defined this year's testing cycle:

1. **New Attack Vector: Padding Evasion & React2Shell** Modern exploits are getting larger. The recent **React2Shell** vulnerability (CVE 2025 55182, CVSS 10.0) demonstrated that critical payloads often exceed the standard 8KB or 128KB inspection buffers used by many legacy WAFs. To test this, we introduced a Padding Evasion dataset containing hundreds of malicious payloads hidden behind junk data of varying lengths.

   a. **The Industry Challenge:** We observed that some vendors, notably Cloudflare, default to ignoring payloads exceeding specific sizes (e.g., 128KB) to maintain performance, effectively failing open on large attacks.

   b. **The Solution:** The test confirms that specific architectural approaches - specifically streaming analysis combined with Machine Learning - are required to inspect the entire request body without compromising performance.

2. **Tooling Evolution: PDF Reports & Docker** Based on extensive feedback from the DevSecOps community, we have transformed the testing tool from a script into a platform.

   c. **Automated PDF Reporting:** The tool now auto-generates a boardroom-ready PDF report, visualizing Balanced Accuracy, specific attack vectors, and providing log samples for True/False Positives/Negatives.

   d. **Docker & Quick Run:** To democratize testing, we added a Docker container for single-command execution and a "Quick Run" mode (testing 5% of the dataset) for rapid baseline checks.

3. **Imperva: Anomalous Blocking Behavior** We intended to include Imperva Cloud WAF in this year's comparison. However, shortly after testing began, the Imperva environment ceased inspecting traffic and started responding with a blanket HTTP 302 redirect to all requests - both malicious and legitimate. The redirect location was hardcoded to a non-existent domain (checkpoint.waftest.ca), confirming this was a manual configuration targeting our specific open-source testing tool.

   This behavior forces us to disqualify Imperva from the results. **Rather than inspecting the traffic content for malicious payloads, the WAF appeared to fingerprint and block the testing tool source.** This is a counter-productive strategy: by blindly blocking the tool to avoid "failing" the test, t**hey also blocked 100% of the legitimate traffic** sent by the tool. In a real-world scenario, this logic effectively creates a self-inflicted Denial of Service (DoS) rather than providing genuine security.

DEEP DIVE: THE PADDING EVASION DILEMMA

The introduction of the **Padding Evasion** dataset (inspired by CVE 2025 55182) revealed a fundamental architectural divide in the WAF market. Unlike standard injection attacks, padded payloads test the WAF's engine capacity rather than just its signature database.

When a malicious payload is padded with junk data to exceed the WAF's inspection buffer (typically 8KB–128KB), the WAF is forced into a difficult decision: stop inspecting and let the traffic pass or block the traffic blindly because it is "too big."

Our testing categorized the vendors into three distinct behaviors:

1. **Fail Open (Security Risk):** The WAF inspects the first X kilobytes, sees nothing malicious, and allows the request. This leaves the application completely vulnerable to padded Zero-Day attacks.

2. **Fail Close (Business Risk):** The WAF detects the request is too large and blocks it by default. While this stops the attack, it causes high False Positives by blocking legitimate large requests (e.g., extensive JSON bodies or file uploads).

3. **Full Coverage (Ideal):** The WAF can stream or buffer the entire payload, inspecting the malicious logic hidden deep within the body without blocking legitimate large traffic.

Padding Evasion Results:

WAF Solution	Behavior On Large Payloads	Impact	Status
CloudGuard WAF	Full Coverage	Protected	✓
Google Cloud Armor	Full Coverage	Protected	✓
Microsoft Azure WAF	Fail Close	Blocks legitimate large traffic	✕
AWS WAF (Managed & F5 Rules)	Fail Close	Blocks legitimate large traffic	✕
Barracuda WAF	Fail Close	Blocks legitimate large traffic	✕
Cloudflare WAF	Fail Open	Vulnerable	✕
F5 NGINX AppProtect (Strict/Default)	Fail Open	Vulnerable	✕
F5 BIG-IP Advanced WAF	Fail Open	Vulnerable	✕
NGINX ModSecurity	Fail Open	Vulnerable	✕
Fortinet FortiAppSec	Fail Open	Vulnerable	✕

Conclusion: Only **CloudGuard WAF** and **Google Cloud Armor** successfully inspected the full payload depth. The majority of the market (including F5, Cloudflare, and Fortinet) defaulted to a "Fail Open" state, rendering them ineffective against padded RCE vulnerabilities like React2Shell. Conversely, AWS and Azure prioritize security over usability, likely requiring significant exception handling for modern, data-heavy applications.

METHODOLOGY

DATASETS

Each WAF solution was tested against two large data sets: **Legitimate** and **Malicious**.

LEGITIMATE REQUESTS DATASET

The Legitimate Requests Dataset is carefully designed to test WAF behaviors in real-world scenarios. This year, we have updated the dataset to include **1,040,242 HTTP requests from 692 real-world websites**.

The data set was recorded by browsing real-world websites and conducting various operations on the site (for example, sign-up, selecting products and placing in a cart, searching, file uploads, etc.) ensuring the presence of 100% legitimate requests.

The selection of real-world websites of different types is essential because it is important for WAFs to examine all components of an HTTP request, including headers, URL, and Body as well as complex request structures like large JSON or other complex body types. This allows for accurate testing, as these elements can sometimes be the source of False Positives in real-world applications. Often synthetic datasets will overlook some of these.

The dataset in this test allows us to challenge the WAF systems by examining their responses to a range of website functionalities. For example, many HTTP requests are traffic to e-commerce websites. These websites often employ more intricate logic, making them an ideal ground for rigorous testing. Features such as user login processes, complex inventory systems equipped with search and filter functionalities, dynamic cart management systems, and comprehensive checkout processes are common in e-commerce sites. The dataset also includes file uploads and many other types of web operations. Incorporating these features allows us to simulate a wide range of scenarios, enabling an exhaustive evaluation of the efficiency and reliability of WAF systems under diverse conditions.

The distribution of site categories in the dataset is as follows:

Category	Websites	Examples
E-Commerce	404	eBay, Ikea
Travel	75	Booking, Airbnb
Information	59	Wikipedia, Daily Mail
Food	40	Wolt, Burger King
Search Engines	24	Duckduckgo, Bing
Social media	17	Facebook, Instagram
Files uploads	16	Adobe, Shutterfly
Content creation	13	Office, Atlassian
Games	13	Roblox, Steam
Videos	8	YouTube, twitch
Files download	7	Google, Dropbox
Applications	7	IBM Quantum Simulator, Planner 5d
Streaming	6	Spotify, Youtube Music
Technology	3	Microsoft, Lenovo
Total	**692**	

The Legitimate Requests Dataset including all HTTP requests is available here. We think that it is a valuable resource for both, users as well as the industry, and we are planning to continue updating it every year.

MALICIOUS REQUESTS DATASET

The Malicious Requests Dataset includes 73,924 malicious payloads from a broad spectrum of commonly experienced attack vectors:

- SQL Injection
- Cross-Site Scripting (XSS)
- XML External Entity (XXE)
- Path Traversal
- Command Execution
- Log4Shell
- Shellshock
- Padding Evasion

The malicious payloads were sourced from the WAF Payload Collection GitHub page that was assembled by mgm security partners GmbH from Germany. This repository serves as a valuable resource, providing payloads specifically created for testing Web Application Firewall rules.

As explained on the GitHub page, mgm collected the payloads from many sources, such as SecLists, Foospidy's Payloads, PayloadsAllTheThings, Awesome-WAF, WAF Efficacy Framework, WAF community bypasses, GoTestWAF, and Payloadbox, among others. It even includes the Log4Shell Payloads from Ox4Shell and Tishna. Each of these sources offers a wealth of real-world, effective payloads and provides a comprehensive approach to testing WAF solutions.

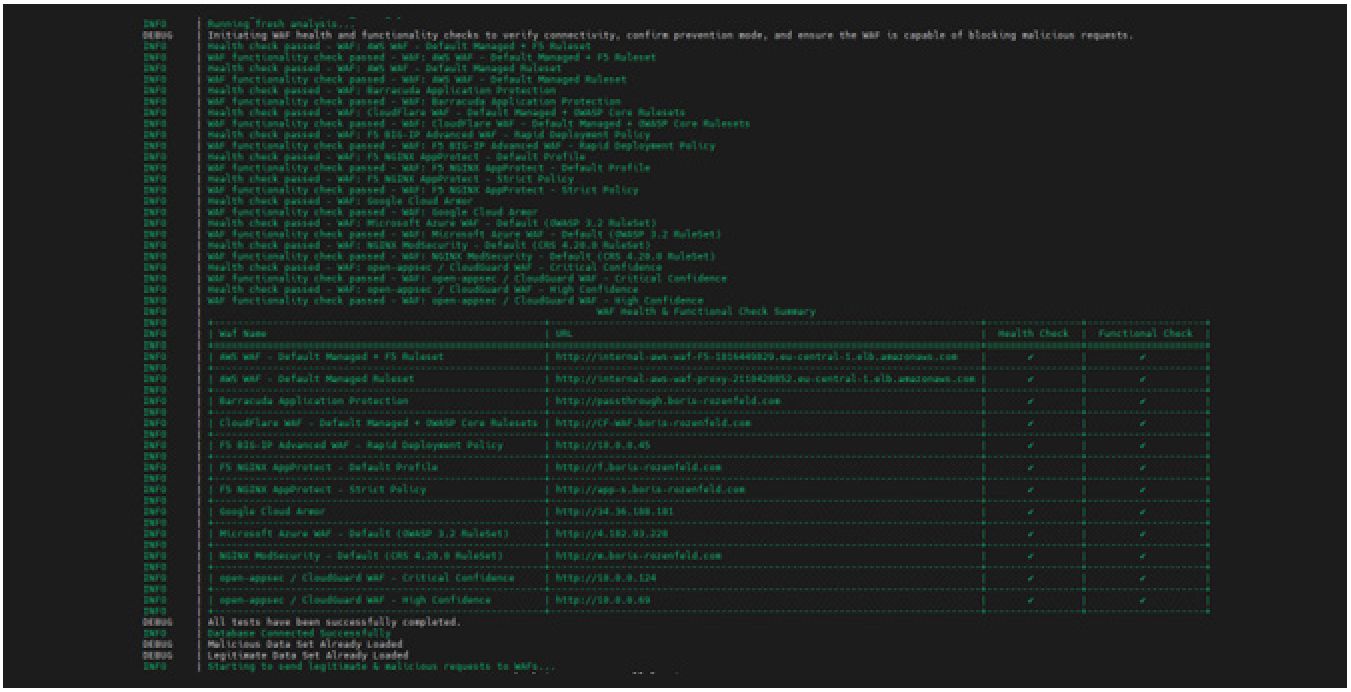
For an in-depth view of each malicious payload utilized in this study, including specific parameters and corresponding attack types, refer to this link.

Combined, the **Legitimate and Malicious Requests datasets** present a detailed perspective on how each WAF solution handles traffic in the real world, thereby providing valuable insights into their efficacy and Detection Quality.

TOOLING

As before, to ensure transparency and reproducibility, the tool is made available to the public here.

During the initial phase, the tool conducts a dual-layer health check for each WAF. This process first validates connectivity to each WAF, ensuring system communication. It then checks that each WAF is set to prevention mode, confirming its ability to actively block malicious requests.



```
INFO    | Running fresh analysis...
DEBUG   | Initiating WAF health and functionality checks to verify connectivity, confirm prevention mode, and ensure the WAF is capable of blocking malicious requests.
INFO    | Health check passed - WAF: AWS WAF - Default Managed + F5 Ruleset
INFO    | WAF functionality check passed - WAF: AWS WAF - Default Managed + F5 Ruleset
INFO    | Health check passed - WAF: AWS WAF - Default Managed Ruleset
INFO    | WAF functionality check passed - WAF: AWS WAF - Default Managed Ruleset
INFO    | Health check passed - WAF: Barracuda Application Protection
INFO    | WAF functionality check passed - WAF: Barracuda Application Protection
INFO    | Health check passed - WAF: CloudFlare WAF - Default Managed + OWASP Core Rulesets
INFO    | WAF functionality check passed - WAF: CloudFlare WAF - Default Managed + OWASP Core Rulesets
INFO    | Health check passed - WAF: F5 BIG-IP Advanced WAF - Rapid Deployment Policy
INFO    | WAF functionality check passed - WAF: F5 BIG-IP Advanced WAF - Rapid Deployment Policy
INFO    | Health check passed - WAF: F5 NGINX AppProtect - Default Profile
INFO    | WAF functionality check passed - WAF: F5 NGINX AppProtect - Default Profile
INFO    | Health check passed - WAF: F5 NGINX AppProtect - Strict Policy
INFO    | WAF functionality check passed - WAF: F5 NGINX AppProtect - Strict Policy
INFO    | Health check passed - WAF: Google Cloud Armor
INFO    | WAF functionality check passed - WAF: Google Cloud Armor
INFO    | Health check passed - WAF: Microsoft Azure WAF - Default (OWASP 3.2 RuleSet)
INFO    | WAF functionality check passed - WAF: Microsoft Azure WAF - Default (OWASP 3.2 RuleSet)
INFO    | Health check passed - WAF: NGINX ModSecurity - Default (CRS 4.20.0 RuleSet)
INFO    | WAF functionality check passed - WAF: NGINX ModSecurity - Default (CRS 4.20.0 RuleSet)
INFO    | Health check passed - WAF: open-appsec / CloudGuard WAF - Critical Confidence
INFO    | WAF functionality check passed - WAF: open-appsec / CloudGuard WAF - Critical Confidence
INFO    | Health check passed - WAF: open-appsec / CloudGuard WAF - High Confidence
INFO    | WAF functionality check passed - WAF: open-appsec / CloudGuard WAF - High Confidence
INFO    |                                                      WAF Health & Functional Check Summary
INFO    | +-----------------------------------------------+-------------------------------------------------------------------+--------------+------------------+
INFO    | | Waf Name                                      | URL                                                               | Health Check | Functional Check |
INFO    | +===============================================+===================================================================+==============+==================+
INFO    | | AWS WAF - Default Managed + F5 Ruleset        | http://internal-aws-waf-F5-1816449829.eu-central-1.elb.amazonaws.com    |      ✓       |        ✓         |
INFO    | +-----------------------------------------------+-------------------------------------------------------------------+--------------+------------------+
INFO    | | AWS WAF - Default Managed Ruleset             | http://internal-aws-waf-proxy-2110420052.eu-central-1.elb.amazonaws.com |      ✓       |        ✓         |
INFO    | +-----------------------------------------------+-------------------------------------------------------------------+--------------+------------------+
INFO    | | Barracuda Application Protection              | http://passthrough.boris-rozenfeld.com                            |      ✓       |        ✓         |
INFO    | +-----------------------------------------------+-------------------------------------------------------------------+--------------+------------------+
INFO    | | CloudFlare WAF - Default Managed + OWASP Core Rulesets | http://CF-WAF.boris-rozenfeld.com                        |      ✓       |        ✓         |
INFO    | +-----------------------------------------------+-------------------------------------------------------------------+--------------+------------------+
INFO    | | F5 BIG-IP Advanced WAF - Rapid Deployment Policy | http://10.0.0.45                                               |      ✓       |        ✓         |
INFO    | +-----------------------------------------------+-------------------------------------------------------------------+--------------+------------------+
INFO    | | F5 NGINX AppProtect - Default Profile         | http://f.boris-rozenfeld.com                                      |      ✓       |        ✓         |
INFO    | +-----------------------------------------------+-------------------------------------------------------------------+--------------+------------------+
INFO    | | F5 NGINX AppProtect - Strict Policy           | http://app-s.boris-rozenfeld.com                                  |      ✓       |        ✓         |
INFO    | +-----------------------------------------------+-------------------------------------------------------------------+--------------+------------------+
INFO    | | Google Cloud Armor                            | http://34.36.108.181                                              |      ✓       |        ✓         |
INFO    | +-----------------------------------------------+-------------------------------------------------------------------+--------------+------------------+
INFO    | | Microsoft Azure WAF - Default (OWASP 3.2 RuleSet) | http://4.182.93.228                                           |      ✓       |        ✓         |
INFO    | +-----------------------------------------------+-------------------------------------------------------------------+--------------+------------------+
INFO    | | NGINX ModSecurity - Default (CRS 4.20.0 RuleSet) | http://m.boris-rozenfeld.com                                   |      ✓       |        ✓         |
INFO    | +-----------------------------------------------+-------------------------------------------------------------------+--------------+------------------+
INFO    | | open-appsec / CloudGuard WAF - Critical Confidence | http://10.0.0.124                                            |      ✓       |        ✓         |
INFO    | +-----------------------------------------------+-------------------------------------------------------------------+--------------+------------------+
INFO    | | open-appsec / CloudGuard WAF - High Confidence | http://10.0.0.69                                                 |      ✓       |        ✓         |
INFO    | +-----------------------------------------------+-------------------------------------------------------------------+--------------+------------------+
DEBUG   | All tests have been successfully completed.
INFO    | Database Connected Successfully
DEBUG   | Malicious Data Set Already Loaded
DEBUG   | Legitimate Data Set Already Loaded
INFO    | Starting to send legitimate & malicious requests to WAFs...
```

The responses from each request sent by the test tool to the WAFs were systematically logged in a dedicated database for further analysis.

The database used for this test was a DuckDB. However, the test tool is designed to be flexible. Readers can configure it to work with any SQL database of their preference by adjusting the settings in the config.py file.

Following the data collection phase, the performance metrics, including False Positive rates, False Negative rates, and Balanced Accuracy results, were calculated. This was done by executing specific SQL queries against the data in the database.

Based on user feedback, we have updated this year the testing tool to improve usability:

1. **Containerized Environment:** A pre-configured Docker image allows users to reproduce these results with a single command, eliminating environment dependency issues.
2. **Sampling Mode:** A -quick flag allows for rapid health checks by running random samples (5%) of the dataset.
3. **Report Generation:** The tool now includes a report generator that processes the DuckDB results into a detailed PDF, breaking down performance by attack family (SQLi, RCE, Padding) and offering visual score comparisons.

COMPARISON METRICS

To quantify the efficacy of each WAF, we use statistical measures. These include **Security Quality** (also known as Sensitivity or True Positive Rate), **Detection Quality** (also known as Specificity or True Negative Rate), and **Balanced Accuracy**.

Security Quality, also known as the true positive rate (TPR), measures the proportion of actual positives correctly identified. In other words, it assesses the WAF's ability to correctly detect and block malicious requests.

$$TPR = \frac{TP}{P} = \frac{TP}{TP + FN} = 1 - FNR$$

Detection Quality, or the true negative rate (TNR), quantifies the proportion of actual negatives correctly identified. This pertains to the WAF's capacity to correctly allow legitimate traffic to pass.

$$TNR = \frac{TN}{PN} = \frac{TN}{TN + FP} = 1 - FPR$$

Balanced Accuracy (BA), an especially crucial metric in this study, provides a balanced measurement by considering both metrics. It is calculated as the arithmetic mean of TPR and TNR. In other words, it provides a more balanced measure between True Positives and True Negatives, irrespective of their proportions in the data sets.

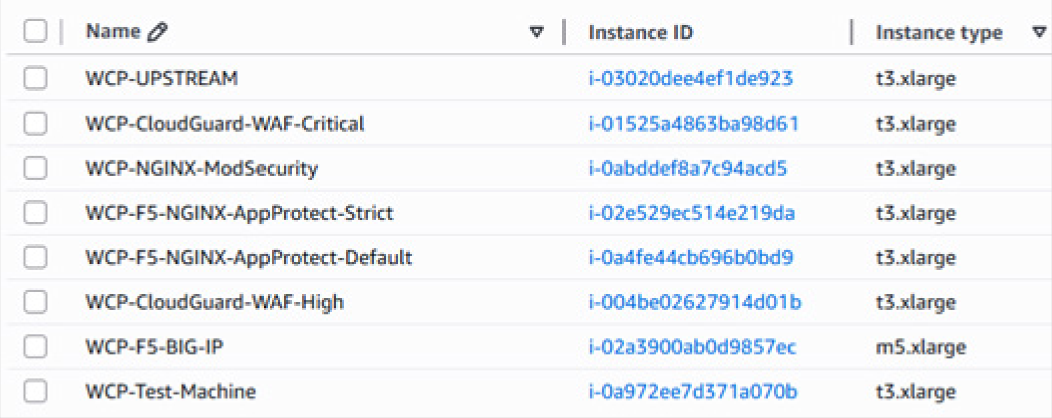
$$BA = \frac{TPR + TNR}{2}$$

This choice of metrics is fundamental, as we aim to assess not just the WAF's ability to block malicious traffic but also to allow legitimate traffic. Most importantly, we want to evaluate the overall balance between these two abilities, given that both are critical for a real-world production system. Thus, we not only examine the number of attacks each WAF can correctly identify and block but also scrutinize the number of legitimate requests it correctly allows. A WAF with high TPR but low TNR might block most attacks but at the cost of blocking too many legitimate requests, leading to poor user experience. Conversely, a WAF with high TNR but low TPR might allow most legitimate requests but fail to block a significant number of attacks, compromising the security of the system. **Therefore, the optimal WAF solution should strike a balance, exhibiting high performance on both TPR and TNR, which is aptly represented by Balanced Accuracy.**

TEST ENVIRONMENT

The test includes both, products that are deployed as standard software and products available as SaaS. The standard software products were staged within Amazon Web Services (AWS). The main testing apparatus was an AWS EC2 instance, housed in a separate VPC. This facilitated the simulation of a real-world production environment while keeping the testing isolated from external influences.

To maintain the integrity of the test and ensure that performance wasn't a distorting factor, all embedded WAF solutions were hosted on AWS xlarge instances. These instances are equipped with 4 virtual CPUs and 16GB of RAM, providing ample computational power and memory resources well beyond what is typically required for standard operations. This configuration was deliberately chosen to eliminate any possibility of hardware constraints influencing the outcome of the WAF comparison, thereby ensuring the results accurately reflect the inherent capabilities of each solution.



□	Name	Instance ID	Instance type
□	WCP-UPSTREAM	i-03020dee4ef1de923	t3.xlarge
□	WCP-CloudGuard-WAF-Critical	i-01525a4863ba98d61	t3.xlarge
□	WCP-NGINX-ModSecurity	i-0abddef8a7c94acd5	t3.xlarge
□	WCP-F5-NGINX-AppProtect-Strict	i-02e529ec514e219da	t3.xlarge
□	WCP-F5-NGINX-AppProtect-Default	i-0a4fe44cb696b0bd9	t3.xlarge
□	WCP-CloudGuard-WAF-High	i-004be02627914d01b	t3.xlarge
□	WCP-F5-BIG-IP	i-02a3900ab0d9857ec	m5.xlarge
□	WCP-Test-Machine	i-0a972ee7d371a070b	t3.xlarge

FINDINGS

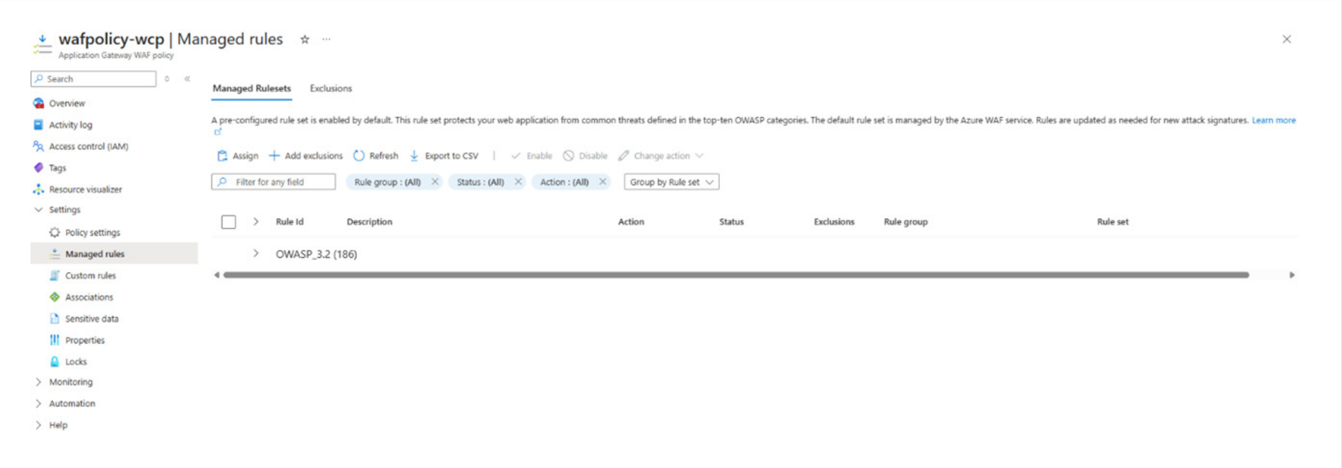
In this section, we describe the configuration of each product tested and the score of the **Security Quality** or True Positive Rate measurement (higher is better) and **Detection Quality** or False Positive Rate measurement (lower is better) as well as the **Balanced Accuracy** score (higher is better).

In the test we used the Default Profile Settings of each product without any tuning and when available also an additional Profile that allows for the highest Security Quality available in the product.

MICROSOFT AZURE WAF

Azure WAF is a cloud-based service implementing ModSecurity with OWASP Core RuleSet. The Microsoft Azure WAF was configured with the Default suggested OWASP CRS 3.2 ruleset.

For the second year in a row, we see that Azure WAF still relies on the old OWASP CRS 3.2 ruleset. Even though the industry moved to version 4.x a long time ago, Microsoft has not updated the core rules for more than two years.



A pre-configured rule set is enabled by default. This rule set protects your web application from common threats defined in the top-ten OWASP categories. The default rule set is managed by the Azure WAF service. Rules are updated as needed for new attack signatures. Learn more

Rule Id	Description	Action	Status	Exclusions	Rule group	Rule set
OWASP_3.2 (186)						

The results were as follows:

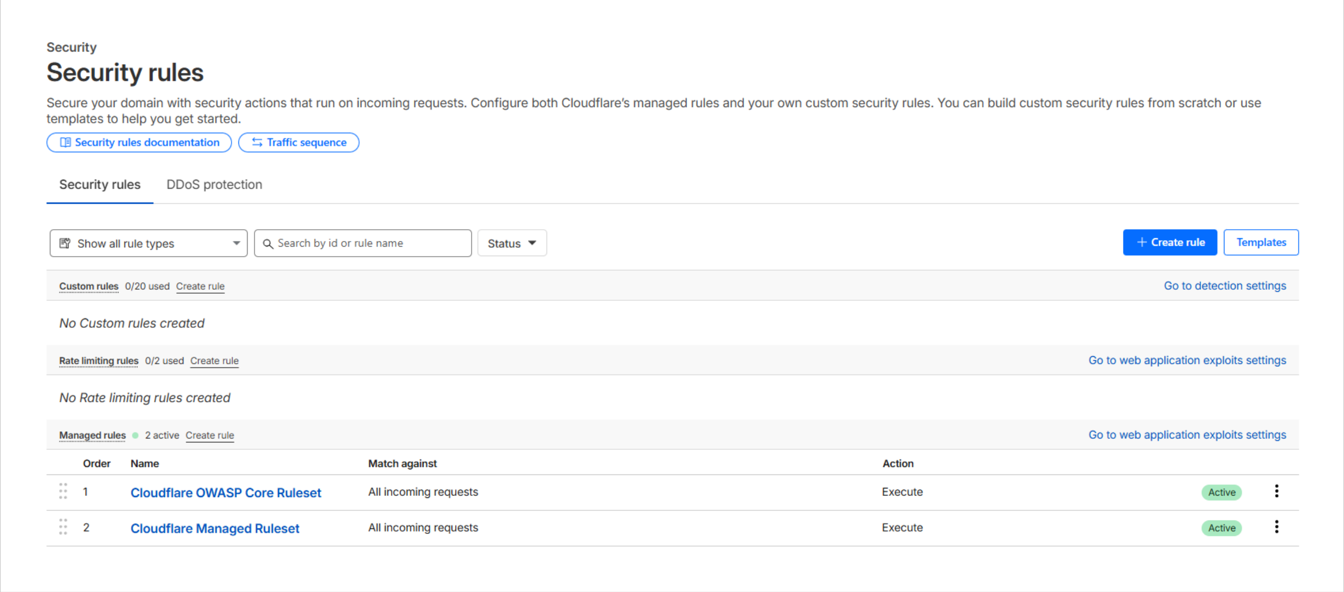
Security Quality (True Positive Rate): **97.537%**

Detection Quality (False Positive Rate): **54.412%**

Balanced Accuracy: **71.562%**

CLOUDFLARE WAF

CloudFlare WAF is a cloud-based service based on ModSecurity with OWASP Core RuleSet. CloudFlare provides a Managed ruleset as well as a full OWASP Core RuleSet. We tested the product activating both rulesets.



Security

Security rules

Secure your domain with security actions that run on incoming requests. Configure both Cloudflare's managed rules and your own custom security rules. You can build custom security rules from scratch or use templates to help you get started.

Custom rules 0/20 used Create rule — Go to detection settings

No Custom rules created

Rate limiting rules 0/2 used Create rule — Go to web application exploits settings

No Rate limiting rules created

Managed rules ● 2 active Create rule — Go to web application exploits settings

Order	Name	Match against	Action	
1	Cloudflare OWASP Core Ruleset	All incoming requests	Execute	Active
2	Cloudflare Managed Ruleset	All incoming requests	Execute	Active

The results were as follows:

Security Quality (True Positive Rate): **63.462%**

Detection Quality (False Positive Rate): **0.06%**

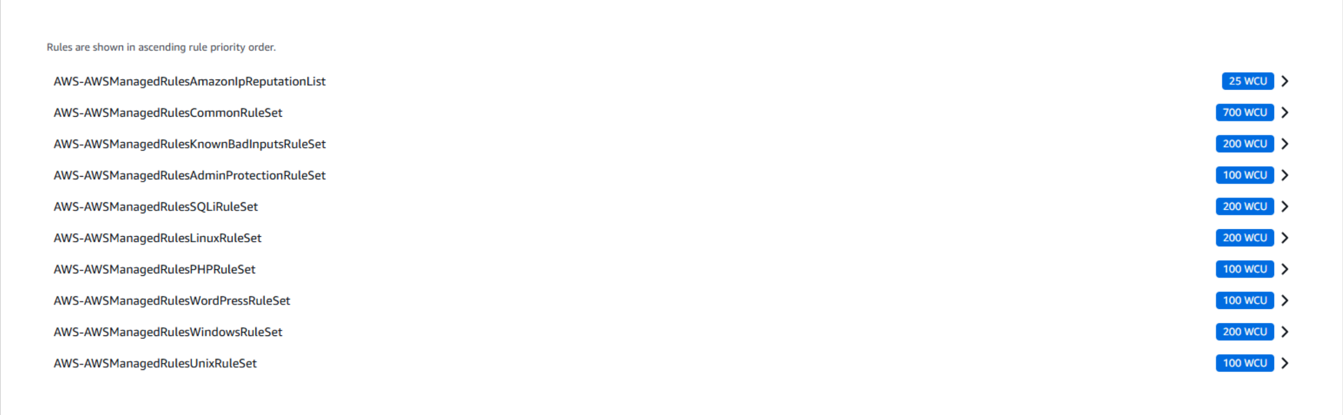
Balanced Accuracy: **81.701%**

AWS WAF

AWS WAF is a cloud-based service implementing ModSecurity. AWS provides a default Managed ruleset and optional additional paid-for rulesets from leading vendors such as F5.

We tested the service in two configurations.

AWS WAF – AWS managed ruleset:



Rules are shown in ascending rule priority order.

AWS-AWSManagedRulesAmazonIpReputationList	25 WCU
AWS-AWSManagedRulesCommonRuleSet	700 WCU
AWS-AWSManagedRulesKnownBadInputsRuleSet	200 WCU
AWS-AWSManagedRulesAdminProtectionRuleSet	100 WCU
AWS-AWSManagedRulesSQLiRuleSet	200 WCU
AWS-AWSManagedRulesLinuxRuleSet	200 WCU
AWS-AWSManagedRulesPHPRuleSet	100 WCU
AWS-AWSManagedRulesWordPressRuleSet	100 WCU
AWS-AWSManagedRulesWindowsRuleSet	200 WCU
AWS-AWSManagedRulesUnixRuleSet	100 WCU

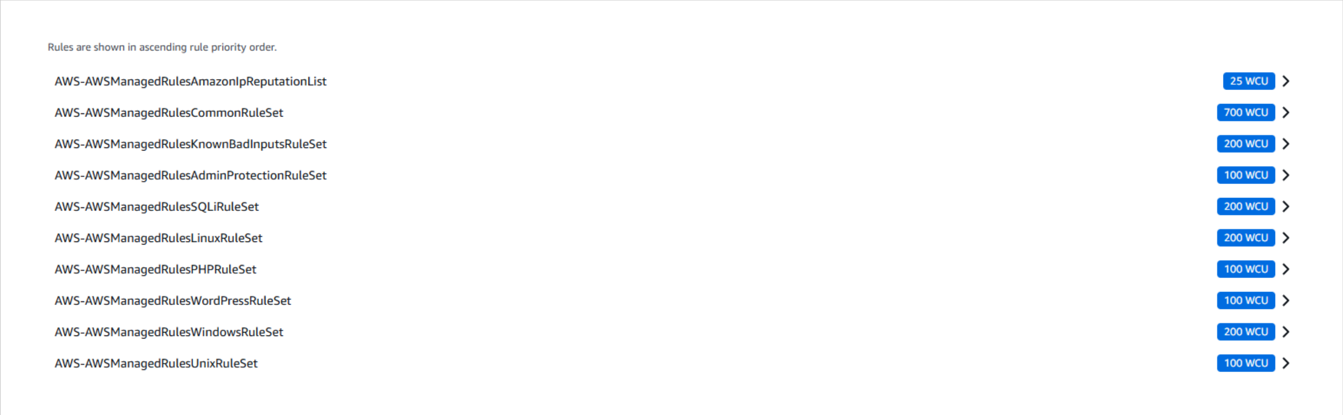
The results were as follows:

Security Quality
(True Positive Rate):
79.891%

Detection Quality
(False Positive Rate):
6.046%

Balanced Accuracy:
86.922%

AWS WAF – AWS managed ruleset plus F5 Rules:



Rules are shown in ascending rule priority order.

AWS-AWSManagedRulesAmazonIpReputationList	25 WCU
AWS-AWSManagedRulesCommonRuleSet	700 WCU
AWS-AWSManagedRulesKnownBadInputsRuleSet	200 WCU
AWS-AWSManagedRulesAdminProtectionRuleSet	100 WCU
AWS-AWSManagedRulesSQLiRuleSet	200 WCU
AWS-AWSManagedRulesLinuxRuleSet	200 WCU
AWS-AWSManagedRulesPHPRuleSet	100 WCU
AWS-AWSManagedRulesWordPressRuleSet	100 WCU
AWS-AWSManagedRulesWindowsRuleSet	200 WCU
AWS-AWSManagedRulesUnixRuleSet	100 WCU

The results were as follows:

Security Quality
(True Positive Rate):
80.445%

Detection Quality
(False Positive Rate):
6.133%

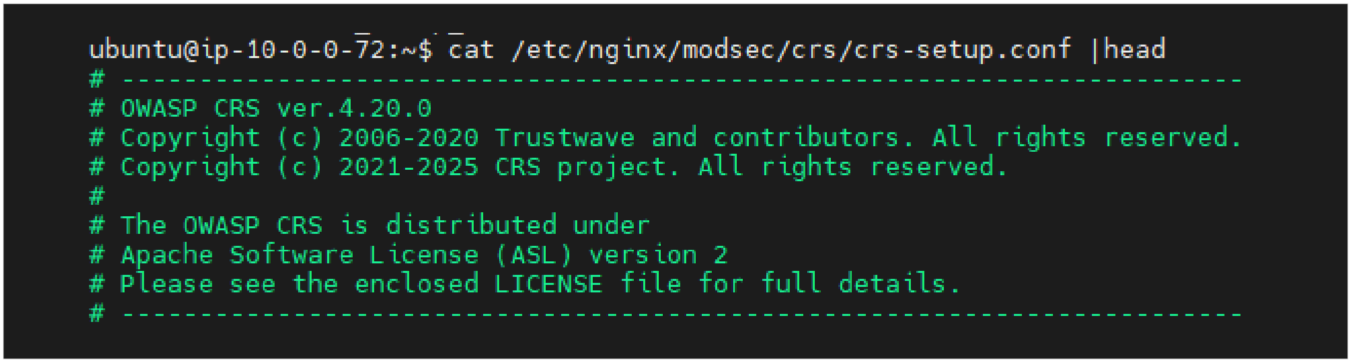
Balanced Accuracy:
87.156%

NGINX MODSECURITY

ModSecurity has been the most popular open-source WAF Engine available in the market for 20 years and is signatures-based. Many of the solutions tested here use it as a base.

ModSecurity has reached its End-of-Life as of July 2024. Last year, the OWASP Core Rule Set (CRS) received a major update, which impacted effectiveness. The updated CRS focused more on security, leading to an improvement in Security Quality but a decrease in Detection Quality, reflecting a shift in balance towards blocking threats over allowing legitimate traffic.

We tested NGINX with ModSecurity Core Rule Set v4.20.0 (latest version) Default settings:



```
ubuntu@ip-10-0-0-72:~$ cat /etc/nginx/modsec/crs/crs-setup.conf |head
# ------------------------------------------------------------------------
# OWASP CRS ver.4.20.0
# Copyright (c) 2006-2020 Trustwave and contributors. All rights reserved.
# Copyright (c) 2021-2025 CRS project. All rights reserved.
#
# The OWASP CRS is distributed under
# Apache Software License (ASL) version 2
# Please see the enclosed LICENSE file for full details.
# ------------------------------------------------------------------------
```

The results were as follows:

Security Quality
(True Positive Rate):
92.257%

Detection Quality
(False Positive Rate):
18.638%

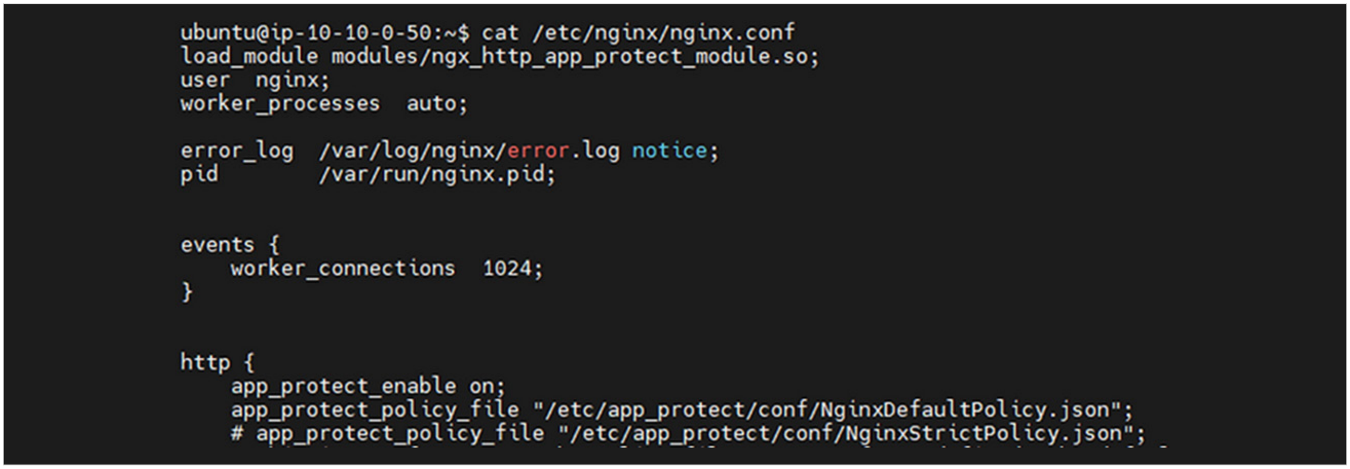
Balanced Accuracy:
86.81%

F5 NGINX APPPROTECT

NGINX AppProtect WAF is a paid add-on to NGINX Plus and NGINX Plus Ingress based on the traditional F5 signature-based WAF solution. The AppProtect WAF comes with two policies - Default and Strict. The Default policy provides OWASP-Top-10 protection. The Strict policy is recommended by NGINX for "protecting sensitive applications that require more security but with a higher risk of false positives." It includes over 6000 signatures.

We tested the product with both policies.

NGINX AppProtect Default was configured as follows:



```
ubuntu@ip-10-10-0-50:~$ cat /etc/nginx/nginx.conf
load_module modules/ngx_http_app_protect_module.so;
user  nginx;
worker_processes  auto;

error_log  /var/log/nginx/error.log notice;
pid        /var/run/nginx.pid;


events {
    worker_connections  1024;
}


http {
    app_protect_enable on;
    app_protect_policy_file "/etc/app_protect/conf/NginxDefaultPolicy.json";
    # app_protect_policy_file "/etc/app_protect/conf/NginxStrictPolicy.json";
```

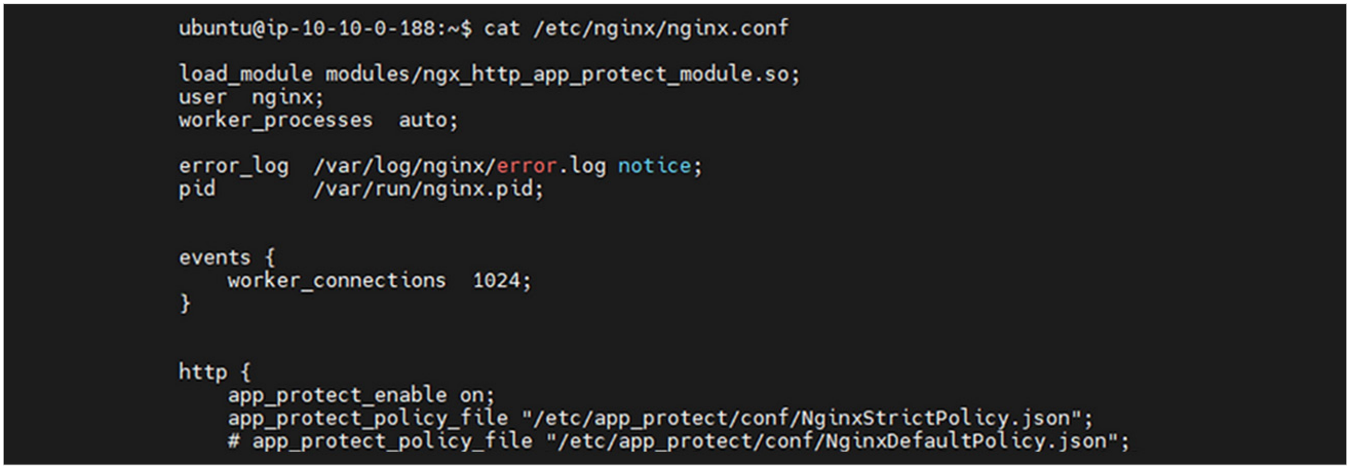
The results were as follows:

Security Quality
(True Positive Rate):
91.009%

Detection Quality
(False Positive Rate):
2.868%

Balanced Accuracy:
94.071%

NGINX AppProtect Strict WAF was configured as follows:



```
ubuntu@ip-10-10-0-188:~$ cat /etc/nginx/nginx.conf

load_module modules/ngx_http_app_protect_module.so;
user  nginx;
worker_processes  auto;

error_log  /var/log/nginx/error.log notice;
pid        /var/run/nginx.pid;


events {
    worker_connections  1024;
}


http {
    app_protect_enable on;
    app_protect_policy_file "/etc/app_protect/conf/NginxStrictPolicy.json";
    # app_protect_policy_file "/etc/app_protect/conf/NginxDefaultPolicy.json";
```

The results were as follows:

Security Quality
(True Positive Rate):
97.907%

Detection Quality
(False Positive Rate):
25.4%

Balanced Accuracy:
86.254%

CLOUDGUARD WAF

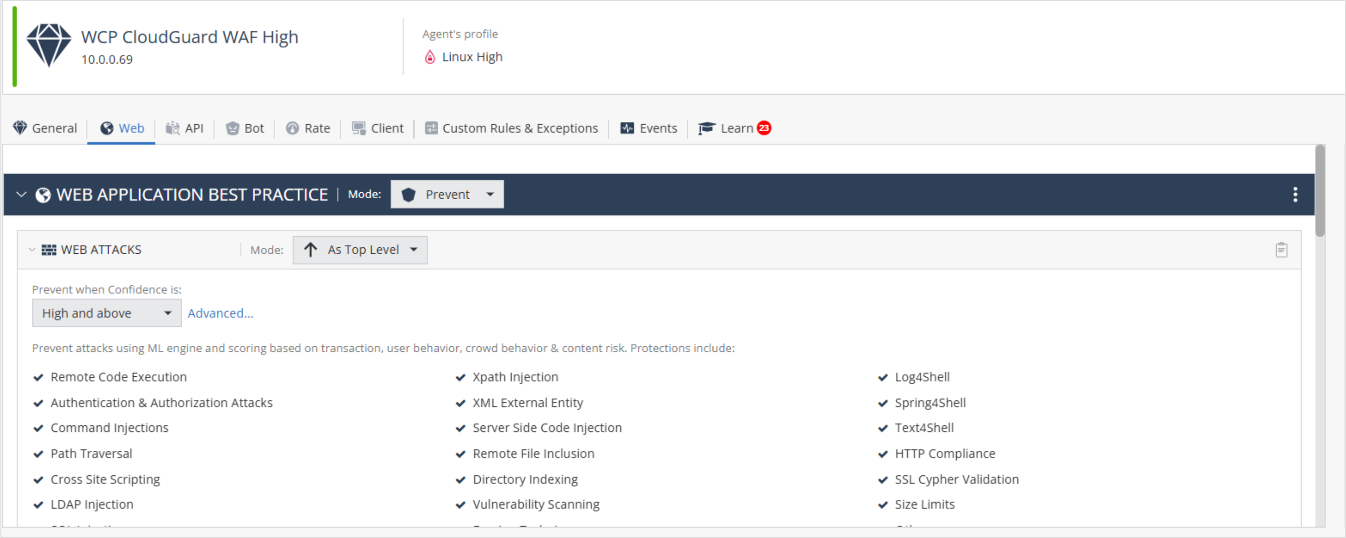
CloudGuard WAF by Check Point is a machine-learning-based WAF that uses supervised and unsupervised machine-learning models to determine whether traffic is malicious.

We tested the product in two configurations (Default and Critical) using out-of-the-box settings with no learning period.

This year, CloudGuard WAF achieved even better results than last year thanks to the new released v2.1 of the ML Engine, which features an updated supervised learning methodology alongside a modified scoring system, specifically designed to provide greater accuracy across both small and large indicator sets.

- In the **Default Profile**, Balanced Accuracy improved from **98.966%** to **99.283%** (False Positive Rate improved from **1.436%** to **0.994%** and Security Quality improved from **99.368%** to **99.56%**)
- In the **Critical Profile** Balanced Accuracy improved from **99.139%** to **99.453%** (False Positive Rate improved from **0.81%** to **0.563%** and Security Quality improved from **99.087%** to **99.47%**)

Default - activate protections when Confidence is set to "High and above":



WCP CloudGuard WAF High
10.0.0.69

Agent's profile
Linux High

WEB APPLICATION BEST PRACTICE | Mode: Prevent

WEB ATTACKS | Mode: As Top Level

Prevent when Confidence is:
High and above
Advanced...

Prevent attacks using ML engine and scoring based on transaction, user behavior, crowd behavior & content risk. Protections include:

- ✓ Remote Code Execution
- ✓ Authentication & Authorization Attacks
- ✓ Command Injections
- ✓ Path Traversal
- ✓ Cross Site Scripting
- ✓ LDAP Injection
- ✓ Xpath Injection
- ✓ XML External Entity
- ✓ Server Side Code Injection
- ✓ Remote File Inclusion
- ✓ Directory Indexing
- ✓ Vulnerability Scanning
- ✓ Log4Shell
- ✓ Spring4Shell
- ✓ Text4Shell
- ✓ HTTP Compliance
- ✓ SSL Cypher Validation
- ✓ Size Limits

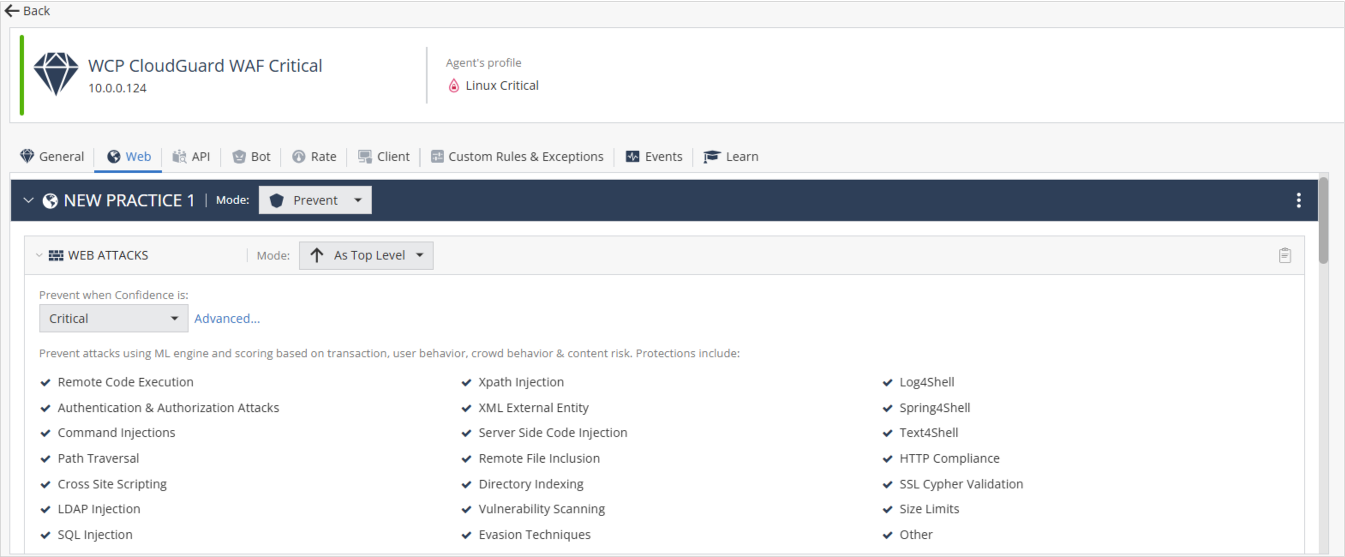
The results were as follows:

Security Quality (True Positive Rate): **99.56%**

Detection Quality (False Positive Rate): **0.994%**

Balanced Accuracy: **99.283%**

Critical - activate protections when Confidence is set to “Critical”:



WCP CloudGuard WAF Critical
10.0.0.124

Agent's profile: Linux Critical

NEW PRACTICE 1 | Mode: Prevent

WEB ATTACKS | Mode: As Top Level

Prevent when Confidence is: Critical

Advanced...

Prevent attacks using ML engine and scoring based on transaction, user behavior, crowd behavior & content risk. Protections include:

- ✓ Remote Code Execution
- ✓ Authentication & Authorization Attacks
- ✓ Command Injections
- ✓ Path Traversal
- ✓ Cross Site Scripting
- ✓ LDAP Injection
- ✓ SQL Injection
- ✓ Xpath Injection
- ✓ XML External Entity
- ✓ Server Side Code Injection
- ✓ Remote File Inclusion
- ✓ Directory Indexing
- ✓ Vulnerability Scanning
- ✓ Evasion Techniques
- ✓ Log4Shell
- ✓ Spring4Shell
- ✓ Text4Shell
- ✓ HTTP Compliance
- ✓ SSL Cypher Validation
- ✓ Size Limits
- ✓ Other

The results were as follows:

Security Quality
(True Positive Rate):
99.47%

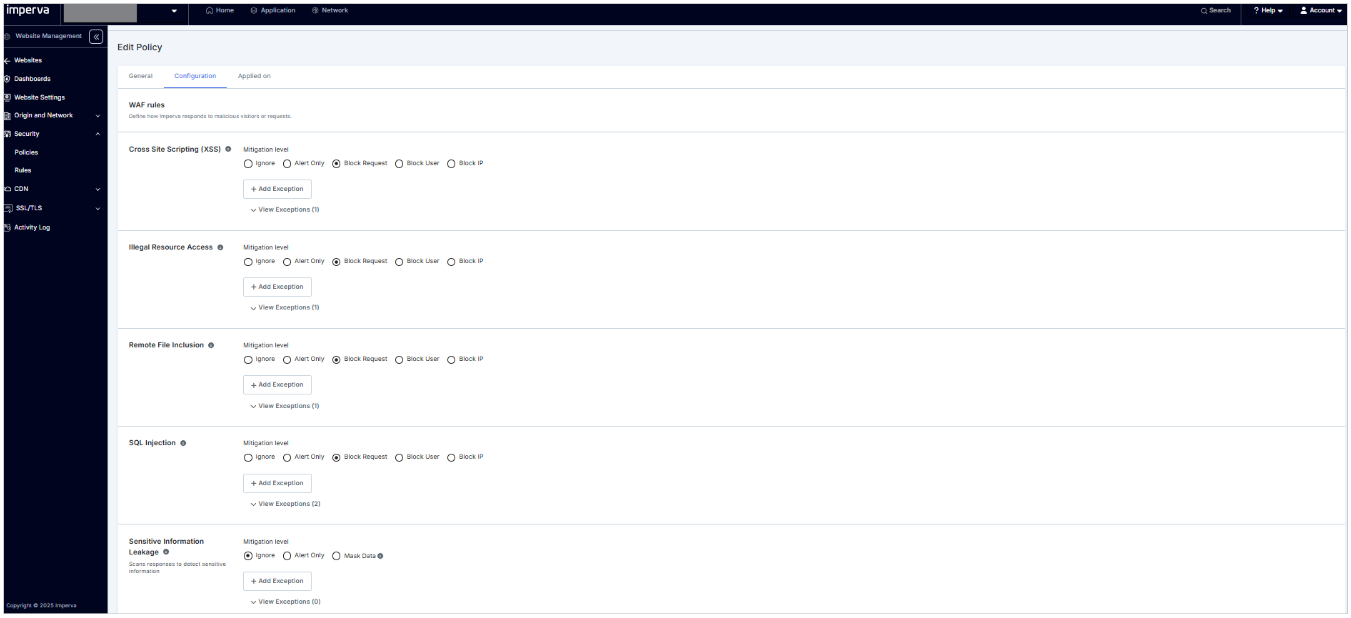
Detection Quality
(False Positive Rate):
0.563%

Balanced Accuracy:
99.453%

IMPERVA CLOUD WAF (2024-2025)

Imperva Cloud WAF is one of the well-known WAF solutions in the industry. We tested it in the Default configuration.

***Note:** The results displayed for Imperva Cloud WAF are based on the 2024-2025 test data. We were unable to update the score for this year due to the specific blocking behavior described in the introduction*



Edit Policy

General | Configuration | Applied on

WAF rules

Define how Imperva responds to malicious visitors or requests.

Cross Site Scripting (XSS)
Mitigation level
○ Ignore ○ Alert Only ● Block Request ○ Block User ○ Block IP
+ Add Exception
View Exceptions (1)

Illegal Resource Access
Mitigation level
○ Ignore ○ Alert Only ● Block Request ○ Block User ○ Block IP
+ Add Exception
View Exceptions (1)

Remote File Inclusion
Mitigation level
○ Ignore ○ Alert Only ● Block Request ○ Block User ○ Block IP
+ Add Exception
View Exceptions (1)

SQL Injection
Mitigation level
○ Ignore ○ Alert Only ● Block Request ○ Block User ○ Block IP
+ Add Exception
View Exceptions (2)

Sensitive Information Leakage
Scans responses to detect sensitive information
Mitigation level
● Ignore ○ Alert Only ○ Mask Data
+ Add Exception
View Exceptions (0)

The results were as follows:

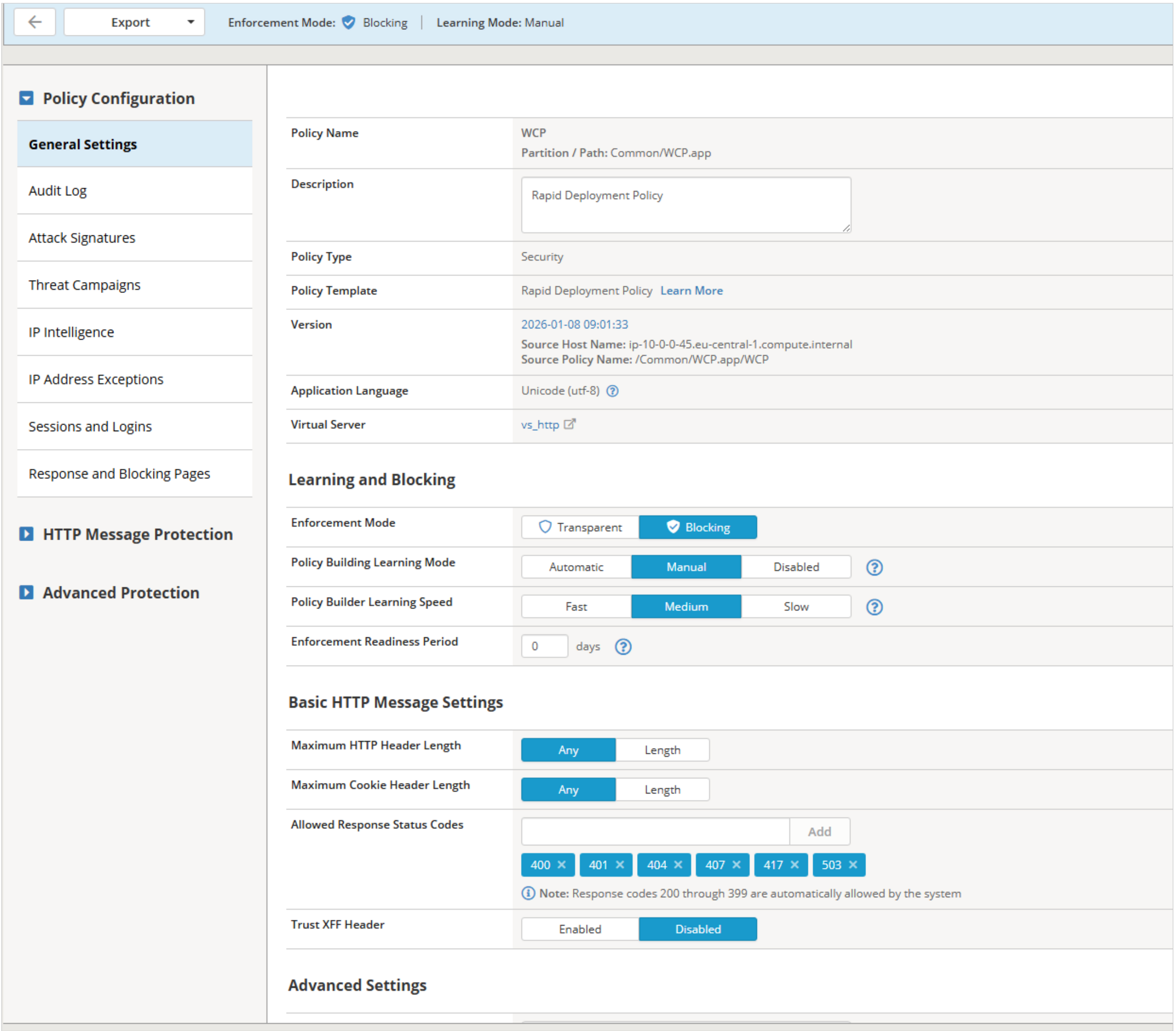
Security Quality
(True Positive Rate):
11.97%

Detection Quality
(False Positive Rate):
0.009%

Balanced Accuracy:
55.981%

F5 BIG-IP ADVANCED WAF

We tested F5 BIG-IP Advanced WAF using the Rapid Deployment Policy configuration. The solution was deployed on a virtual machine within AWS.



Enforcement Mode: Blocking | Learning Mode: Manual

Policy Name	WCP Partition / Path: Common/WCP.app
Description	Rapid Deployment Policy
Policy Type	Security
Policy Template	Rapid Deployment Policy Learn More
Version	2026-01-08 09:01:33 Source Host Name: ip-10-0-0-45.eu-central-1.compute.internal Source Policy Name: /Common/WCP.app/WCP
Application Language	Unicode (utf-8)
Virtual Server	vs_http

Learning and Blocking

Enforcement Mode	Transparent [✓ Blocking]
Policy Building Learning Mode	Automatic [✓ Manual] Disabled
Policy Builder Learning Speed	Fast [✓ Medium] Slow
Enforcement Readiness Period	0 days

Basic HTTP Message Settings

Maximum HTTP Header Length	[✓ Any] Length
Maximum Cookie Header Length	[✓ Any] Length
Allowed Response Status Codes	Add 400 ✕ 401 ✕ 404 ✕ 407 ✕ 417 ✕ 503 ✕ Note: Response codes 200 through 399 are automatically allowed by the system
Trust XFF Header	Enabled [✓ Disabled]

Advanced Settings

The results were as follows:

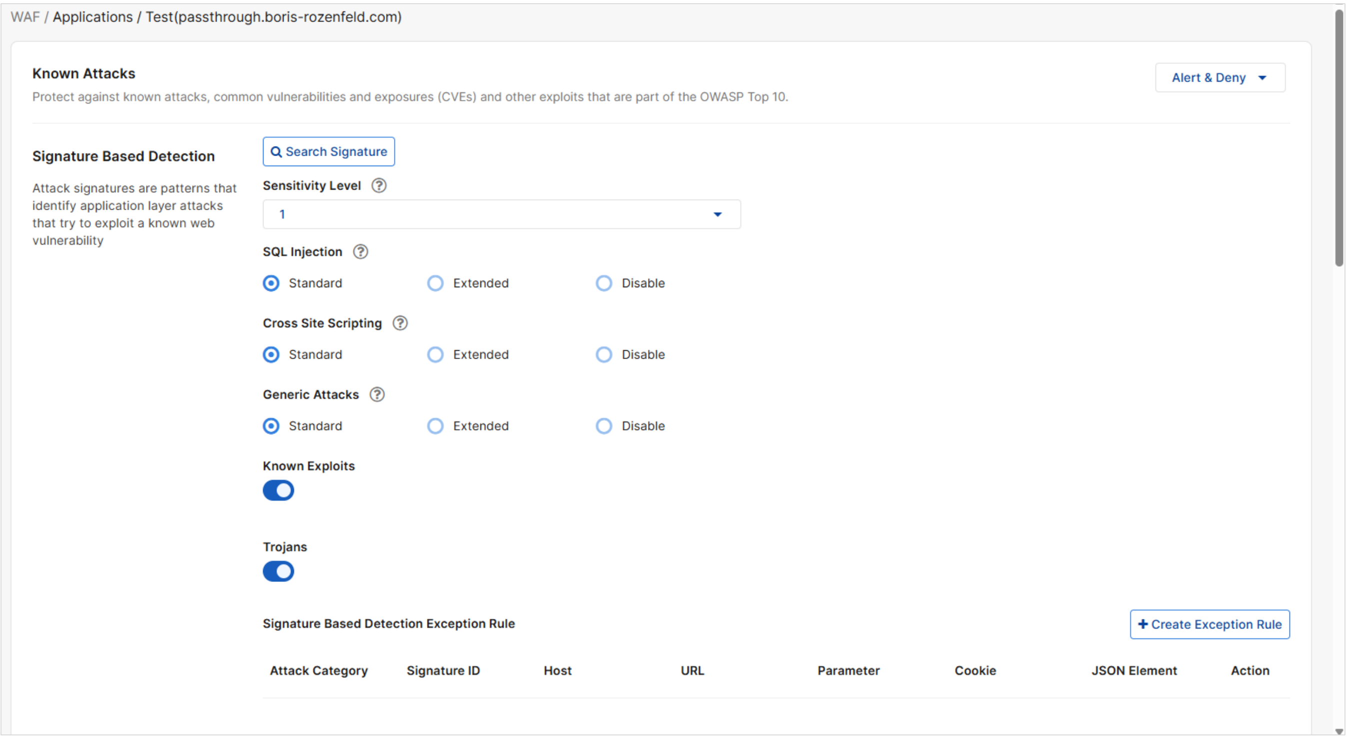
Security Quality (True Positive Rate): **79.039%**

Detection Quality (False Positive Rate): **2.894%**

Balanced Accuracy: **88.072%**

FORTINET FORTIWEB

Fortinet FortiWeb offers several deployment options. For this test, we used the SaaS deployment option to evaluate its performance and protection capabilities directly from the cloud environment.



WAF / Applications / Test(passthrough.boris-rozenfeld.com)

Known Attacks

Protect against known attacks, common vulnerabilities and exposures (CVEs) and other exploits that are part of the OWASP Top 10.

Alert & Deny

Signature Based Detection

Attack signatures are patterns that identify application layer attacks that try to exploit a known web vulnerability

Sensitivity Level: 1

SQL Injection: ● Standard ○ Extended ○ Disable

Cross Site Scripting: ● Standard ○ Extended ○ Disable

Generic Attacks: ● Standard ○ Extended ○ Disable

Known Exploits: ● on

Trojans: ● on

Signature Based Detection Exception Rule

Attack Category	Signature ID	Host	URL	Parameter	Cookie	JSON Element	Action

The results were as follows:

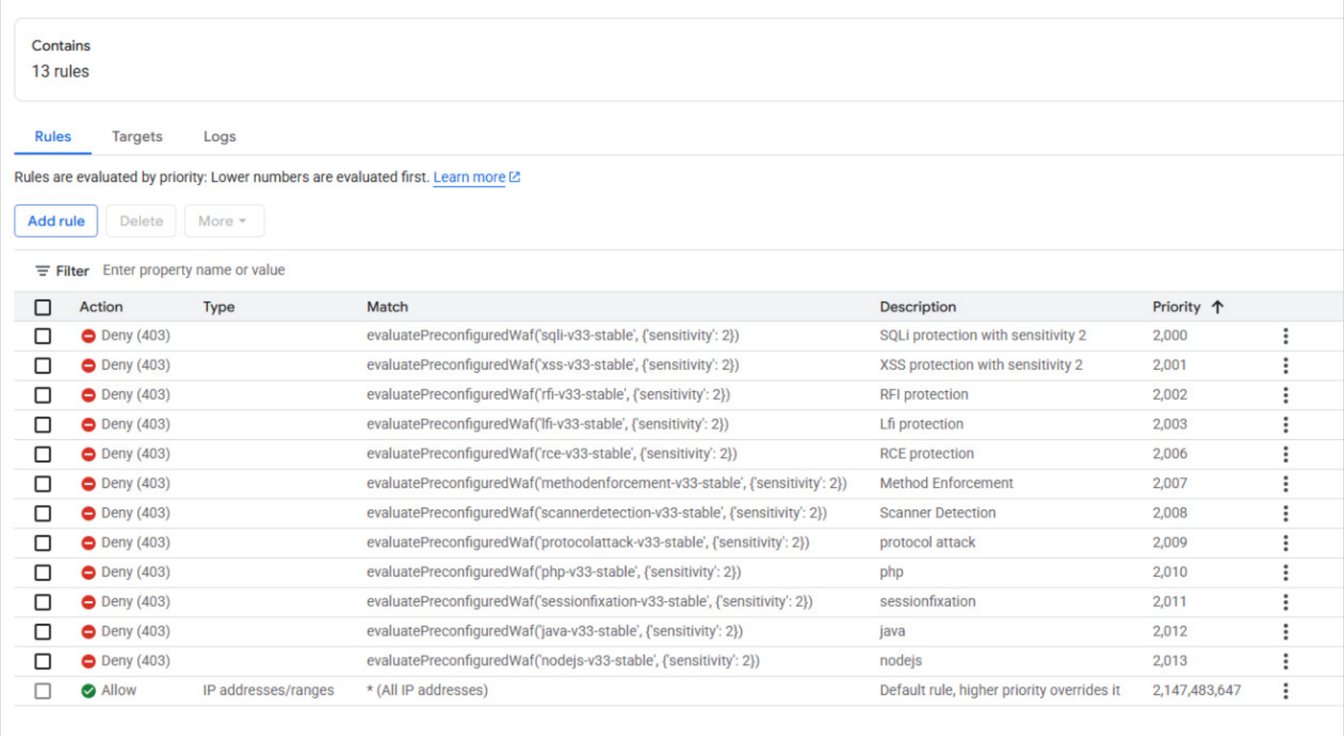
Security Quality (True Positive Rate): **70.143%**

Detection Quality (False Positive Rate): **21.388%**

Balanced Accuracy: **74.378%**

GOOGLE CLOUD ARMOR

Google Cloud Armor is a cloud-based WAF solution that utilizes ModSecurity with the OWASP Core Rule Set (CRS). For this test, we enabled all available rules with the base sensitivity set to level 2.



Contains
13 rules

Rules are evaluated by priority: Lower numbers are evaluated first. Learn more

□	Action	Type	Match	Description	Priority ↑	
□	Deny (403)		evaluatePreconfiguredWaf('sqli-v33-stable', {'sensitivity': 2})	SQLi protection with sensitivity 2	2,000	⋮
□	Deny (403)		evaluatePreconfiguredWaf('xss-v33-stable', {'sensitivity': 2})	XSS protection with sensitivity 2	2,001	⋮
□	Deny (403)		evaluatePreconfiguredWaf('rfi-v33-stable', {'sensitivity': 2})	RFI protection	2,002	⋮
□	Deny (403)		evaluatePreconfiguredWaf('lfi-v33-stable', {'sensitivity': 2})	Lfi protection	2,003	⋮
□	Deny (403)		evaluatePreconfiguredWaf('rce-v33-stable', {'sensitivity': 2})	RCE protection	2,006	⋮
□	Deny (403)		evaluatePreconfiguredWaf('methodenforcement-v33-stable', {'sensitivity': 2})	Method Enforcement	2,007	⋮
□	Deny (403)		evaluatePreconfiguredWaf('scannerdetection-v33-stable', {'sensitivity': 2})	Scanner Detection	2,008	⋮
□	Deny (403)		evaluatePreconfiguredWaf('protocolattack-v33-stable', {'sensitivity': 2})	protocol attack	2,009	⋮
□	Deny (403)		evaluatePreconfiguredWaf('php-v33-stable', {'sensitivity': 2})	php	2,010	⋮
□	Deny (403)		evaluatePreconfiguredWaf('sessionfixation-v33-stable', {'sensitivity': 2})	sessionfixation	2,011	⋮
□	Deny (403)		evaluatePreconfiguredWaf('java-v33-stable', {'sensitivity': 2})	java	2,012	⋮
□	Deny (403)		evaluatePreconfiguredWaf('nodejs-v33-stable', {'sensitivity': 2})	nodejs	2,013	⋮
□	✓ Allow	IP addresses/ranges	* (All IP addresses)	Default rule, higher priority overrides it	2,147,483,647	⋮

The results were as follows:

Security Quality
(True Positive Rate):
91.006%

Detection Quality
(False Positive Rate):
56.999%

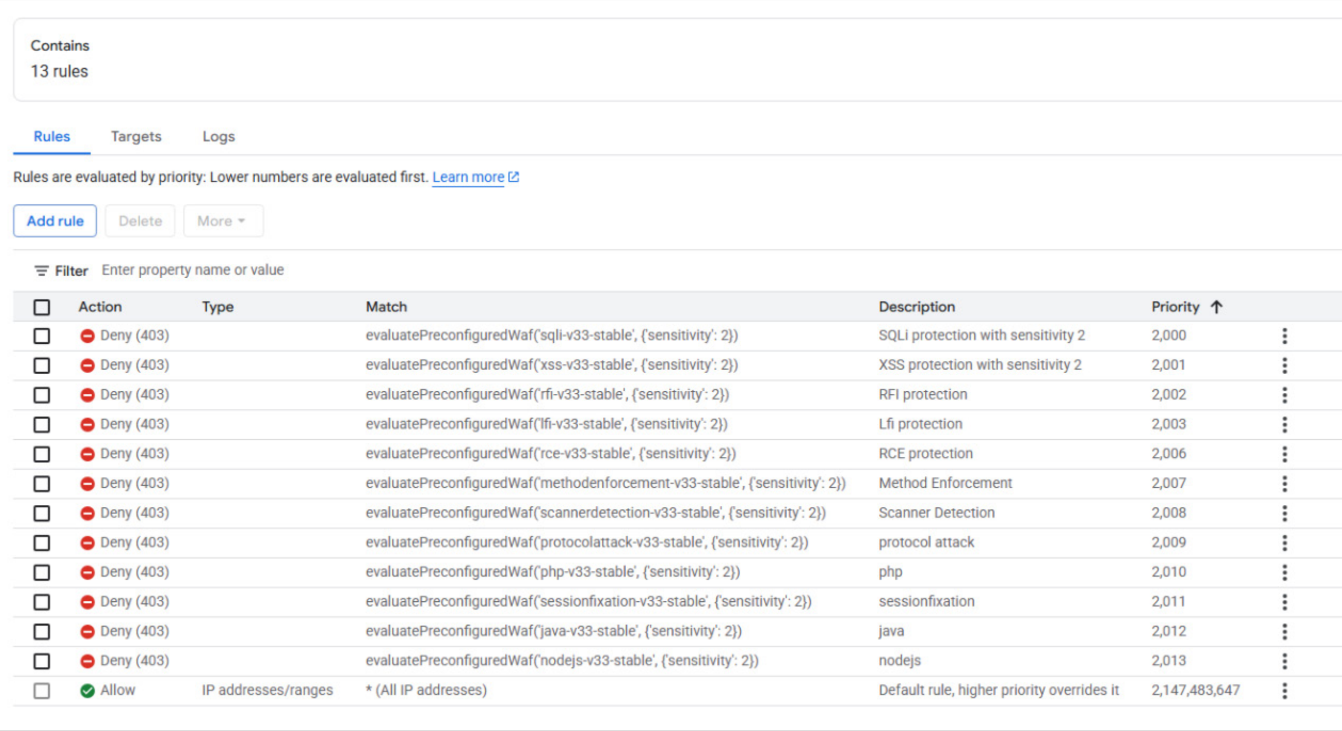
Balanced Accuracy:
67.004%

BARRACUDA WAF (NEW)

We tested the Barracuda Web Application Firewall using its default recommended configuration. This is the first year Barracuda has been included in the comparison.

During the initial phase, **we encountered a critical anomaly in the default policy**. The WAF blocked nearly all traffic - both malicious and legitimate - citing a "Header Limit Exceeded" error. Upon analyzing the logs, we observed that the WAF was counting approximately 10 additional headers beyond what was actually sent in the HTTP request. This suggests that internal processing headers added by the WAF itself were incorrectly counting toward the incoming limit.

To proceed with the test and avoid a 100% False Positive rate, **we were forced to manually disable this specific header limit constraint**. It was surprising to encounter such a fundamental blocking issue within a recommended "out-of-the-box" configuration.



Contains
13 rules

Rules are evaluated by priority: Lower numbers are evaluated first. Learn more

□	Action	Type	Match	Description	Priority ↑	
□	⛔ Deny (403)		evaluatePreconfiguredWaf('sqli-v33-stable', {'sensitivity': 2})	SQLi protection with sensitivity 2	2,000	⋮
□	⛔ Deny (403)		evaluatePreconfiguredWaf('xss-v33-stable', {'sensitivity': 2})	XSS protection with sensitivity 2	2,001	⋮
□	⛔ Deny (403)		evaluatePreconfiguredWaf('rfi-v33-stable', {'sensitivity': 2})	RFI protection	2,002	⋮
□	⛔ Deny (403)		evaluatePreconfiguredWaf('lfi-v33-stable', {'sensitivity': 2})	Lfi protection	2,003	⋮
□	⛔ Deny (403)		evaluatePreconfiguredWaf('rce-v33-stable', {'sensitivity': 2})	RCE protection	2,006	⋮
□	⛔ Deny (403)		evaluatePreconfiguredWaf('methodenforcement-v33-stable', {'sensitivity': 2})	Method Enforcement	2,007	⋮
□	⛔ Deny (403)		evaluatePreconfiguredWaf('scannerdetection-v33-stable', {'sensitivity': 2})	Scanner Detection	2,008	⋮
□	⛔ Deny (403)		evaluatePreconfiguredWaf('protocolattack-v33-stable', {'sensitivity': 2})	protocol attack	2,009	⋮
□	⛔ Deny (403)		evaluatePreconfiguredWaf('php-v33-stable', {'sensitivity': 2})	php	2,010	⋮
□	⛔ Deny (403)		evaluatePreconfiguredWaf('sessionfixation-v33-stable', {'sensitivity': 2})	sessionfixation	2,011	⋮
□	⛔ Deny (403)		evaluatePreconfiguredWaf('java-v33-stable', {'sensitivity': 2})	java	2,012	⋮
□	⛔ Deny (403)		evaluatePreconfiguredWaf('nodejs-v33-stable', {'sensitivity': 2})	nodejs	2,013	⋮
□	✓ Allow	IP addresses/ranges	* (All IP addresses)	Default rule, higher priority overrides it	2,147,483,647	⋮

The results were as follows:

Security Quality
(True Positive Rate):
34.32%

Detection Quality
(False Positive Rate):
18.197%

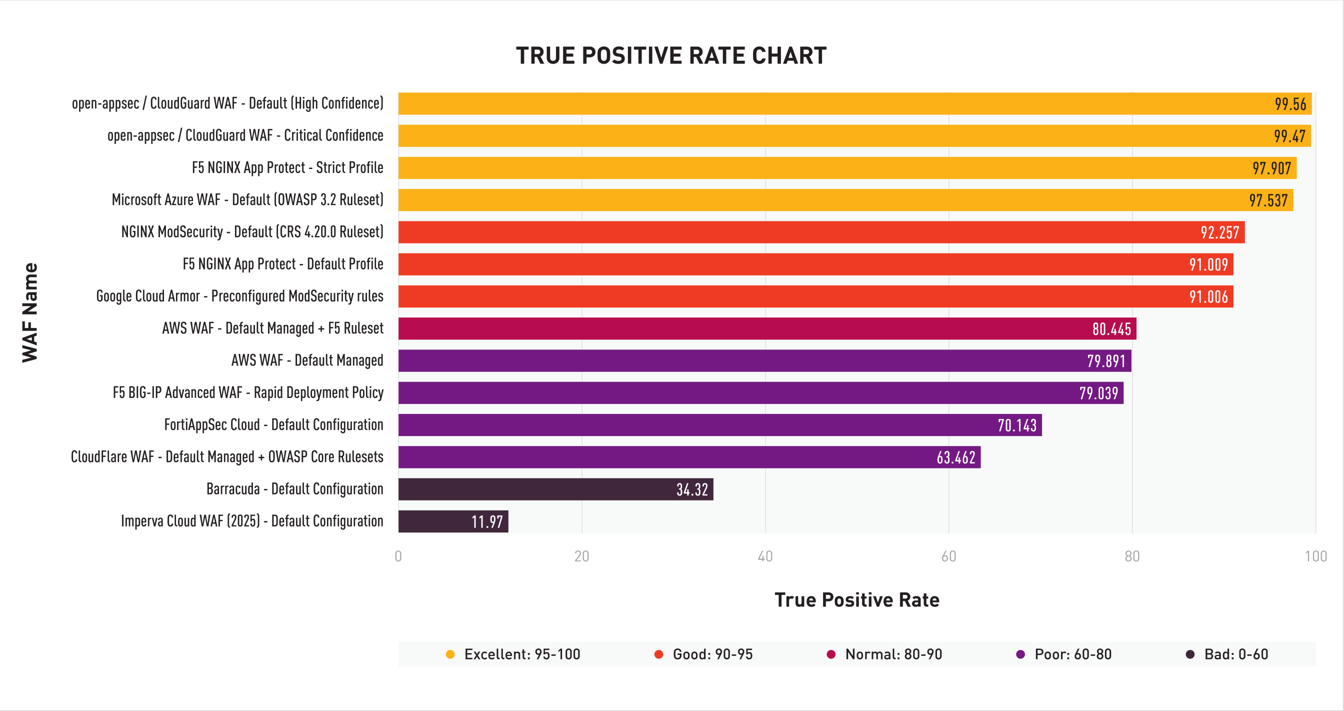
Balanced Accuracy:
58.061%

ANALYSIS

Understanding the metrics used in this comparison is key to interpreting the results accurately. Below we delve deeper into each one of these metrics.

SECURITY QUALITY (TRUE POSITIVE RATE)

The True Positive Rate gauges the WAF's ability to correctly detect and block malicious requests. A higher TPR is desirable as it suggests a more robust protection against attacks.

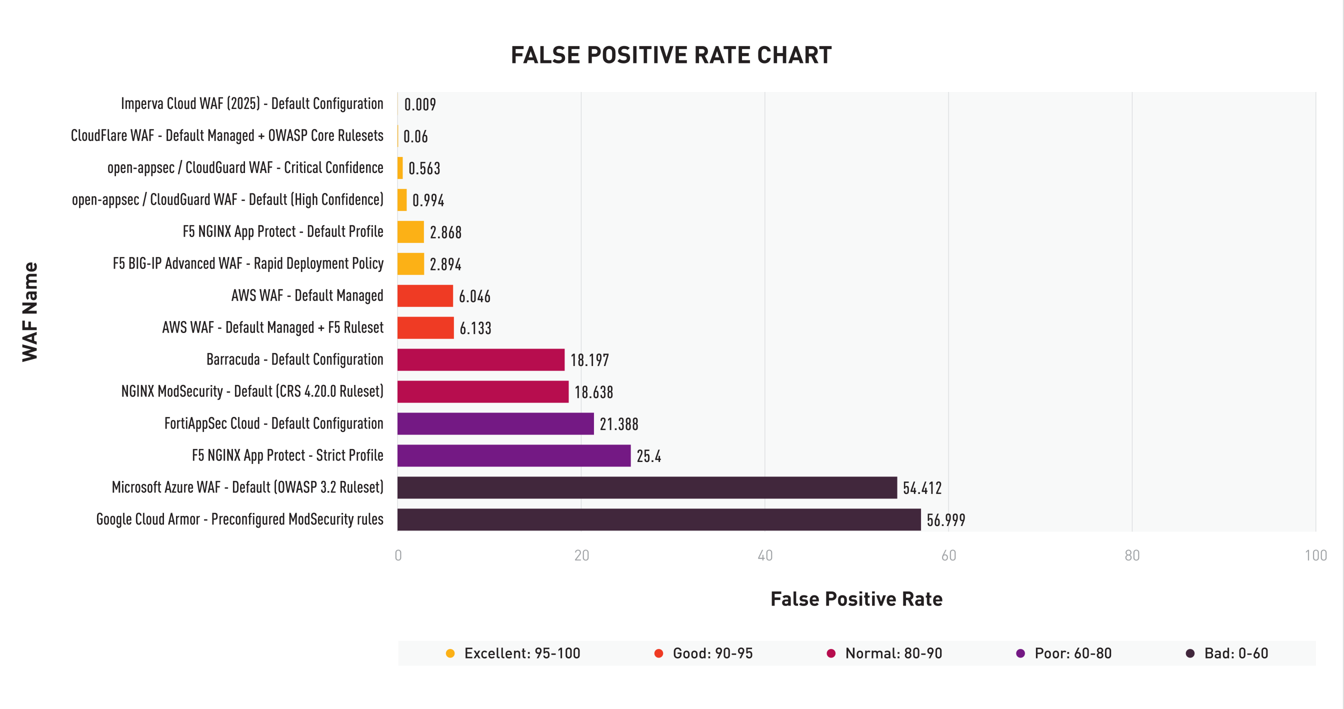


In the test, the highest True Positive Rate (TPR) was achieved by the CloudGuard WAF registering a TPR of **99.56%** with the out-of-the-box Default profile. Close after was F5 NGINX App Protect with the Strict Rule set with a TPR of **97.849%** and Microsoft Azure WAF with a TPR of **97.526%**.

The remaining WAFs showed variable results with Imperva demonstrating the lowest security quality at **11.97%** with default settings. The results of many products are discouraging as most attacks tested are known and published for a long time.

DETECTION QUALITY (FALSE POSITIVE RATE)

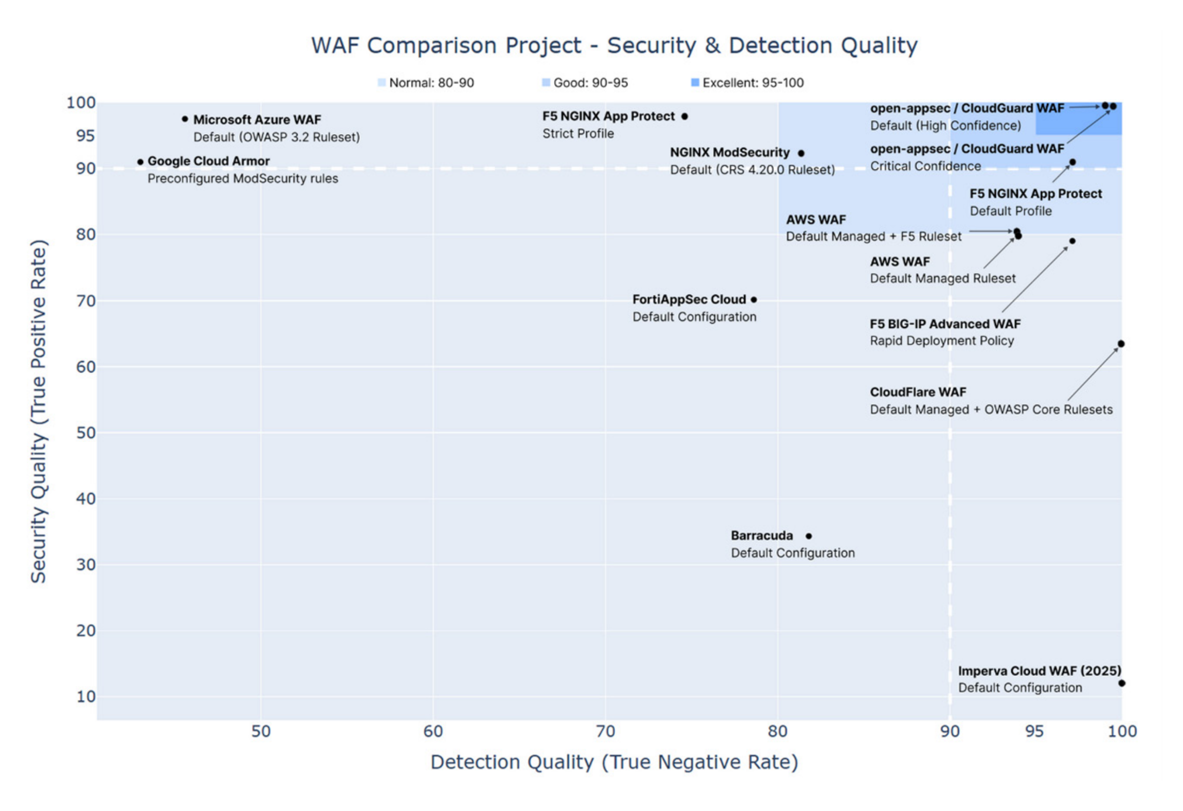
The False Positive Rate measures the WAF's ability to correctly identify and allow legitimate requests. A lower FPR means the WAF is better at recognizing correct traffic and letting it pass, which is critical to avoid unnecessary business disruptions and administration overhead.


FALSE POSITIVE RATE CHART

In the test lowest False Positive Rate (FPR) was achieved by Imperva registering an almost perfect **0.009%**. Cloudflare close behind with **0.06%**. And CloudGuard WAF - followed behind with a FPR of **0.563%** using the Critical Profile.

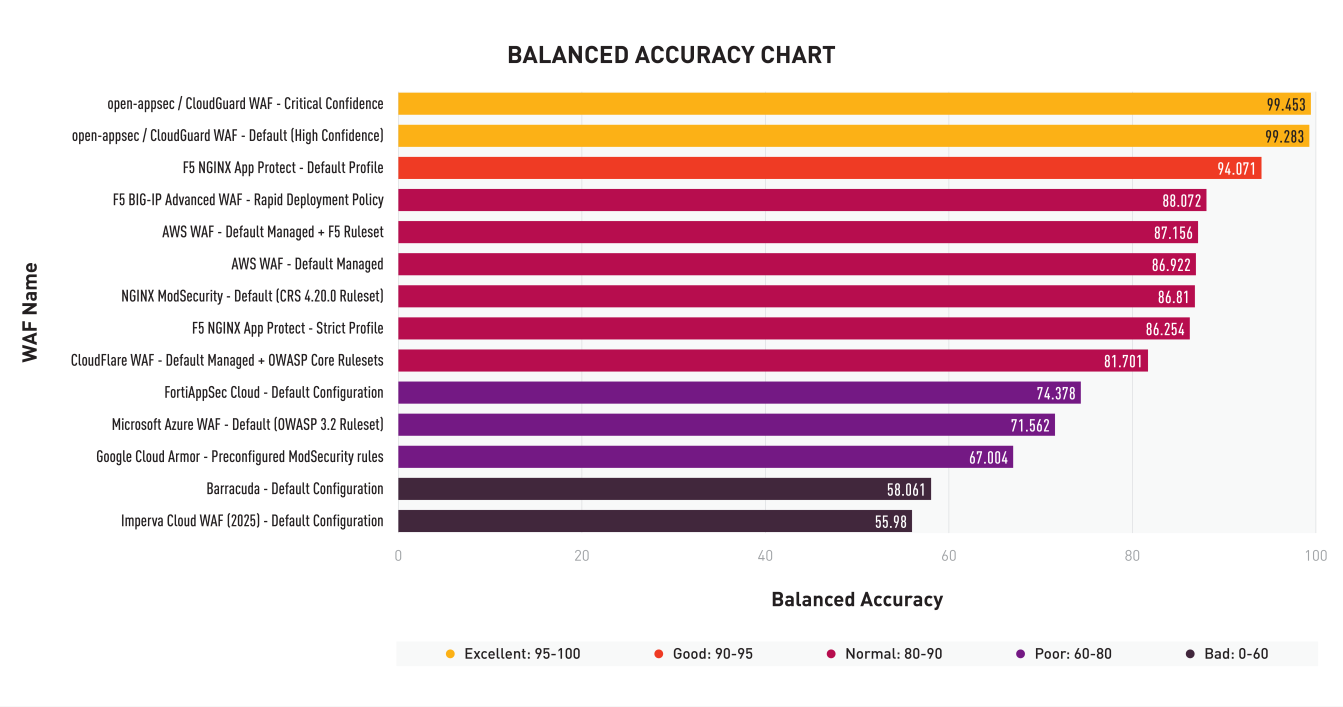
Microsoft Azure WAF presented a very high false positive rate of **54.412%** and Google Cloud Armor WAF with the Preconfigured ModSecurity Rule set had the highest False Positive Rate of **56.999%**. These products need very heavy tuning initially and ongoing, for the product to be used in real-world environments.

Below is a graphical representation of Security Quality and Detection Quality results. The visualization provides an immediate, intuitive understanding of how well each WAF solution achieves the dual goals of blocking malicious requests and allowing legitimate ones. WAF solutions that appear towards the top right of the graph have achieved a strong balance between these two objectives.


WAF Comparison Project - Security & Detection Quality
Normal: 80-90
Good: 90-95
Excellent: 95-100
Security Quality (True Positive Rate)
Detection Quality (True Negative Rate)
100
95
90
80
70
60
50
40
30
20
10
50
60
70
80
90
95
100
Microsoft Azure WAF
Default (OWASP 3.2 Ruleset)
Google Cloud Armor
Preconfigured ModSecurity rules
F5 NGINX App Protect
Strict Profile
NGINX ModSecurity
Default (CRS 4.20.0 Ruleset)
open-appsec / CloudGuard WAF
Default (High Confidence)
open-appsec / CloudGuard WAF
Critical Confidence
F5 NGINX App Protect
Default Profile
AWS WAF
Default Managed + F5 Ruleset
AWS WAF
Default Managed Ruleset
FortiAppSec Cloud
Default Configuration
F5 BIG-IP Advanced WAF
Rapid Deployment Policy
CloudFlare WAF
Default Managed + OWASP Core Rulesets
Barracuda
Default Configuration
Imperva Cloud WAF (2025)
Default Configuration

BALANCED ACCURACY

Balanced Accuracy provides a more holistic view of the WAF's performance, considering both Security Quality and Detection Quality. Higher balanced accuracy indicates an optimal WAF solution that balances attack detection and legitimate traffic allowance.



CloudGuard WAF - Critical Profile led the pack with a BA of 99.453%, closely followed by the CloudGuard WAF - Default Profile, registering a BA of **99.283%**. Imperva had the lowest BA, standing at **55.98%**.

SUMMARY

To effectively protect Web Applications and APIs, a solution must excel in two areas: Security Quality (blocking attacks) and Detection Quality (allowing legitimate traffic). This real-world comparison reveals that many leading solutions fail to strike this balance. Some pose genuine security risks by failing to detect modern threats, while others require massive amounts of manual tuning just to function in a production environment.

We believe in full transparency. By sharing our methodology, datasets, and the new Docker-based testing tool, we invite you to validate these findings yourself. We encourage every security team to use the tool to test the solutions they currently use and see the results firsthand.

Finally, CloudGuard WAF / open-appsec by Check Point proves once again that the best way to implement Web Application Security is by using Machine Learning rather than static signatures. This approach provides the best balance of security and usability and remains the only solution to preemptively block Zero day attacks such as React2Shell, Log4Shell, Spring4Shell, Text4Shell, React2Shell and Claroty WAF on Bypass.

Worldwide Headquarters
5 Shlomo Kaplan Street, Tel Aviv 6789159, Israel | Tel: +972-3-753-4599

U.S. Headquarters
100 Oracle Parkway, Suite 800, Redwood City, CA 94065 | Tel: 1-800-429-4391

www.checkpoint.com