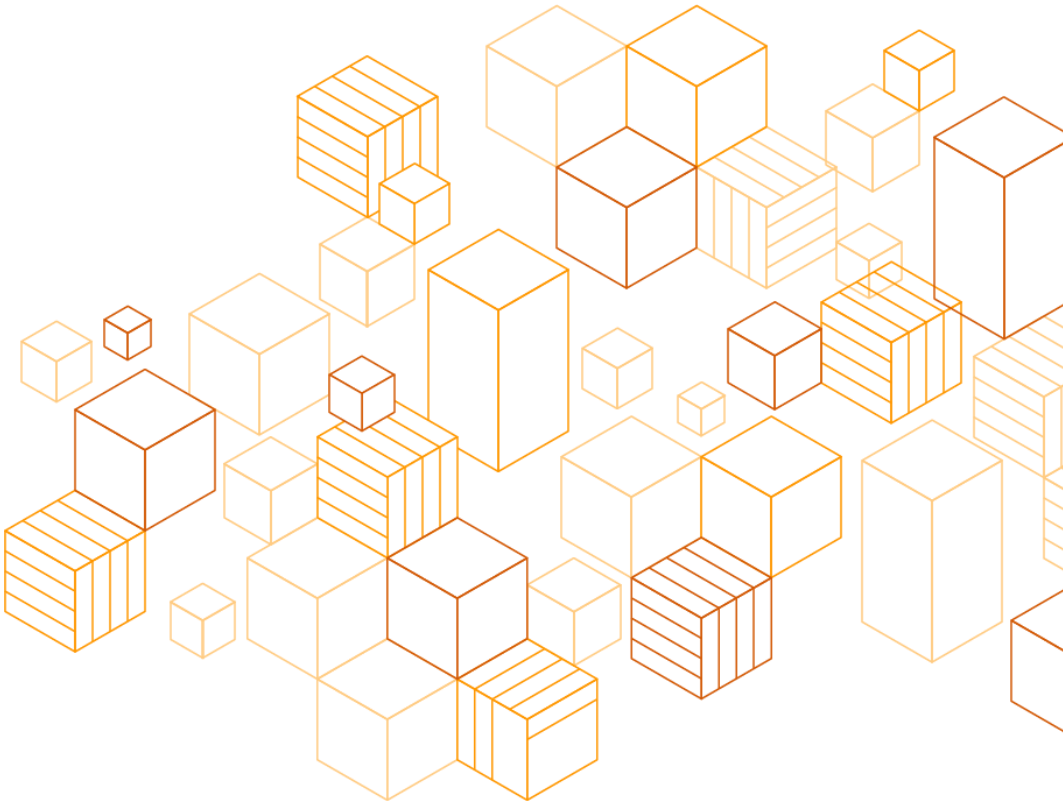
Guidelines for Implementing AWS WAF

May 2020



Notices

Customers are responsible for making their own independent assessment of the information in this document. This document: (a) is for informational purposes only, (b) represents current AWS product offerings and practices, which are subject to change without notice, and (c) does not create any commitments or assurances from AWS and its affiliates, suppliers or licensors. AWS products or services are provided “as is” without warranties, representations, or conditions of any kind, whether express or implied. The responsibilities and liabilities of AWS to its customers are controlled by AWS agreements, and this document is not part of, nor does it modify, any agreement between AWS and its customers.

Contents

About this Guide

AWS WAF is a web application firewall (WAF) that helps you protect your websites and web applications against various attack vectors at the application layer (OSI Layer 7). This whitepaper outlines current recommendations for implementing AWS WAF to protect existing and new web applications. This paper applies to anyone who is tasked with protecting web applications.

Overview

Security is a shared responsibility between AWS and the customer, with boundaries that vary depending on factors such as the AWS services used. For example, when you build your web application with AWS services like Amazon CloudFront, Amazon API Gateway, and Application Load Balancer, you are responsible of protecting your web application at layer 7 of the OSI model. AWS WAF is a tool that helps you protect web applications by filtering and monitoring HTTP/HTTPS traffic, including from the public internet. Web application firewalls (WAFs) protect applications at the application layer from common web exploits that can affect application availability, compromise security, and/or consume excessive resources. For example, you can use AWS WAF to protect against attacks such as cross-site request forgery, cross-site scripting (XSS), file inclusion, and SQL injection, among other threats in the OWASP Top 10. This layer of security can be used together with a suite of tools to create a holistic defense-in-depth architecture.

AWS WAF is a managed web application firewall (WAF) that can be used in conjunction with a wide variety of networking and security services such as Amazon VPC, AWS Shield Advanced, and more. AWS WAF can be natively enabled on Amazon CloudFront, Amazon API Gateway, and Application Load Balancer and is deployed alongside these services. The previously mentioned AWS services terminate the TCP/TLS connection, process incoming HTTP requests, and then forward the request to AWS WAF for inspection and filtering. Unlike traditional appliance-based WAFs, there is no need to deploy and manage infrastructure, or plan for capacity. AWS WAF provides flexible options for implementing protections via managed rules, partner provided rules, and custom rules that you can write yourself.

It's important to understand that with AWS WAF, you control ingress traffic to your application. To control egress traffic, see Security best practices for your VPC.

In this document, we provide you with recommendations for protecting existing and new applications with AWS WAF. We outline the following steps and options to consider when deploying AWS WAF:

1. Understanding threats and mitigations
2. Requirements for AWS WAF
3. Implementing AWS WAF
4. Deploying AWS WAF to production

5. Cost considerations

> **Note:** AWS WAF provides two versions of the Service: WAFv2 and WAF Classic. We recommend using AWS WAFv2 to stay up-to-date with the latest features. AWS WAF Classic no longer receives updates. AWS WAFv2 includes features that are not available in WAF classic, including a separate API and Console. In this paper, we focus on implementation with AWS WAFv2.

Understanding Threats and Mitigations

Before deciding how to deploy AWS WAF, you need to understand what type of threats your web applications may be facing and the protection options available with AWS WAF. Web applications face different kinds of threats that AWS WAF can help you mitigate.

- **Distributed denial of service (DDoS)** attacks try to exhaust your application resources so that they are not available to your customers. At layer 7, DDoS attacks are typically well-formed HTTP requests that attempt to exhaust your application servers and resources.
- **Web application** attacks try to exploit a weakness in your application code or its underlying software to steal web content, gain control over web servers, or alter databases; these can involve HTTP requests with deliberately malformed arguments.
- **Bots** generate a large portion of the internet's website traffic. Some "good" bots associated with search engines crawl websites for indexing. However, "bad" bots may scan applications looking for vulnerabilities, looking to scrape content, poison backend systems or disrupt analytics.

AWS WAF helps you to improve your security posture against these types of threats (*Figure 1*).

Figure 1 – Types of threats at Layer 7

DDoS Attacks at Layer 7

For HTTP floods, you can use AWS WAF rate limiting rules to block clients from specific IP addresses that are sending abusive amount of requests to your application. AWS WAF also provides the ability to block known malicious IP addresses using the Amazon IP reputation list from the AWS Managed Rules or by subscribing to AWS partner IP reputation lists from the AWS Marketplace. For more advanced mitigations, see the AWS WAF Security Automations Solution to activate:

- **The scanners and probes protections** which parse application access logs searching for suspicious behavior, such as an abnormal amount of errors generated by an origin to block bad actors.
- **The Reputation list protection** to block requests from IP addresses on third-party reputation lists like DROP and EDROP from Spamhaus, the Tor exit node list and the Proofpoint Emerging Threats IP list.

In addition to using AWS WAF, we recommend reviewing AWS Best Practices for DDOS Resiliency whitepaper when architecting for DDoS resiliency at OSI layers 3, 4, and 7.

Web Application Attacks

AWS WAF provides the following options for protecting against web application exploits.

AWS Managed Rules

You can select and add some of AWS managed rule groups to protect your application from various threats. Managed rule groups include:

- A baseline rule group that covers some of the common threats and security risks described in OWASP Top 10 publication;
- Use case specific rule groups for incremental protection based on your application characteristics, such as the application OS or database;
- An IP reputation list derived from the Amazon threat intelligence team to block known malicious IPs.

Custom Rules

In addition to AWS Managed Rules, you can also write custom rules specific to your application to block undesired patterns in parts of the HTTP request, such as headers, method, query string, URI, body, and/or IP address. You can use these rules together with the AWS Managed Rules groups to provide customized protections. You can construct custom rules using the rule builder in the AWS Management Console. Or, you write custom rules in JSON and configure the rules using the AWS Command Line Interface (AWS CLI) or using automation tools such as AWS CloudFormation. For example, you can use custom rules to block requests that do not respect your expected API URL scheme. For the full list of logical statements that you can express using custom rules, see Rule statements list.

AWS Marketplace Rules

On the AWS Marketplace, you can also find rules created by security vendors that have built their own rule sets on AWS WAF. These rules are available based on subscription and can be used together with AWS Managed Rules and your own custom rules.

Advanced Automated Mitigations

For behavioral protections, you can use AWS WAFs APIs to react to threats detected from logs, honeypot URLs, and more to automatically update rules and block malicious IP addresses. With changes propagating within a minute (globally for Amazon

CloudFront and regionally for Application Load Balancer and Amazon API Gateway), you can react quickly to threats by updating rules in existing WAF web access control lists (ACLs). AWS provides the AWS WAF Security Automations Solution as a reference for various protection techniques.

Bad Bots

To stop traffic generated by bad bots, we recommend using the IP reputation lists within AWS Managed Rules to cover some of the scanner type bots. In addition, you can use the AWS WAF Security Automations Solution to defend against bots by implementing honeypots and behavioral detections with WAF logs. For more sophisticated detections of the most difficult bots involved in application-level attacks (such as bots attempting credential-stuffing), we recommend adding a bot management solution to your architecture. You can find third-party solutions on the AWS Marketplace that provide advanced bot mitigation capabilities. Some of these solutions also provide the ability to integrate with Amazon CloudFront using Lambda@Edge for inline protection.

Requirements

As a first step towards implementing AWS WAF, we recommend that you gather and define the requirements which will make this implementation successful for your business. In this section, we go through some of the common WAF requirements.

Protections

Once you have identified which threats are applicable for your application, define your baseline criteria for success. These criteria can include passing penetration tests performed by third-party or internal security teams, meeting specific compliance requirements, or simply having coverage for common web vulnerabilities (e.g., OWASP Top 10). For example, the sensitivity of the content that your application serves may dictate whether you choose to implement a positive security vs. negative security model (allow vs. deny APIs) when creating your WAF web access control list (web ACL). If your application does not use an SQL database, you can save WAF Capacity Units (WCUs) by not adding SQL injection detection rules. We recommend that you add WAF rules that are specific to your application's requirements, as adding unnecessary rules can lead to an increase in false positives. False positives are legitimate requests that were considered by WAF as attacks and blocked as a consequence.

For existing applications, you may already have visibility into the application's usage patterns and be looking to block malicious requests identified from previous incidents and observations. Therefore, you may be looking for protections against a specific attack. If you are already using a WAF implementation, you may have a baseline of the average number of requests blocked by the existing WAF rules. In some cases, you may have visibility into the existing rules implemented and you can implement similar rules in AWS WAF.

Managed vs Custom Rules

Depending on your organization's resources and security culture, you must decide how to implement AWS WAF. You can deploy out-of-the-box AWS Managed Rules sets, create your own custom rules, or use a combination of both. For most applications, we recommend starting with the baseline rule groups and the Amazon IP reputation list from the AWS Managed Rules, then selecting application specific rule groups that match the application's profile.

For some workloads, advanced protections may be required. In such cases, you may add additional custom rules on top of those already provided. Managing and implementing your own rules requires that your security and/or applications teams develop skills in creating and managing WAF rules. To help with these workloads, AWS Professional Services or AWS WAF Partners can help you create these rules, perform periodic reviews, and train your teams to develop this expertise.

Governance

You may also have governance requirements to define how to manage and monitor WAF implementations across your organization. In some organizations, WAF configurations are managed centrally by a security team. In this case, the security team must audit and ensure that WAF is configured correctly across resources managed by application teams. In other organizations, WAF configuration and deployment is managed by the application teams so that the WAF rules deployed can be specific to the protected application.

To simplify centralized management of AWS WAF, AWS Firewall Manager allows you to define security policies that automatically deploy WAF across accounts within your AWS Organization. AWS Firewall Manager provides you with visibility to ensure that resources have the appropriate WAF web ACL associated and are within compliance of the WAF policies. To illustrate the possibilities, see the following governance examples:

Example 1 – AWS Managed Firewall Implementation

In this example, you have autonomous application teams that own WAF configurations with the supervision of a central security team.

- The central security team provides and documents generic guidance in the form of best practices for the application teams to follow.
- The central security team uses AWS Firewall Manager with a WAF policy to deploy a central web ACL (based on AWS managed baseline rule groups) to each team's account without automatic remediation. This policy is configured to deploy a copy of the web ACL but not automatically associate it to application resources (e.g., CloudFront, Application Load Balancer, Amazon API Gateway). Although this approach does not force the protection on the application teams, it provides the central security team with visibility of which applications have WAF attached to their endpoints.
- Application teams can choose to apply the central web ACL as it is, or modify it before application. Their choice is mostly driven by their security requirements and governance.

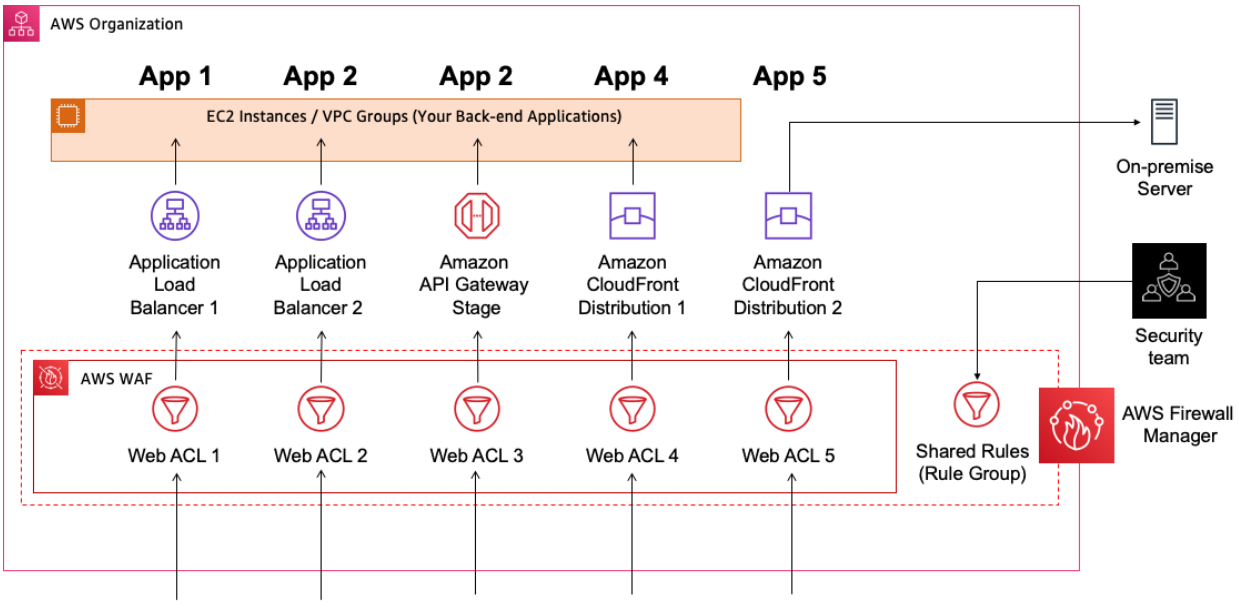


Figure 2 - Example of an AWS Firewall Manager implementation

Example 2 – AWS Firewall Manager Implementation with Two WAF Policies

In this example, you have a central security team that manages WAF deployments and rules for applications across your organization.

- The central security team creates two AWS Firewall Manager WAF policies with automatic remediation:
  - One policy uses managed rules for WordPress (e.g., as a sample application) for all resources tagged as a WordPress application.
  - One policy uses Amazon IP reputation list and rate limiting rules for all other HTTP(S) applications.
- Applications teams tag resources associated with WordPress applications accordingly.
- In each AWS Account within the organization, Firewall Manager creates two web ACLs, one for each policy. Firewall manager automatically associates the web ACLs to the appropriate resources as configured by the policy. When this occurs, existing WAF web ACLs associated to those resources are overridden.

  Note: At the time of this publication, it is not possible to control which policy is enforced if more than one can apply to a single resource. For that reason, refrain from overlapping policies to have consistency across your organization.
- The security team can monitor WAF compliance through the AWS Firewall Manager in the AWS Management Console. Firewall Manager allows you to identify if resources have the correct WAF web ACL associated as configured by the Firewall Manager policy. You can also integrate AWS Security Hub with AWS Firewall Manager to detect resources that are not properly protected by WAF rules.

Logging

WAF logging is a common requirement for security teams to meet their compliance and auditing needs. AWS WAF provides near-real-time logs through Amazon Kinesis Data Firehose. For each inspected request by AWS WAF, a corresponding log entry is written that contains request information such as timestamp, header details, and the action for the rule that matched. Currently, AWS WAF does not log the request body. You can use logs for debugging and additional forensics by integrating with your Security Information and Event Management (SIEM) or other log analysis tools. By default, logging is not enabled when you create a web ACL. To automate log enabling, you can use AWS Config to configure logging whenever a new WAF web ACL is created.

Figure 3 - Automation of logging activation

Implementation

Select a Starting Point

Once you've identified your requirements, you must decide which application to start with. If you are new to AWS WAF, we recommend starting with a non-critical application where possible. This approach allows you to familiarize yourself with the new platform and tune your configuration while limiting the risks of misconfiguration to your business. We also recommend starting with an application for which you have a good understanding of the application traffic patterns. This approach allows you to quickly identify the impact of deploying WAF on your application traffic so that you can tune accordingly.

AWS WAF Integration Design

Depending on your application's requirements, you must decide where to deploy AWS WAF. As mentioned previously, you can configure AWS WAF on Amazon CloudFront, Amazon API Gateway, and Application Load Balancer. For all public facing web applications, we recommend deploying AWS WAF with Amazon CloudFront for the best security posture, unless you have constraints that require otherwise. CloudFront can be used for both dynamic and static content. By default, CloudFront blocks non-HTTP(S) traffic, malformed HTTP requests, and provides in-line DDoS protection for attacks at network layers 3 and 4 with sub-second time-to-mitigation. CloudFront employs advanced DDoS protections, such as stateless SYN Flood mitigation and automated traffic engineering systems that can disperse or isolate the impact of large volumetric attacks on the Amazon CloudFront Global Edge Network (most effectively when

deployed in conjunction with Amazon Route 53). If your application is hosted outside of AWS, CloudFront provides a seamless way to use the AWS global network to stop threats before they reach your data centers.

For some applications with an additional level of requirements, you may choose to implement a layered WAF model by using AWS WAF in conjunction with another WAF offering at the origin providing or load-balancing your service:

- For example, you may want to inspect responses returned by the origin. In this case, you can use AWS WAF to inspect incoming requests to CloudFront at the edge, and an appliance-based WAF to inspect incoming requests and outgoing responses from your origin. Some appliance-based WAFs have the ability, when they detect attack traffic, to synthesis rules for the AWS WAF and push them into your AWS WAF rule set, if configured appropriately and given an IAM role with appropriate Allowed Actions.
- Another example is protecting multiple applications on a single domain served by CloudFront. You can use AWS WAF on CloudFront for common IP and IP geolocation-based blocking at the edge, and deploy additional AWS WAF capability on each of your Applications Load Balancers for application-specific rules.

Validation in Staging Environment

Once you pick an application to start with, we recommend setting up a staging environment for your application. This approach allows you to experiment with AWS WAF without negatively impacting production traffic. You can approach staging in two ways according to practices in your organization :

- Replicate your entire application stack to a staging environment including AWS WAF.
- Create a new endpoint for your production environment. Your staging environment is based on this new endpoint with AWS WAF deployed. For example, you can create a new CloudFront distribution with WAF web ACL attached and set the origin to your existing application’s load balancer. **Note:** If you are already using CloudFront, you can still create a new CloudFront distribution but you can’t reuse the same domain attached to the existing distribution.

With your WAF staging environment set up, we recommend that you restrict access to the entirety of this environment to the authorized developer team. There are multiple options for achieving this:

- Use the AWS WAF to block requests that do not come from your organization's public IP address range. However, this approach doesn't offer authentication and can be difficult to manage if developers works from home (unless they are required to VPN and proxy into the corporate network environment).
- Implement an authorization mechanism in your application, and forward the authorization header in CloudFront's cache behavior configuration.
- If you do not want to make a change to your application, you can offload authorization to your endpoint. For example, if you are using CloudFront, you can add a Lambda@Edge function that provides access control, or use CloudFront native signed cookies for the same purpose.

Deployment in Staging

When deploying a web ACL, we recommend starting with the following setup:

1. **Add rules based on your defined requirement.**

   If you are new to AWS WAF or do not have specific requirements you can start with coverage for common web vulnerabilities offered by AWS Managed Rules. It is important to think about the order of rules in your web ACL, since AWS WAF processes rules in order of priority, and stops the web ACL evaluation when there is a match to execute the action of the matching rule.

   Note: Deploying rules in block mode allows you to see how rules impact test traffic in your staging environment. However, consider executing your production deployment procedures in staging so that your operators understand how WAF behaves before moving to production. In many cases, it is common to start new rules in count mode before switching to block mode when deploying to production. This way, you avoid compromising the availability of your application because of a misconfigured WAF rule that could block legitimate traffic.

2. **Enable rate-based rules to protect yourself against DDoS types of attack (e.g., HTTP flood).**

The rate-based rule keeps track of the number of requests seen per IP address based on a sliding time window of 5 minutes. The sliding window is updated every 30 seconds and once the rate limit is reached, the rule immediately takes action pertinent to the IP address. It does not wait until the 5 minutes has passed before taking the action. The rate-based rule keeps blocking requests from the offending IP address until the address lowers the rate of requests being sent from it.

Note: You are able to set the limit as low as 100 requests per 5 minutes; however, we recommend that you start with 2,000 requests and gradually reduce this number as needed.

Once you are satisfied with the rules you created within the staging environment, you can duplicate them to your production environment or another account by simply copying the rules from the web ACL. In the web ACL overview page, there is an option to download the entire web ACL configuration, including all rules, in a JSON file. Once downloaded, you can either manually copy the rules and recreate the web ACL or convert the JSON to YAML and use it in a CloudFormation template to deploy the web ACL.

Monitoring and Visibility

Having good visibility of what is being blocked by your web ACL is important for operating your WAF implementation. This visibility is useful for threat intelligence, hardening rules, troubleshooting false positives, and responding to an incident. There are multiple monitoring options available with AWS WAF.

Monitoring using Amazon CloudWatch

You can set up a dashboard for AWS WAF to display information about the activity of rules in your web ACL. For each rule, CloudWatch emits near-real-time metrics like *AllowedRequests, BlockedRequests and PassedRequests* which are recorded for a period of two weeks. The following image is an example of what you can easily set up with CloudWatch to display real time and historical information about how your web ACL is protecting your application. In addition, you can set up alarms on CloudWatch metrics to receive notifications when a certain WAF rule is abnormally triggered based on predefined thresholds.

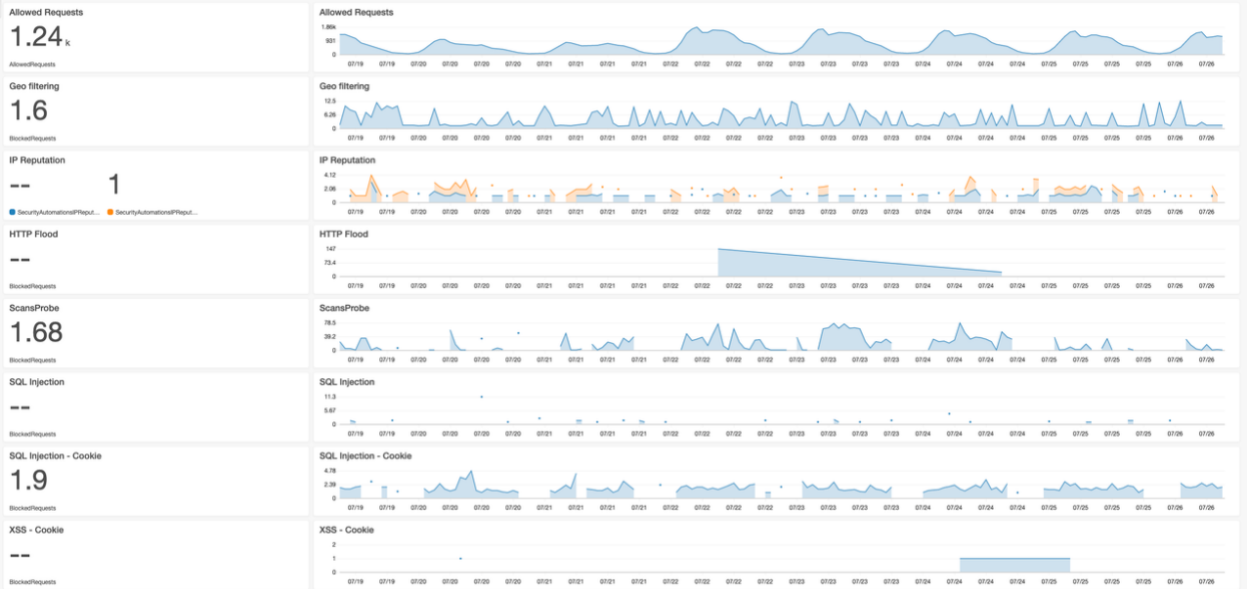


Figure 4 - Security dashboard using CloudWatch

However, CloudWatch doesn't provide you with information about the processed requests themselves. If you need to get more details more about inspected requests, you have two options:

- **View a sample of the WAF log in the WAF console.** For each sampled request, you can view detailed data about the request, such as the originating IP address and the headers included in the request. With this approach, you can quickly debug false positives in a staging environment. The sampled request works by randomly fetching 5,000 requests that the web ACL has processed over the time period that you chose (up to the previous three hours).
- **Enable and process AWS WAF logs for full and detailed information**. This approach more suitable for deeper troubleshooting in a production environment. For each request, AWS WAF logs provide raw HTTP/S headers along with information on which rules were triggered. In addition, AWS WAF logs provide the exact patterns that triggered SQLi and XSS rules in the '*terminatingRuleMatchDetails*' field. AWS WAF logs are ingested using Amazon Kinesis Data Firehose and can be delivered in JSON format to multiple destinations, including Amazon S3. We recommend this approach for all production workloads for the best visibility and troubleshooting.

It's common to build custom dashboards based on AWS WAF logs, to provide a near-real-time global view of your application security, and deep dive into request details when needed. With AWS WAF logs, you can build your own dashboard using AWS

services, or use third-party services. If you are already using third-party monitoring services like Splunk, Datadog, or Sumo Logic, you can export WAF logs to these services. For example, Sumo Logic has a template to create a dashboard for examining WAF logs.

Testing and Tuning

After your initial implementation of WAF, there is normally a phase of tuning to mitigate potential false positives and false negatives. False negatives are those attacks that were not caught by your WAF and require you to harden your rules. False positives represent legitimate requests that were considered by WAF wrongly as attacks and blocked as a consequence.

False Negatives

To identify **false negatives** based on your security requirements, you can use penetration testing providers, automated vulnerabilities scanners from third-parties, or open source web application security scanners. Although automated vulnerability scanners are great tools to test your architecture quickly against known vulnerabilities and pre-defined attack vectors, they only cover known cases and may falsely flag an issue when there isn't one. Therefore, keep in mind that testing your WAF rules using a vulnerability scanner is not a guarantee that your application is fully protected. However, you can use a vulnerability scanner for sanity-checking. In addition, make sure to regularly update your scanner and set a regular cadence for running it, so that you are aware of new attacks and can update the WAF rules whenever necessary.

False Positives

False positives are generally identified by quality assurance (QA) teams by testing the application after changes are introduced to the application's code or WAF configuration. In many cases, this testing requires a deep understanding of the application to tell the difference between suspicious request signatures and a malicious attack. In some cases, false positives may only appear in production owing to a shortcoming in QA test coverage. To help identify these cases, consider the following:

- Set up alarming on selected WAF rules in CloudWatch to be notified when a rule is triggered above predefined thresholds.

- Update your application experience to allow real users to report unexpected unauthorized access to your application. For example, when WAF is deployed with CloudFront, you can use custom error pages to catch 403 error codes and serve a friendly response. This page can prompt the user to provide information on the issue they've encountered.
- Enable WAF logging. Have security and application teams review and baseline blocked traffic on a regular cadence to identify threat patterns and anomalies in blocked traffic.

When you detect a false positive, you must understand what has triggered it. A quick way to identify the trigger by using the AWS Management Console to check sample logs that corresponds to the rule being triggered. A more robust way to identify the false positive is to check the logs generated by AWS WAF and filter on the blocked request by URL, IP, and timestamp. If you are using AWS WAF with CloudFront, when a request is blocked, CloudFront returns a `Request ID` within the response headers and body. You can use this `Request ID` to find the corresponding record in WAF logs.

403 ERROR

The request could not be satisfied.

Request blocked. We can't connect to the server for this app or website at this time. There might be too much traffic or a configuration error. Try again later, or contact the app or website owner.
If you provide content to customers through CloudFront, you can find steps to troubleshoot and help prevent this error by reviewing the CloudFront documentation.

```
Generated by cloudfront (CloudFront)
Request ID: L5NAf5OXnd5zLf8Jdl75Ke_4AUjIfYpF5VQS5EghVMwr6qjzJs5VMA==
```

Figure 5 - Error page displayed by CloudFront

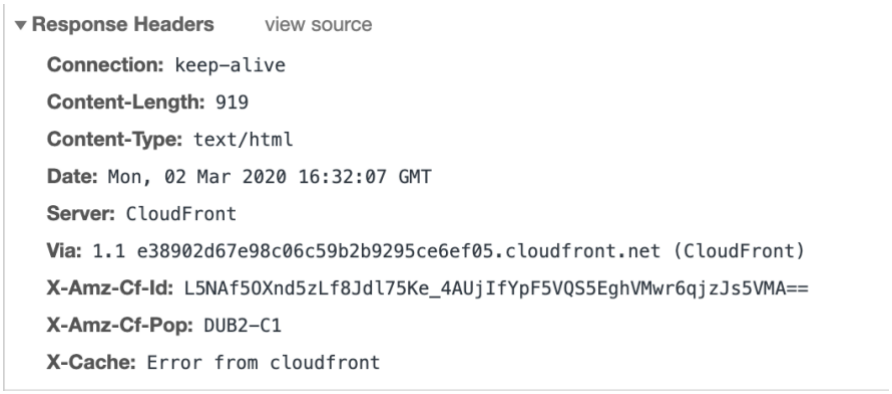


Figure 6 - Error page response headers

With the log record, you have the information about which rule was triggered—for example, a specific header doesn't respect your size constraint rule. In some cases, the information is not sufficient to identify the issue. For example, an SQLi rule triggers on a cookie in the request, but it's not obvious to you which part of the cookie is causing this trigger. For SQLi and XSS rules, the *terminatingRuleMatchDetails* field in the log record provides more details on the match.

Resolving False Positives

Once you understand the issue, you can resolve it. The best approach is to change the application code that is generating requests that look like attacks, but that may take some time and effort. As a quick fix, you can create an exception to the rule in WAF, but note that creating exceptions in WAF exposes your application to potential attacks.

Example 1: Override rules using AWS CLI

Suppose that you have a legitimate URL pattern 'xxxx' that is blocked by an XSS rule on the path of the URI. You can override this block by adding an additional rule to include an exception on the 'xxxx' URL pattern:

```
BLOCK [XSS condition] AND NOT[String Match Condition on Path]
```

The following WAF rule statement illustrates this example. This solution works for SQLi and XSS rules as well as for any custom rules.

```
{
  "Name": "XSSprotection",
  "Priority": 0,
  "Action": {
    "Block": {}
  },
  "VisibilityConfig": {
    "SampledRequestsEnabled": true,
    "CloudWatchMetricsEnabled": true,
    "MetricName": "XSSprotection"
  },
  "Statement": {
    "AndStatement": {
      "Statements": [
        {
          "XssMatchStatement": {
            "FieldToMatch": {
              "UriPath": {}
            },
            "TextTransformations": [
              {
                "Type": "URL_DECODE",
                "Priority": 0
              }
            ]
          }
        },
        {
          "NotStatement": {
            "Statement": {
              "ByteMatchStatement": {
                "FieldToMatch": {
                  "UriPath": {}
                },
                "PositionalConstraint": "CONTAINS",
                "SearchString": "xxxx",
                "TextTransformations": [
                  {
                    "Type": "LOWERCASE",
                    "Priority": 0
                  }
                ]
              }
            }
          }
        }
      ]
    }
  }
}
```

Example 2: Override rules using Managed Rules

Suppose that you have a legitimate URL pattern ‘xxxx’ that is blocked by a managed rule such as a Core rule set by AWS. You can override the action of the blocking sub rule by setting it to count mode to mitigate false positives temporarily. However, you open your environment to attacks previously stopped by this sub rule.

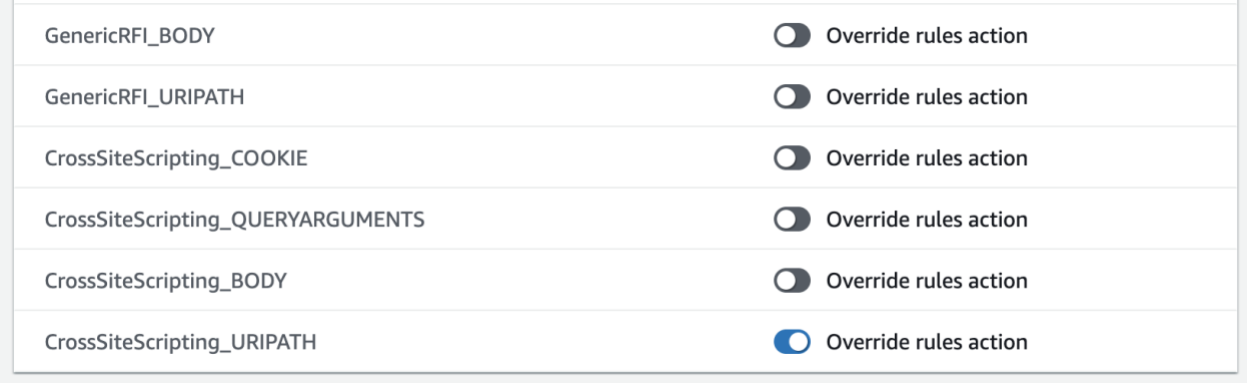


Figure 7 – Overriding the action of a sub rule in Managed Rules

Another approach is creating an allowed list rule before the managed rule, which allows traffic when it matches the false positive ‘xxx’ pattern and stops evaluating the rest of rules. With this approach, you risk exposing your application to the attacks normally stopped by the rules coming after your false positive allowed list rule when it matches. One way to reduce this risk is by setting the allowed list rule to the lowest priority in the web ACL rules order.

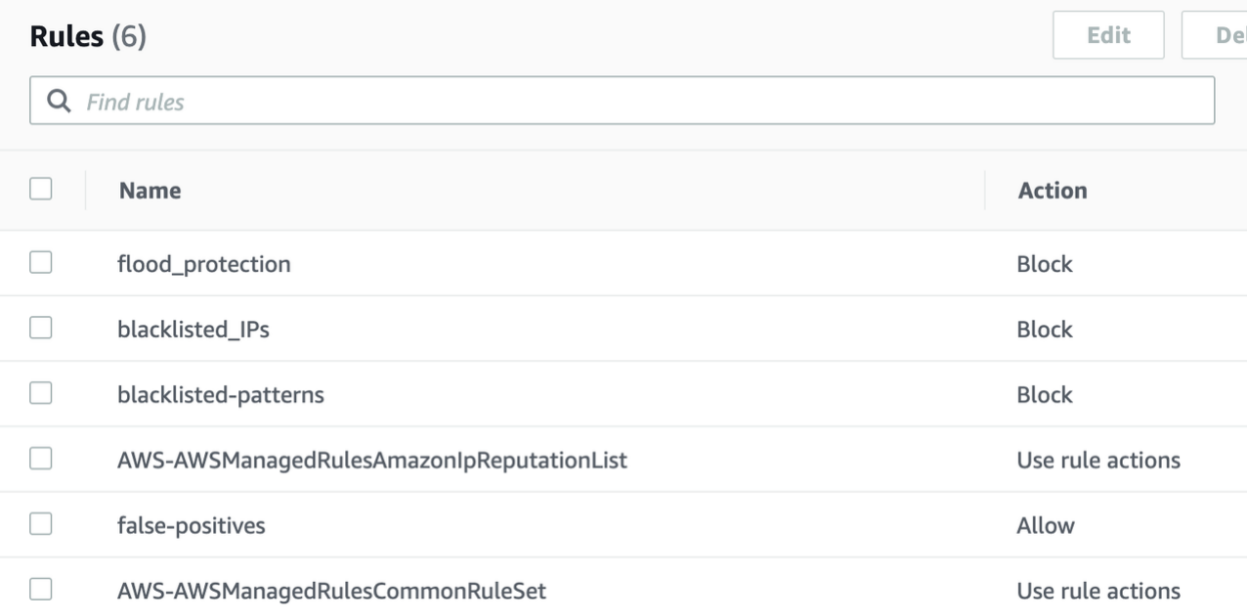


Figure 8 - Adding a rule to handle exceptions

After you deal with false positives, either by adding exceptions to WAF rules or by changing your application code, you can validate your remediation by replaying the requests that caused the false positive using tools like cURL or PostMan. If the false positive is mitigated properly, the request should not be blocked.

Deployment to Production

Operational Readiness

Once you have tested and validated your WAF implementation in your staging environment, decide when you should deploy it to production. Select a date and time when you expect to have lowest user traffic. Before deployment, make sure your application and security teams review operational readiness, discuss how to rollback changes, and review dashboards to ensure all metrics and alarms required are configured. You can create runbooks so that team members understand how to execute rollback and other mitigating operations. In the event of a security threat, your teams should know how to update your configuration, deploy in a different account, troubleshoot problems, and respond. These runbooks outline steps to take when an event occurs. For example, your runbook may include the following instructions:

1. How to deploy the setup.
2. How to mitigate false positives.

3. How to troubleshoot issues.
4. Details on which metrics to review for understanding applications health, such as: CloudFront response types (HTTP 200, 4xx, 5xx), application and database servers CPU/memory, WAF metrics.
5. Common queries to execute on logs data to filter request data for deep inspection.
6. Steps for engaging security teams who configure and deploy AWS WAF.
7. Steps for engaging AWS.

If you are subscribed to AWS Shield Advanced and are under a DDoS attack, you can engage the AWS DDoS response team (DRT). The DRT helps you analyze the suspicious activity and assists you in mitigating the issue. This mitigation often involves creating or updating AWS WAF rules and web ACLs in your account. The DRT can inspect your AWS WAF configuration and create or update AWS WAF rules and web ACLs for you. We recommend that as part of setting up AWS Shield Advanced, you proactively provide the DRT with the needed authorization to complete these tasks. Providing authorization ahead of time helps prevent mitigation delays in the event of an actual attack.

Also ahead of deployment, be sure to create a common messaging channel with all stakeholders, including network engineers, security engineers, application teams and any AWS account teams supporting you. Make sure you communicate updates through this channel.

Deployment

Once you are ready enable AWS WAF for your production endpoints, decide if you want to first trial the rules in count mode to catch any potential false positives that were not found during staging. If you are deploying a WAF rule for the first time, this technique can help avoid blocking legitimate traffic. However, understand that placing rules in count mode makes your application vulnerable to attacks mitigated by the rule until you place the rule in block mode. Trialing rules in count mode may be undesirable if you have a high degree of confidence that the rule will not generate false positives. After you review metrics and dashboards and you are comfortable with how your rules match, you can change the rules from count to block mode.

If you are already using another WAF offering, for example a CDN-based WAF, we recommend that you shift traffic progressively from the existing offering to AWS WAF.

For example, you can use the Amazon Route 53 weighted routing policy to shift traffic progressively to a newly created CloudFront endpoint with WAF activated on it.

We do not recommend that you stack AWS WAF with other WAF offerings for evaluation as this can result in conflicts in how rules are matched. For example, one of the capabilities of a WAF is the ability to block IP addresses. Many WAF solutions rely on the IP address from the TCP connection to determine the client IP address. When you layer WAF solutions, the upstream WAF solution loses visibility into the client IP address which prevents IP rules from working effectively. In some cases, you may accidentally block legitimate traffic from one WAF to the other.

If you choose to use AWS Firewall Manager to deploy AWS WAF, we recommend starting with a policy that deploys to a single endpoint first. After you test and validate your WAF policy with one endpoint, you can update your policy to match all endpoints you want to protect. You can tag similar endpoints (CloudFront distributions, Application Load Balancer, and API Gateway) with a common tag and create a Firewall Manager policy to apply the rule group to all the endpoints with that tag. It is also important to tag production endpoints with the correct tag when you create a new production endpoint, so the correct Rule group is attached to the endpoint upon deployment.

Post Deployment

After deploying to production, it is critical to regularly review and monitor your application. Application and security teams should review dashboards to develop a baseline understanding of application traffic patterns. With AWS WAF logs, you can use Amazon ElasticSearch Service, Amazon Athena, or third-party SIEM tools to analyze application traffic patterns in detail and identify past trends in application traffic and changes in behavior. With this information you can dive deeper into anomalies to identify new threats or false alarms and iterate on your WAF rules to defend accordingly. Your runbooks should be reviewed, practiced, and updated on a regular basis. We recommend exercising runbooks on a regular basis to ensure that your teams are comfortable responding to security events. For example, simulate security incidents so that operators can practice these procedures.

Consider scheduling regular penetration tests to ensure that you stay on top of the latest threats to address vulnerabilities. It is important to keep WAF rules up to date to defend against newly identified threats. To reduce this effort, you can use managed rules instead of custom rules. Managed rules are updated by the WAF vendor (AWS or partner) based on the evolving threat landscape. However, it is also important that you

update your application specific custom WAF rules to match changes made to your applications.

Cost Considerations

AWS WAF offers standalone pricing that is charged based on your usage of web ACLs, rules, and the number of requests that are inspected. For logging configurations, you will be charged based on your usage of Amazon Kinesis Data Firehose. If you choose to use WAF managed rules from the AWS Marketplace, you can subscribe to managed rules and pay only for what you use. There are no contracts or subscription commitments as managed rules are charged by the hour.

For workloads with high volumes of requests, consider evaluating AWS Shield Advanced to reduce the per request charges. When AWS WAF is used with resources protected by AWS Shield Advanced, there is no additional charges for using AWS WAF and AWS Firewall Manager. You simply pay for the charges associated with AWS Shield Advanced. This approach can help optimize cost for request-heavy workloads. For more details on pricing, see AWS Shield, AWS Firewall Manager, and AWS WAF pricing pages.

Conclusion

This paper describes the various threats that AWS WAF can address, presents the different requirements that should be considered when implementing a WAF and how to deploy AWS WAF to production environments with minimal disruptions. Anyone tasked with protecting web applications can use this whitepaper to better understand how to leverage and implement AWS WAF as a part of their protection tools.

Contributors

Contributors to this document include:

- Achraf Souk, Principal Solutions Architect, Edge Services
- Tino Tran, Principal Solutions Architect, Edge Services
- Damindra Bandara, Senior Security Consultant, Professional Services

Further Reading

For additional information, see:

- AWS WAF Documentation
- AWS Security Incident Response Guide
- AWS WAF Security Automations Solution
- AWS Best Practices for DDOS Resiliency

Document Revisions

Date	Description
May 2020	First publication